The Design of COVE: a Collaborative Ocean Visualization Environment

Keith Grochow

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

University of Washington

2011

Program Authorized to Offer Degree: Computer Science & Engineering

University of Washington
Graduate School

This is to certify that I have examined this copy of a doctoral dissertation by

Keith Grochow

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Chair of the Supervisory Committee:

_____
Edward Lazowska

Reading Committee:

_____
Edward Lazowska

_____
James Fogarty

_____
John Delaney

Date: _____

In presenting this dissertation in partial fulfillment of the requirements for the doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of the dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to ProQuest Information and Learning, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, 1-800-521-0600, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature_____

Date_____

University of Washington

Abstract

The Design of COVE: a Collaborative Ocean Visualization Environment

Keith Grochow

Chair of the Supervisory Committee:
Professor Edward Lazowska
Computer Science & Engineering

Ocean observatories, exemplified by the NSF Ocean Observatories Initiative (OOI), aim to transform oceanography from an expeditionary to an observation-based science. To do so, new cyberinfrastructure environments are helping scientists from disparate fields jointly conduct experiments, manage large collections of instruments, and explore extensive archives of observed and simulated data. However, such environments often focus on systems, networking, and databases and ignore the critical importance of rich 3D interactive visualization, asset management, and collaboration needed to effectively communicate across interdisciplinary science teams.

This dissertation presents the design, implementation, and evaluation of an interactive ocean data exploration system designed to satisfy the unmet needs of the multidisciplinary ocean observatory community. After surveying existing literature and performing a multi-month contextual design study that included input from scientists at multiple institutions, I propose a set of guidelines for the system's user interface and design. Motivated by these guidelines and informed by close collaboration with multidisciplinary ocean scientists, I then present the Collaborative Ocean Visualization Environment (COVE), a new data exploration system that combines the ease of use of geobrowsers, such as Google Earth, with the data exploration and visualization capabilities of sophisticated science systems.

To validate COVE's design, I evaluated its capabilities in three ways. (1) User studies showed that it works efficiently for expert and novice data explorers as well as visualization producers and consumers. (2) Multiple real-world science deployments, both on land and at

sea, saw it replace existing systems for observatory design, provide faster and more engaging planning and data analysis for science teams, and enhance mission preparation and navigation for the ALVIN research submarine. (3) An analysis of COVE over local, server and cloud-based resources indicated that its flexible work partitioning architecture is essential for real-world observatory data analysis and visualization tasks.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ALVIN: A manned, deep-ocean research submersible operated by the Woods Hole Oceanographic Institution (WHOI), which allows two scientists and one pilot to dive for up to nine hours at 4500 meters collecting images, samples, and data measurements.

API: An **A**pplication **P**rogramming **I**nterface implemented by a software program to enable interaction with other software, similar to the way a user interface facilitates interaction between humans and computers.

AUV: An **A**utonomous **U**nderwater **V**ehicle is a research platform that travels underwater without requiring input from an operator.

AZURE: Microsoft's cloud computing platform that provides software developers with on-demand compute and storage for Web applications running in Microsoft datacenters.

BATHYMETRY: The depth of the seafloor relative to sea level. A bathymetry set is the collected depths for a section of seafloor (usually collected via sonar), which provides underwater terrain for ocean data visualization systems.

CONTEXTUAL INQUIRY: A user-centered method that calls for one-on-one observations and user interviews regarding work practice in naturally occurring context wherein the users' daily routines or processes are discovered.

CTD: A commonly used oceanographic sensor set that measures salinity via sea water **C**onductivity, **T**emperature, and **D**epth via an equation involving pressure as well as sampled salinity and temperature.

CYBERINFRASTRUCTURE: A set of systems designed to enable virtual observatories by providing advanced data acquisition, storage, management, integration, visualization and other computing and information processing services, usually over the Internet.

GEOBROWSER: A software system, exemplified by Google Earth, that lets users navigate over a 3D model of the earth's geography and provides the ability to layer geolocated data, such as images, over the geography.

GEOLOCATED DATA: A dataset that includes information about the geographic location of data points, usually consisting of latitude, longitude, and altitude or depth. This may also be referred to as *geo-positioned* or *geo-referenced* data.

MAP PROJECTION: Any method of representing the surface of the earth on a plane. All map projections distort the surface in some fashion, and different map projections exist in order to preserve some properties at the expense of others.

MATLAB: A numerical computing environment and programming language developed by MathWorks that is widely used in academic and research institutions and provides matrix manipulations, algorithm implementation and plotting of data and functions.

MBARI: **M**onterey **B**ay **A**quarium **R**esearch **I**nstitute is the largest privately funded oceanographic organization in the world and acquires data through fixed and mobile instruments, ship based cruises, and occasional large-scale, multi-institute projects.

METADATA: A description of other data. Metadata provides information about a certain item's content. For example, an image may include metadata that describes how large the picture is, the color depth, the image resolution, when the image was created, etc.

NetCDF: **Net**work **C**ommon **D**ata **F**orm is a set of software libraries and machine-independent data formats that supports the creation, access, and sharing of array-oriented scientific data. CF-Metadata is a supplementary standard for defining geolocated NetCDF files commonly used in the earth sciences.

OCEAN OBSERVATORY: A suite of instruments and sensors deployed in the ocean with long-term power supplies and permanent communications links to collect data for use by ocean scientists via interfaces and APIs over the Internet.

OCEAN MODEL: A simulation of ocean processes that is based on features such as bathymetry and accepted atmospheric, thermodynamic, and physical laws. Models are

used to generate a 3D grid of attributes, such as temperature or current velocity over a set of time steps.

OOI: The **O**cean **O**bservatories **I**nitiative is a multi-year NSF program to develop a continuous viewing platform in the oceans consisting of three major science environments: a global observatory based on buoy and satellite data, a coastal observatory near shore, and the cabled regional observatory (see RSN) for deep water exploration.

PARTICIPATORY DESIGN: An approach to design that attempts to actively involve all stakeholders in the design process to help ensure that the final product or system meets their needs and is usable.

REST API: A **RE**presentational **S**tate **T**ransfer API is a programming interface for getting information content from a Web site by taking advantage of existing technology and protocols of the Web, including HTTP and XML, for simplicity and performance.

ROV: A **R**emotely **O**perated **V**ehicle is an underwater research platform controlled and powered from the surface by an operator via an umbilical connection or remote control.

RSN: The **R**egional **S**cale **N**odes is a cabled ocean observatory being constructed off the coast of Washington and Oregon. It spans hundreds of miles via cables connecting to a set of primary connection points (nodes) that support secondary cables, instruments and moorings.

SCIENTIFIC WORKFLOW: A set of discrete computational components connected in a sequence to carry out scientific computation. Workflow systems, such as Microsoft Trident, provide interfaces to design, execute, archive and share scientific workflows.

VERTICAL SECTION: A cross section of the ocean from the sea surface to the seafloor, providing a continuous plane of values for ocean parameters, such as temperature or salinity. These may be extrapolated either from observed data or an ocean model.

VIRTUAL OBSERVATORY: A suite of software applications that lets users uniformly find, access, and use resources – data, documents, software, processing capability, image products, and services – from distributed product repositories and service providers.

XML: E**X**tensible **M**arkup **L**anguage is a text-based data format for Internet files. Unlike fixed formats, such as HTML, XML is a meta-language – a language for describing other languages – used to create document specific languages.

DEDICATION

To Jim Gray for his vision and to Jane Raffan for her constant love and support.

ACKNOWLEDGMENTS

First, I would like to thank Jim Gray for starting this adventure. Much has been written about what a wonderful person he was and what a huge loss it was for the scientific community when he was lost at sea in 2007. All of it is true. He showed me that the smartest people can also be the nicest. Next, I would like to thank Ed Lazowska who was involved in this work from the beginning and has provided support and direction throughout.

This dissertation would have been impossible without the support of the University of Washington School of Oceanography and, in particular, John Delaney and Deborah Kelley, who were always champions for COVE. Mark Stoermer and the wonderful group at Center for Environmental Visualization – Shawn Thomas, Hunter Hadaway, Donald Averill and Dave Collins – were always available to help me make the system better. I would like to thank the ocean modeling team – Parker MacCready, Neil Banas, and David Sutherland – and the team at the Monterey Bay Aquarium Research Institute – Jim Bellingham, Michael Godin, Mike McCann and Kevin Gomes – for data and technical advice. I want to express my appreciation for the science teams and ships' crews I worked with on the deployments at sea and, in particular, on the ALVIN cruise, PI Jim Holden, ALVIN expedition leader Bruce Strickrott and my dive partners – John Jamieson and Mark Spear – for giving me the adventure of a lifetime.

I'm very appreciative of the great support I received from Microsoft and, in particular, Microsoft's Technical Computing Initiative – Roger Barga, Jared Jackson, Nelson Araujo, Dean Guo – and Microsoft Research – Catharine van Ingen and Cathy Marshall. From the University of Washington, I would like to thank James Fogarty for his user interface feedback and guidance, Charlotte Lee for offering her time regarding contextual inquiry, and Bill Howe for his work with me on the architectural analysis. I would especially like to thank Tapan Parikh, whose great work in interdisciplinary interface design was a constant inspiration and guide. Funding for this work came from the University of Washington

Department of Computer Science & Engineering, Microsoft Corporation and the National Science Foundation.

I would finally like to express my gratitude to all my friends and loved ones who never let me forget that I needed to finish, and, in particular, Laura Seaver and Tom Grenon, who let me write at their house while mine was being rebuilt. And, of course, Jane Raffan, who was willing to do all the editing passes necessary, give me constant encouragement, provide a patient ear when I needed it, and help me bring this all to the finish line.

CHAPTER 1

INTRODUCTION

The oceans are an important focus of scientific study and exploration. They cover over 70% of the earth's surface, are intricately connected to its climate and weather patterns, and provide a significant source of food for a large part of the planet. Due to the impact of climate change, understanding ocean processes is increasingly central to predicting how our climate will evolve during the next century and the impact its changes will have on ocean and human health.

To date, our ability to collect data about the oceans has been extremely limited relative to this need, as oceanography has traditionally been an expeditionary science: small crews of oceanographers periodically go to sea in ships to collect data and conduct observations. Ocean observatories that continuously collect and analyze diverse oceanic data offer a new approach. Over the next few years, the National Science Foundation's (NSF) Ocean Observatories Initiative (OOI) will create an ocean observatory of unprecedented scale. The Regional Scale Nodes (RSN) portion of the OOI, installed off the Washington and Oregon coasts, will support thousands of sensors from the ocean surface to deep in the seafloor, connected to cables delivering power and bandwidth. The RSN aims to build a flexible platform allowing scientists from disparate fields to conduct experiments together, providing real-time sensor and data access through the Internet, and creating a vast archive of datasets that span several decades.

The RSN is just one example of large, multidisciplinary earth science projects currently in progress or being considered around the world. These new projects are intended to be transformational in nature – to change the way scientists conduct and communicate earth science. In many cases these projects use no instruments; rather they seek to provide a *virtual observatory* via a shared collection of data, analysis systems, and data exploration tools. In other cases, the projects collect and make available measurements from a set of pre-

configured and statically located instruments. And recently, with projects like the RSN observatory, they are attempting to provide an instrument system that can be dynamically reconfigured and deployed based on the needs of specific experiments and ocean events. All these types of observatories will make vast amounts of observed and simulated data available to a wide range of scientists, encouraging multidisciplinary science and enabling new discoveries that cross traditional science boundaries.

One key to the success of observatories is their computer systems and tools, or *cyberinfrastructure*. Drawing from several areas of computer science, cyberinfrastructures collect, manage, and provide efficient access to the terabytes of accumulated data. For these observatories to transform science, it is also crucial that the cyberinfrastructure provides systems and interfaces that let scientists easily explore and visualize data, plan experiments together, and collaborate across disciplines. A gap analysis of existing literature in the area of visualizing and exploring oceanographic data shows that current tools do not effectively meet these needs. Furthermore, it demonstrates the need for a new system that combines data exploration and visualization, asset management, collaboration, and communication capabilities in an interface that is more suited to the needs of diverse science teams.

To determine detailed user requirements for this new system, I conducted a ten-week contextual design study to gain insight into the current data exploration and visualization practices of ocean scientists. The investigation consisted of multiple site observations and interviews with oceanographers at the University of Washington and the Monterey Bay Aquarium Research Institute. Based on this work, I determined a set of interface guidelines to direct system design.

Motivated by these guidelines and informed by close collaboration with multidisciplinary ocean scientists, I developed and deployed the Collaborative Ocean Visualization Environment (COVE). COVE offers a novel science environment for the observatory: an intuitive, geobrowser-based layering model; point-and-click experiment layout capabilities, and sophisticated data exploration tools that can run interactively on a local machine or over terabyte scale datasets in the cloud.

I then evaluated COVE's usability via three methods: (1) user studies focusing on specific audiences, (2) multiple deployments both on land and at sea with working ocean science teams, and (3) an analysis of COVE over local, server, and cloud-based resources. As described below, each of these investigations validated key design elements and provided insights to improve the system.

With observatories, sophisticated data exploration and visualization is important not only for domain scientists, but also for scientists from other fields, and increasingly for citizen scientists. In a study carried out at the University of Washington, novice and expert users were asked to explore a variety of oceanographic datasets. An instrumented version of the system tracked exploration pathways, which were analyzed along with participant observations, surveys and interviews. Results showed that the integration of rich interactive data exploration capabilities in COVE supported novices and experts, as well as visualization producers and consumers.

COVE's use as an observatory design tool was tested through a multi-month deployment with scientists and engineers from the RSN observatory design team. The objective of instrument layout in observatories is to determine the configuration of primary cables, connection hubs, and instruments that form the infrastructure for future science experiments in a section of the ocean. These choices must be explored in the context of available data, iteratively refined, and presented to constituencies for feedback. In this deployment, iterative prototypes and collected user feedback provided key guidance to validate and improve system design. COVE replaced existing tools and practices by the end of the study and is poised to become the primary design system for future RSN observatory instrument layouts.

Testing COVE's planning and collaboration capabilities occurred via two deployments on scientific expeditions at sea. The first expedition assessed two research sites for the observatory, where COVE supported the creation of high resolution bathymetry and integrated diverse datasets to carry out daily planning, execution, and review of missions. In the second expedition, COVE supported mission planning for manned submarine dives to study volcanic vents on the seafloor. The results of these deployments demonstrated that COVE's easy-to-use interface provided a more effective tool for multidisciplinary scientific

collaboration and engaged a wider range of participants than existing methods employed by the team.

Finally, COVE's scalability was investigated for its ability to handle increasing volumes of data being collected by observatories. In this context it is crucial to explore system requirements for resources at all scales: local Graphics Processing Units (GPUs) for interactive visualization, server-side multi-core machines, and scalable, pay-as-you-go cloud resources. Based on data collected during the contextual design study, I defined a suite of representative visual data analytics workflows. I then analyzed a variety of workload partitioning strategies spanning all three platforms, discussed their tradeoffs and requirements, and evaluated their performance. I determined that no single partitioning strategy is optimal in all cases, and rich visual data analytics requires a flexible, cross-scale architecture such as provided by COVE.

1.1 Thesis Contributions

These are the research contributions that are included in this thesis:

- **A Gap Analysis Assessing the Capabilities of Existing Interactive Science Tools for Observatories** – Based on a survey of relevant literature, I present an overview of the needs for an interactive data exploration system for observatories and an analysis of existing systems and technologies that reveals the need for a new type of science tool.
- **A Set of Design Guidelines for Ocean Observatory User Interfaces** – I conduct a contextual design study with multidisciplinary ocean scientists at multiple institutions in order to determine a set of user interface design guidelines for an observatory data exploration system.
- **COVE: a Collaborative Environment for the Ocean Sciences** – I describe the design and implementation of COVE, a novel science tool created in close collaboration with ocean scientists to meet the design guidelines.
- **Evaluation of COVE for Usability and Real-World Science Impact** – I provide a usability evaluation of COVE, by describing the results of usability studies with ocean

data users and assess its impact in multiple deployments involving observatory design, data exploration and multidisciplinary collaboration.

- **A Quantitative Evaluation of COVE's Scalability to Server and Cloud Platforms** – I analyze COVE's performance across local, server, and cloud resources for a representative set of visual data analysis tasks and present results and analysis that validate COVE's architecture.

1.2 Outline of the Thesis

This dissertation is divided into eight chapters. The balance of the current chapter reviews the domain of ocean data and observatories and its key features. Chapter 2 presents a framework for discussing data exploration needs for oceanographic data and reviews the relevant literature to determine gaps between user needs and existing systems. Chapter 3 outlines a contextual study that informs user interface design guidelines for an interactive ocean observatory data exploration system. Using these guidelines, Chapter 4 describes the design and implementation of COVE, a new geobrowser based system for observatories that provides easy-to-use experiment planning, data exploration, and collaboration facilities in a common visual environment. Chapter 5 evaluates COVE's data exploration capabilities and shows that the system scales well from novice to experienced users and from visualization consumers to producers. Chapter 6 evaluates COVE's observatory design capabilities in a long-term collaboration with the science team planning the RSN cable ocean observatory and discusses COVE's deployment on two ocean expeditions to assess the multidisciplinary collaboration abilities of the system in real-world ocean science environments. Chapter 7 explores COVE's extensibility from local computing environments to network and cloud resources and quantitatively analyzes the system based on a benchmark set of observatory data exploration and visualization tasks. Finally, Chapter 8 reviews the contributions of this research, unaddressed elements, and possible future directions for COVE.

Figure 1.1: One of the earliest published oceanographic data visualizations, created by Benjamin Franklin in 1769 to show the Gulf Stream.

1.3 Observing the Ocean

One of the earliest scientifically published visualizations of oceanographic data was created by Benjamin Franklin in 1769 to illustrate the flow of the Gulf Stream [79]. Shown in Figure 1.1, it was created based on the reported time it took mail ships to make their voyages between England and the Colonies. This figure illustrates some of the traits still common in oceanographic data exploration and visualization. Relatively sparse data points are collected and extrapolated into a model that explains an ocean process. This model is then visualized on a map of some type to give geographic context to the data, which lets other scientists and non-scientists quickly understand and use the information. In this case, sailors could use simple thermometers to see if they were sailing in the warmer and faster currents of the Gulf Stream.

Since then, there have been great advances in systems used to collect ocean data, as described in the many texts on introductory oceanography [79, 98, 102]. Besides ships, data

is collected from buoys and instruments traveling from the surface to the seafloor on their mooring lines, from satellites crisscrossing the planet, and from instruments on mobile platforms. For example, *Autonomous Underwater Vehicles (AUVs)* can travel either under power or passively, through buoyancy control (a *glider*), on a preset course through the ocean collecting data; *Remotely Operated Vehicles (ROVs)* carry out similar tasks under human control. Many of these systems have suites of sensors that are collecting data simultaneously. One of the most commonly used sensors is a *CTD*, which determines salinity, temperature, and depth over time, but many additional features of ocean chemistry and composition are sampled with increasingly sophisticated tools. In addition to these point sensors, deep sea photography and video equipment is now common and scanning devices can measure the distance to objects using lasers (*LIDAR*) or sound (*SONAR*) to create a surface map or volumetric data. All these sensors sample data at a specific time and place and give each data point geospatial and temporal components.

Some important limitations of these systems in collecting ocean observations are worth noting. First, data collection can be slow. Unlike the atmosphere, where light travels relatively easily, oceans allow only limited use of light-based sensors due to attenuation in water and lack of ambient light. Sonar affords longer distance views of a volume or the ocean bottom, but coverage of large areas takes more time because it is attached to ships or ROVs. The ocean environment is also harsh: pressure increases one atmosphere for approximately every 10 meters of depth, the medium is corrosive, and the temperature for seafloor instruments can vary from near freezing to over 400° at geothermal sites. Placing or maintaining an instrument usually requires use of a ship or ROV, and power and data storage must be self-contained if access to a ship or ROV is unavailable. These factors currently make oceanographic data difficult to collect, leading to datasets that are frequently sparse, selective in location, and of relatively short duration.

As computers have become more powerful and data storage more plentiful, oceanographers have built sophisticated simulations of ocean processes based on features such as bathymetry and accepted atmospheric, thermodynamic, and physical laws [87]. An assortment of ocean modeling programs is available to today's oceanographers, such as the

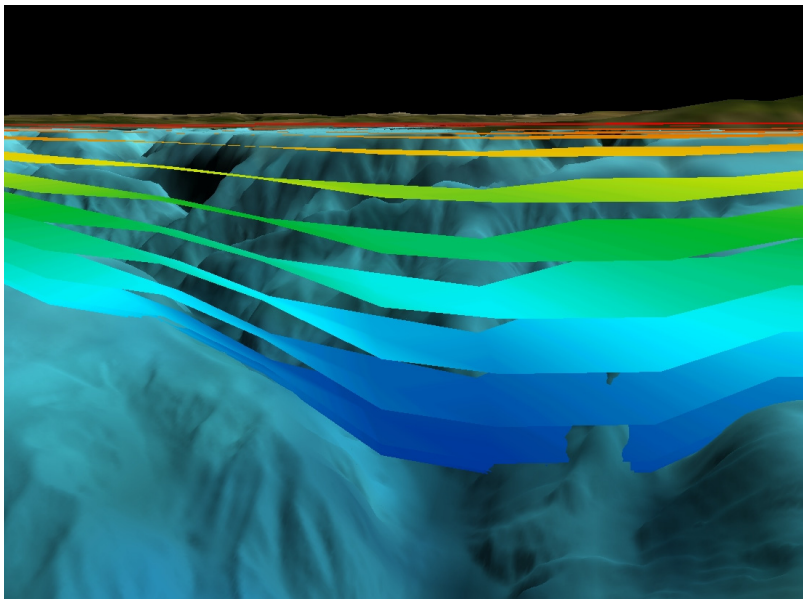

Figure 1.2: An example of depth varying layers in an oceanographic model for Monterey Bay.

*R*egional *O*cean *M*odeling *S*imulation (*ROMS*) and the *H*arvard *O*cean *P*rediction *S*ystem (*HOPS*). These generate a 3D grid of attributes (such as temperature or current velocity) over a set of time steps. These grids are often de-formed to more accurately follow features of the ocean bottom and shore in their specific region, as shown in the depth layers in a model of Monterey Bay depicted in Figure 1.2.

Science computation resources have made it easier to distribute and accelerate calculation of these simulations, but they still may take hours to run. The generated datasets also quickly overwhelm an average PC [33]. For example, a low resolution model of a 300 x 300 kilometer area calculating a point at each kilometer and 20 layers deep is 15 MB per scalar value for each time step. For a set of 10 data values simulated at hour intervals, this represents 3.6 gigabytes for a day of data, or 1.3 terabytes for a year of data. Re-gridding these large models to work with other datasets leads to performance issues and accuracy concerns, stemming from the interpolation necessary to generate attributes at new locations. These modeling programs can also be hard to adapt to a new area because several variables are site specific and extremely sensitive to initial conditions. In addition, due to the

sparseness of observed data and the relatively short time over which much of the data has been collected, there are often insufficient historical measurements to accurately initialize and validate models. As a result of these issues, few ocean regions have detailed models, almost no regions have multiple models running, and results of different models running over the same area may vary greatly from each other and from measurement data.

As more data is produced, and more oceanographers use datasets they did not personally collect or create, it is becoming more important to manage the sharing of data. The easiest method is to copy a file to the local system from a disc or local server, but this is often infeasible for security or size reasons. To let users access data over the internet or to download only portions of data files, one widely used solution is the OpenDAP Internet data portal [23]. This system allows a user to select a set of variables and range of dimensions from a Web interface, such as temperature over a set of time intervals, and download a data file to the local system or directly into analysis packages. The solution has two benefits: (1) it is easy to set up, i.e., just point the system at a selection of files, and (2) the user interface, while limited, is easy to understand. However, the data being shared is static by design, there is no built-in search functionality to locate the desired data, and there is no inherent server-based data analysis capability.

A new approach to better support the collection, management, and sharing of ocean data is an *ocean observatory*. Ocean observatories are instrument arrays and data repositories intended to be in place for decades that answer a range of scientific questions by providing constant intensive monitoring of a location. NSF is investing more than $US 385 million over the next five and half years for the design and construction of the Ocean Observatories Initiative (OOI) [71]. After the platform is in place, it will provide a continuous viewing platform in the oceans over its 20-30 year lifespan, using new and transformative sensor and software technology. The Final Network Design for the OOI consists of three major science environments: a *global observatory* based on buoy and satellite data, a *coastal observatory* near shore, and the *cabled Regional Scaled Nodes (RSN)* for deep water exploration on the Juan de Fuca plate off the coasts of Washington and Oregon. The primary focus for the

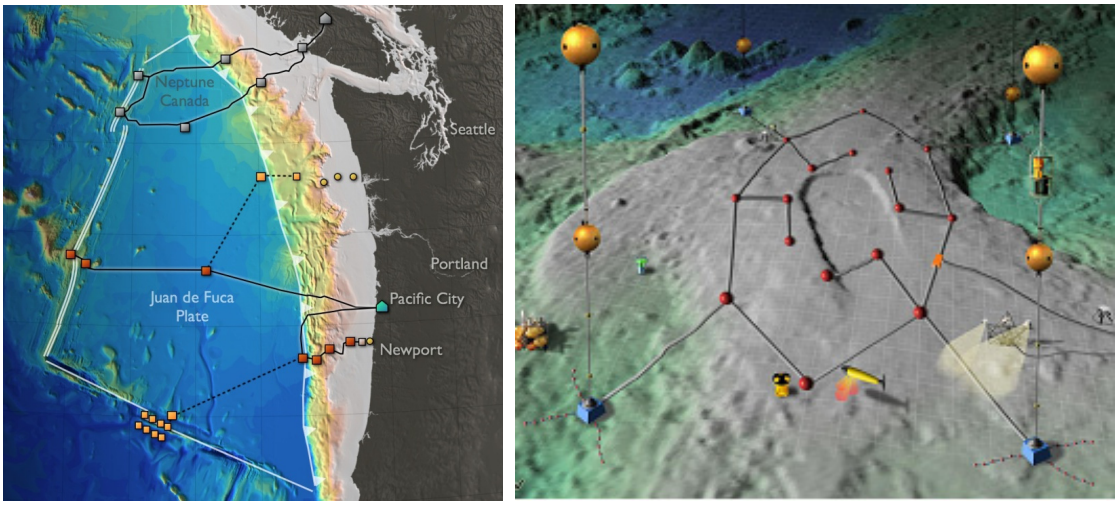

Figure 1.3: On the left is the RSN Observatory layout as of June 2010. On the right is a possible sensor network surrounding the Axial Volcano on the Juan de Fuca Ridge.

remainder of this thesis is the RSN observatory portion of the OOI, which is being designed and implemented at the University of Washington.

The cabled observatory consists of a set of primary cables that supply "hundreds of kilowatts of power, tens of gigabits/second of bidirectional communication and precise, synchronized time to hundreds and potentially thousands of instruments" [60]. These cables support a set of primary experimental sites, whose selection is based on a broad set of scientific goals. At each of these sites, a primary junction box provides a connection point for a complicated set of secondary cables, junction boxes, moorings, and other instruments that will extend the reach of each site by tens to hundreds of kilometers [45]. Figure 1.3 shows the RSN infrastructure and a sample layout for the Axial Volcano site, a geothermal test bed. The network infrastructure is not yet deployed, but a probable model is a local intranet providing data links and control messages to IP addressable instruments [24]. The data collected by instrument sensors will be transported to a shore-side data system for storage and access.

1.4 Ocean Observatory Challenges

What are the challenges of designing systems and interfaces for ocean observatories? Historically, an "observatory" referred to a physical structure on the earth that housed instruments, usually a telescope, used to observe astronomical events. Over time, the definition expanded to include more sciences (e.g., geology, climatology, volcanology, oceanography) and more physical environments (e.g., satellites, connected arrays of permanent instruments, and instruments that are being deployed over time). It has recently been extended again, in the case of *virtual observatories*, to include systems that do not necessarily include instruments but instead focus only on the data and tools that interact with it. As this non-instrument view of observatories is constructive for focusing on their data-driven needs, I will examine a virtual observatory in more detail and then consider the implications of managing the physical instruments. For a sample of observatories, virtual observatories, and other science data portals planned or currently operating, please see Appendix A.

A virtual observatory seeks to increase efficiency and enable new science by greatly enhancing access to data, services, and computing resources available to scientists [64]. It consists of a suite of software applications that lets users find, access, and use resources – data, documents, software, processing capability, image products, and services – from distributed product repositories and service providers. A virtual observatory can have a single subject, such as the Virtual Solar Observatory, or a theme, such as the US National Virtual Observatory supporting astronomy.

The primary virtual observatory interface typically takes the form of an Internet portal offering users such features as:

- Tools that make it easy to search and retrieve data from archives and databases
- Tools for data analysis, simulation, and visualization
- Tools to compare observations with results from models, simulations, and theory
- Additional information: documentation, user guides, reports, publications, news, etc.

Figure 1.4: An example of the assets in use in Monterey Bay for the 2003 AOSN project.

Some current oceanographic projects function as virtual observatories by adding a selection of these services to the OpenDAP data portal described earlier. For example, multidisciplinary teams share instruments and data to carry out research in Monterey Bay as part of the Autonomous Ocean Sampling Network (AOSN) program. This project consists of a series of multi-month activities to measure the effectiveness of adaptive sampling in Monterey Bay by collecting and assessing data from different instruments and platforms, as shown in Figure 1.4. Observed data collected from the instruments and simulated data from multiple ocean models are provided to scientists in a data repository, along with a set of pre-configured visualizations for daily planning. After the initial collection period, the data is provided for historical use via a keyword-based search interface. A similar tool was deployed at sea, with added options for server-side data analysis, to support a multi-organization program in 2010 to study the interaction of typhoons with the ocean surface.

In both programs, the virtual observatory provides data and visualizations, but it does not provide integrated services for planning and monitoring asset deployment. The ocean

observatory incurs not only the challenges of the virtual observatory, but also those of instrument management. This includes the need to make decisions about where to situate major experiment sites and the design of a cabling plan that will support these sites given environment, timing, and cost constraints. There are instruments under observatory staff supervision to collect long running data, as well as instruments and cabling deployed for specific experiments monitored by external ocean scientists.

Given the importance of ocean observatory tasks, the capital expense, and time devoted to building these science platforms, it is crucial that scientists have effective tools for interacting with observatory data and instruments. However, unique user interface challenges are associated with observatories, which make it difficult to find appropriate tools. Based on the literature and interactions with scientists (explained in more detail in Chapter 3) these challenges are briefly described below.

1.4.1 Diverse Data Exploration Needs

With a variety of sensors, many data types and datasets are available. There will be observed data collected by the observatory instruments, as well as multiple simulated datasets for the observatory area. There will be data for long time scales and large geographic regions. All this data will be accessed and explored by a diverse set of users: domain scientists to carry out primary research, non-domain scientists to support inter-disciplinary projects, and citizen scientists for general interest. Interfaces must be designed to scale fluidly across the needs and capabilities of all users.

1.4.2 Planning and Managing Instrument Layout

The RSN differs from other types of earth observatories. *Satellite-based* observatories usually consist of one instrument with a small set of sensors collecting multi-spectral images of the earth. The instrument is expensive to deploy and to modify or update after it is in place. The incoming data stream is homogeneous (images), and the focus of the data system is on cleaning sensor data and managing derived data products based on scripts [27]. Another type of observatory is a *terrestrial sensor network*. These systems can consist of

thousands of independent sensors that often communicate wirelessly and collect various types of data. Placement and maintenance is relatively inexpensive because humans can usually travel to the site to change layout and sensors [17].

A cabled ocean observatory has challenges from both models. Like sensor networks, it consists of many sensors spread over a huge area for collecting a variety of data types. Like satellite-based observatories, adding new capabilities to the system, while less costly than current options, remains expensive and must be carefully managed. However, the sensor environment is more complex than either of these two models: hundreds of sensors collect data at any given time and can independently change state based on detected events or be completely re-tasked to focus on a major ocean event, such as a major volcanic eruption.

1.4.3 Multidisciplinary Science Collaboration

While there must be strong support for traditional oceanographic disciplines, ocean observatories require extensive collaboration by multi-disciplinary teams. Recent exciting discoveries in oceanography (e.g., geo-biology) show the value of scientists working together across conventional domain boundaries. Many important problems, such as climate change, are inherently multi-disciplinary as they involve the effects of the planet's systems on each other. Finally, a primary goal of the observatory is to optimize return on investment by providing support for new ways of doing science. Interfaces that reduce impedance in communication across specialties are therefore crucial.

1.4.4 Data Repository Size

Szalay and Gray [99] discuss issues associated with the massive data repositories that will be created and stored by these observatories. They point out that most scientific data today, if shared at all, is hosted on an Internet service that delivers only simple storage capabilities. On more advanced systems, metadata (information about data, such as format and history of the data) may be available to help find appropriate datasets. Once located, data files must be downloaded to a local system for further computation. They argue that this approach is impractical as data repositories continue to grow by orders of magnitude. With terabyte-

scale data environments, it becomes imperative to leave data on the server whenever possible to avoid being overwhelmed by transfer time. This implies the necessity of hosted data environments and distributed data processing scripts (or *scientific workflows*) that return final data products and visualizations only after computation has been completed at the data source. Rather than transmitting all underlying data, only questions and answers would be sent back and forth.

Several researchers have created systems to explore the needs related to large data repositories in more detail. One example is the Earth Science Workbench from UCSB [27], a flexible data system with file-based data, metadata cataloging, individual scripts for computation on satellite imagery, and individual user data spaces accessed through a Web interface. The HEDC experimental data center for the RHESSI spacecraft [96] supplies similar services, with the addition of approximated data results. These are compressed datasets where information has been removed, allowing scientists to gain insight interactively before committing to the complete dataset to validate results. On the oceanographic front, the OurOcean ocean monitoring site [53] includes the ability to take data from multiple sources and deliver on-demand ocean simulation capabilities, allowing users to configure their own ocean models. This service was used by MBARI for the AOSN program discussed earlier. The more recent LEAD portal as described by Plale *et al.* [77] can also build, run, and monitor computation for scientist specific data products. In each system, analysis has been moved to the server, but data exploration and visualization continues to require the downloading of datasets to local workstations for use in a separate application.

CHAPTER 2

RELATED WORK AND GAP ANALYSIS

Data exploration and visualization systems are particularly important to oceanography. With sparse historical data comes a greater need for human expertise to fill in gaps and identify important trends. Until science better understands the oceans, human experts will be required to assess the validity of oceanographic models. With the RSN observatory, in particular, experimental sites can only be explored virtually and scientists from diverse fields must deal with new types of data. Effective visual tools are thus crucial to let scientists quickly observe patterns, note anomalies, and determine the validity of data based on its environmental context.

This chapter first defines a set of core ocean observatory data types to provide background for later sections. Second, it categorizes data exploration tasks and reviews relevant research about these tasks and ocean observatory needs. Third, the research review is continued to determine possible data exploration and visualization systems to carry out these tasks. The chapter concludes with a gap analysis of systems with regard to their ability to complete tasks and the user costs for each system type based on a usability cost metric.

2.1 Data Types

Categorizing and comparing data by type is a common approach to exploration, and various models have been proposed. MacKinley [58] defined three basic categories of data – nominal, ordered, or quantitative – to indicate data with no natural ordering, with ordering but no value, and with mathematical values. Using these categories, he defined a grammar for the automatic creation of appropriate 2D representations. Card *et al.* [18] built on this work by also categorizing marks, retinal, and positional features of a representation to

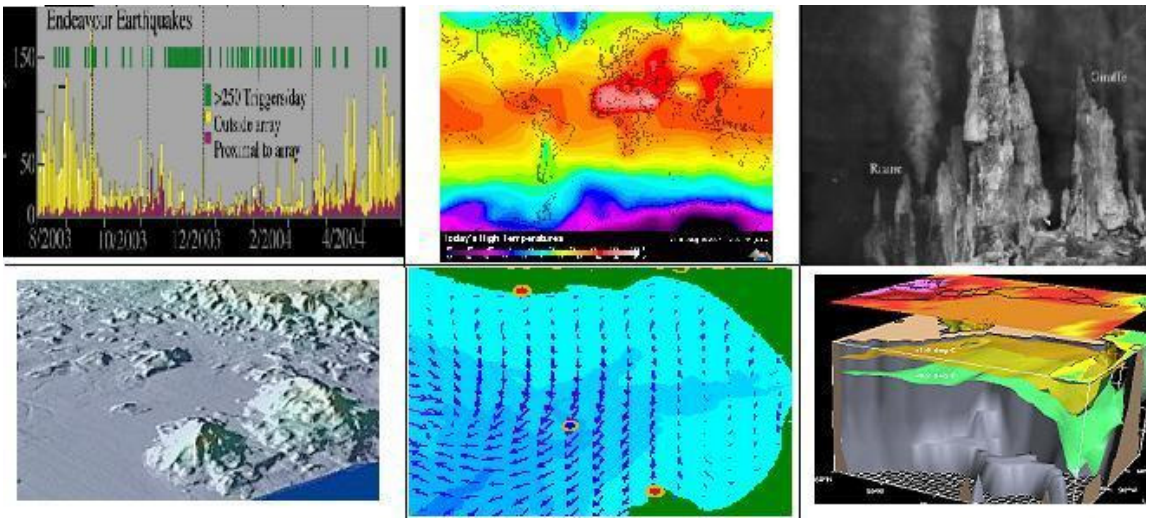

Figure 2.1: Examples of common oceanographic data types. The top row includes a time series plot, an interpolated point set of global temperature, and an image of a geothermal site. The second contains a bathymetry scan, a vector field of surface currents, and a multi-layered ocean model.

classify its type. Schneiderman [89] proposed a simple *task by type* taxonomy with seven data types and seven common user tasks for information visualization.

Based on oceanographic texts [98] and Schneiderman's approach, seven primary data types and examples of each are described below. Representations of six of these types are shown in Figure 2.1.

Time Series: A set of scalar data points with a monotonically increasing time component. This is the basic data type collected by a positional measuring device on buoys, ships, AUVs, and ROVs. For example, a CTD on a glider measures 3 scalar values – salinity, temperature, and pressure – at a point as it traces a path through the ocean. In addition to quantitative data, nominal data can be recorded, such as tagged species that are tracked over time.

Spatial: A set of scalar data points with possibly equivalent time components, often constructed from time series collected by several instruments. Examples are global ocean temperatures collected from buoys, locations of multiple species tagged from video records, or seismic events triangulated from a sensor array.

Image: A 2D spectral map collected by 2D sensors usually in still or video cameras. For example, cameras are used on a submersible to return images and video of cable routes or

bottom features from an ROV, or at specific underwater locations, such as an underwater volcano. Satellite images are used to investigate primary production in the food chain by visually detecting chlorophyll from algae blooms.

Height Field: A 2D map with data values signifying distance. To capture bathymetry for the ocean floor, LIDAR can be used over short distances and SONAR for longer distances. In addition to detecting terrain features, split-beam SONAR can be used to detect pressure changes in the ocean caused by swim bladders for schools of fish. These datasets may require extensive post-processing to generate viable data products.

Vector Field: 2D or 3D directional vector sets indicate movement of sample points over time, based on currents in the ocean. Flow can be measured with a current meter, or at the surface using *CODAR* (COastal raDAR). Vector fields can also be produced by computer models in 2 and 3 dimensions.

Volumetric: Multiple scalar values distributed in space for some given time that are generated by computer models or interpolated from data collected by sensors. For example, zooplankton that lives off algae can be measured by towing a fine net, or objects can be detected over a volume with SONAR.

Sensor Network: A set of records that represents current and historical states of the sensors in the ocean observatory. It includes the location and state of each sensor and instrument, connections between instruments, associated metadata and operational history.

The first 6 data types are common in general scientific visualization, and many techniques for viewing them effectively have been devised. The last data type, an important addition for the ocean observatory, may be viewed by itself or as additional context when exploring other data types.

2.2 Data Exploration Tasks

Scientific data analysis is the process of distilling potentially large amounts of measured or calculated data into a few simple rules or parameters that characterize the phenomenon under study [92]. A common approach to defining tasks in this process is to create a flowchart-like representation. For example, Langley [50] categorizes research discovery tasks in several

systems in order to determine successful automation strategies. His stages include observation, data exploration, model creation, simulation, verification, and communication.

Springmeyer *et al.* [92], on the other hand, presented a much less structured set of tasks by closely observing scientists in their daily data environment. Just as Schneiderman identified seven tasks that commonly occur in general information visualization *(overview, zoom, filter, details, relate, history, extract*), Springmeyer identified a similar set of scientific data exploration tasks through observing scientists interact with datasets. Table 2.1 lists a set of exploration tasks based on this work.

Table 2.1: Scientific data exploration task categories identified by Springmeyer.

Task	**Description**
Generation	Create a visual representation of data using systems and tools.
Examination	Examine attribute-attribute and attribute-time correlations.
Orientation	Modify orientation of a data representation to get a different perspective.
Comparison	Compare datasets with each other to identify relevant differences.
Queries	Find appropriate datasets and select subsets of attributes from data.
History	Store information on the history, processes, and insights involved.

Below, I present a more detailed description of each task based on Springmeyer's categorization. To better understand potential strategies for accomplishing the task, I discuss current research for specific task aspects guided by the observatory data exploration needs noted in Chapter 1. The *Generation* task, addressed separately in the next section of this chapter, lists relevant data exploration and visualization systems and tools.

2.2.1 Examination

Examination is the act of visually inspecting an observatory dataset representation. A basic examination task is to examine a plot of two or more attributes of data to discover correlations. 2D scatter plots compare two selected attributes and may integrate additional

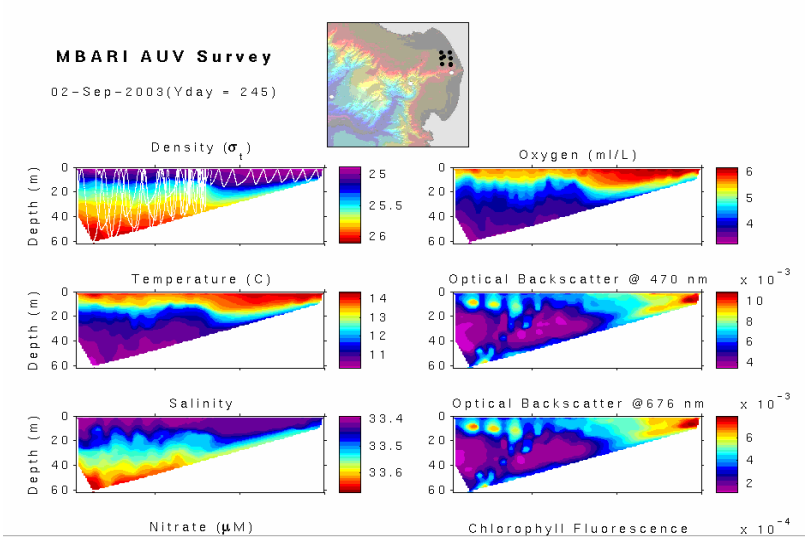

Figure 2.2: 2D plot visualization of CTD data collected from AUVs.

dimensions through use of color, glyphs or other marks [58]. The AUV survey plot (shown in Figure 2.2) is a typical example. Different variables (each plot) are presented for each depth (y axis) along the track of the AUV (x axis) and color is used to indicate an attribute's value. A common oceanographic convention is to interpolate data points, creating a colored mapped plane to ease visual inspection. Adding a third spatial dimension to the presentation creates a 3D plot; the challenge here is that the presentation medium is usually 2D, such as a computer monitor, so user interaction is required to deal with possible occlusion by changing the viewing angle or removing occluding layers.

Given the importance of understanding correlations among variables, much research has gone into automatically generating effective plots. Tufte provides a basis for much work in this area [104]. He presented several data visualizations, both effective and also misleading, to determine rules that can be derived from their examination. MacKinley's work [58] defines a grammar to generate optimal data visualizations, based on data types, following these rules. And recent research, such as Stolte's Polaris project [95], has shown how this grammar can be used to automatically create plots for multi-dimensional data stored in pivot tables.

Setting a plot attribute to ‘time’ is particularly important for a time series. Beyond two dimensions, however, effective examination of time is challenging, and researchers have proposed various techniques. The LifeLines system for visualizing personal histories by Plaisant *et al.* [76] uses a merged screen of time bars for events, with controls provided for filtering and scaling their representation. Geotime [48], on the other hand, represents time as vertical vectors in a 3D environment to detect correlations. Both approaches are based on event visualization and do not scale well to planar or volumetric datasets. Given the limitations of static presentations, animation is often a more appropriate method to display important temporal features. Bederson and Boltman's findings [12] showed that animation effectively helps users create mental maps of changes over time when it is relevant to the task.

The ability to examine correlations between geographic locations and attributes, or observatory assets, is often important. Figure 2.3 shows the possible output of such a representation from the RSN observatory for seismic data. The addition of bathymetry delivers important context for scientists. Contour lines can be added to give further details about the bathymetry, and color maps based on height are an important consideration. Brewer noted that careful use of color is particularly important in interactive and animated map context, where users must monitor changing patterns and have little time to refer to a legend [57]. Rainbow or spectrum variants provide the most differentiation, single color gradations provide the most map-like appearance, and shading is an important feature in either case to effectively determine features.

Examining time-varying simulated data from ocean models for variables such as temperature or salinity is a particularly challenging task. Iso-surfaces can present the boundary of a certain value in a volume, but looking at more than one scalar value requires cutting planes or more sophisticated techniques, such as cutaways, transparency, or exploded views to show occluded layers [54]. Managing the visualization to best use these tools can challenge novices. Viola's work on importance-driven volume sampling provided a way to automatically determine how to most effectively remove layers for the user [111]. More

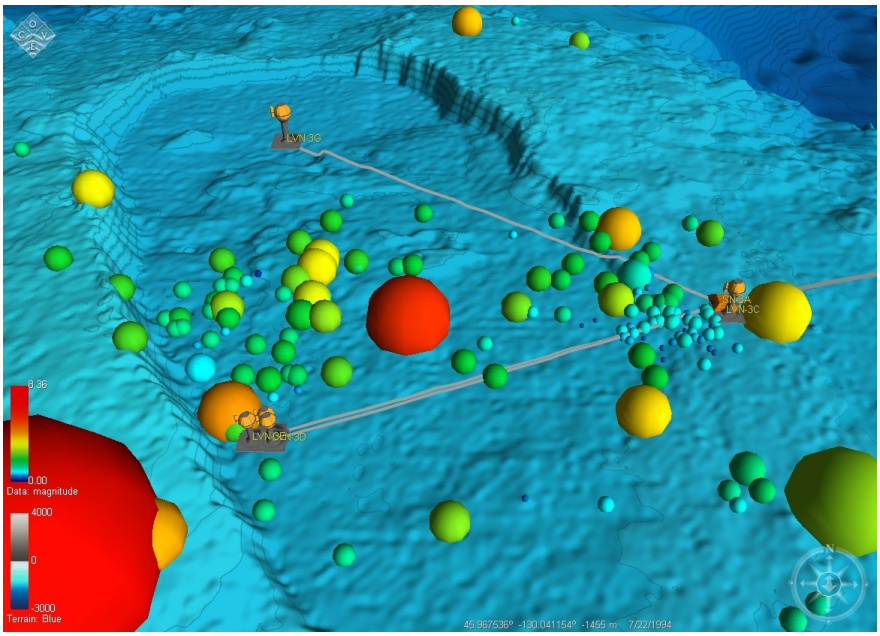

Figure 2.3: A representation of earthquakes at the Axial site of the RSN with bathymetry and instruments for context.

recently, work by Li *et al.* described a conceptual basis for visualizing layered models and showed how rigging for intelligent cutaways can help user interaction [55].

Vector data has been historically quite difficult to view effectively. Gaither and Moorhead furnished a list of techniques common to oceanography [28]. The simplest methods, such as hedgehog displays – representing instantaneous direction and force with arrows – are easy to provide but require care. Displaying several concurrently soon turns to noise, direction can become confusing, and scaling can often lead to what Tufte referred to as a "visualization lie" by causing a nonlinear change in appearance based on viewing distance [104]. More advanced methods, like flow lines that represent continuous movement over several time steps, can help with the visualization problem, but can also increase occlusion. Another approach to visualize flow is presented by Van Wijk; he warped (or advected) texturing to present visual clues about changes in the vector field over time, by presenting simulated flows of white noise patterns [109].

Information about the observatory itself must also be examined. Doing so ensures that the instruments and sensors are located and operating as expected. It also lets users see data in

the context of instruments (Figure 2.3) to determine coverage and plan future locations to modify data collection strategies. For the RSN, this includes instruments and sensors as well as cabling and connection plans to manage power and bandwidth to a site. Because cabled observatories are so new, little research is available for these interfaces. The 2D observatory viewers (discussed in section 2.3.4) illustrate systems for examining sensor networks.

2.2.2 Orientation

Orientation involves altering the view of a data representation to examine the data from a different angle. The simplest form changes the axis in a plot or the viewing position and direction in a 3D environment, but it also includes adding other viewing aids, such as labels, legends, directional symbols, or other navigational aids for the user.

Extensive research has focused on understanding user needs and solutions for navigating 3D environments, beginning with Sutherland's work [97] on head-mounted displays. A task-based taxonomy of interaction techniques in 3D spaces by Tan *et al.* [101] showed that navigation in 3D spaces can be simple and effective if reasonable affordances and constraints are provided. Darken *et al.* presented a toolset for navigating virtual environments [21] and indicated how tools and environmental and contextual clues can be especially useful. For geolocated data, the most obvious clues are representations of geospatial environments.

A zoom-able user interface was presented by Bederson and Hollan in their Pad++ as a model for navigating large, computer-based datasets [14]. It let users zoom in to reveal more detail and zoom out to provide more context, much like a camera's zoom lens. Their work has continued to evolve with the release of the Piccolo and Jazz tool kits for zooming interfaces [13]. Other authors proposed improvements to the original design: providing cues based on interesting parts of the data to aid users in regaining context when usual landmarks are lost [47], making it easy for users to temporarily jump to a home or bookmarked location [75], or providing a context layer and a tree hierarchy [78].

Hornbæk *et al.* [44] showed that zoomable user interfaces are especially effective at helping users maintain context in geospatial environments. The success of this approach is demonstrated by the millions of users of Google Earth's zoomable 3D interface [31]. When

the geographic context of an environment is visible, there is less chance of getting lost; when landmarks become less available, such as the bottom of the ocean, this problem returns. Here, other georeferenced marks, annotations, maps, images, and the terrain itself can help provide context. Other assets are instrument locations and cable layouts of an observatory.

Beyond simple camera control, however, it is necessary in large and complex environments to furnish users with navigational aids to help them discover the most useful data. Gaylean [29] presented work on structured navigation to assist users through a scene. His river metaphor outlines a planned trail to guide but not rigidly restrict users to pre-planned views. Another approach was presented by Snavely *et al.* [91] in their work on photo-tourism for image collection and video navigation, where affordances were defined to transition between a large number of geolocated images.

2.2.3 Comparison

Comparison lets scientists identify differences between datasets. This can be as simple as looking at two representations of data next to each other, or more sophisticated, such as using tools to transform data in a controlled way across multiple windows to determine correlations.

One way to make comparisons is to place multiple datasets in the same window. Datasets can be overlaid directly on top of each other or offset in space or time. Contours or tick marks are useful tools to quickly observe differences visually. Wood *et. al.* used Google Earth as a front end to let users quickly overlay geolocated dataset visualizations to study their interaction with a large geolocated database [115]. They found this approach successful for exploration and sharing of visualization of non-science data across casual users. Nath *et. al.* used Microsoft Virtual Earth to show an example of overlaying real-time sensor data with geographic landmarks [69].

Another way to facilitate comparison is by using multiple windows to show different datasets or different aspects of the same data. Ideally, these are synchronized to automatically update all relevant views. Tufte provided much of the original insight in this area, examining tables of static images to present the importance of multiples in space and

time [105]. Chi discussed key principles for information visualization spreadsheets that provide a basis for effective interaction with these representations [19]. Roberts' research on multi-view and multi-form visualization [82] investigated useful approaches for linking dissimilar views based on common underlying data.

2.2.4 Queries

Beyond altering their point of view and comparing visualizations, scientists use queries to find appropriate datasets and view a subset of attributes. Queries may include data manipulation tasks such as filtering to remove data or aggregation to summarize across multiple datasets. The standard approach for querying datasets is via a dialog using a specific query language, perhaps adding dialog controls to simplify the task.

Visual query approaches let users carry out these tasks interactively with immediate feedback from the environment. The goal is for users to interact with data fast enough to make the underlying database queries transparent and exploration fluid. The process of visual querying is most easily defined by Schneiderman's mantra, *"overview, zoom and filter, details on demand"* [89]. One of the earliest systems that showed the effectiveness of visual queries was IVEE [2], which included sliders to quickly scrub and filter queries to a movie database. Valéria *et al.* [108] showed how this approach could be used with geospatial systems to build fast query systems. Recently, Shen *et al.* [88] applied a similar model to oceanographic data by rewriting the ROMS ocean modeling code to make it responsive to sliders for a low resolution model of salinity and temperature in the Chesapeake Bay and to let users carry out interactive queries.

Authors have shown that direct manipulation of visualizations can be much more effective than query languages for filtering and aggregation tasks as well. Stolte's work [94] with Polaris allowed brushing, or dragging, in the window to select a subset of values. Baudel extended this visual approach by providing ways for users to directly edit database values for certain operations [10]. By identifying bidirectional operations with the database, he showed the effectiveness of interaction for directly editing datasets.

2.2.5 History

The final task considered is storing a history of information processes and insights associated with data exploration. A history allows scientists to record information for both themselves and other scientists. I consider three different types of history: (1) implicit ones created by the user's actions, (2) explicit ones created by annotations, and (3) data transformation artifacts, such as workflow.

Implicit history, created by users through their actions, enables undo and redo of user actions. Exploration requires safe movement through various tasks and the ability to return to a previous context easily [89]. Kreuseler *et al.* [49] describe various ways to capture this kind of history (e.g., storing actions, storing actions and inverse actions, storing state, and storing state change) depending on the data system features. They further investigated storing user actions using XML to integrate a visual history mechanism within a data exploration framework.

A more explicit history method lets users enter notes and descriptions while working with the visualization. As pointed out by Heer *et al.* [38], this type of asynchronous collaborative support had been largely overlooked by researchers. Their Voyagers and Voyeurs project demonstrated the effectiveness of a rich annotation environment for sharing insights on demographic visualizations. Visual exploration of provenance and annotation was examined by Groth and Streefkerk [36]. Their system stored changes in orientation and user generated annotations to allow recording and replay of interactive sessions.

Finally, artifacts may be preserved in the form of workflow descriptions that identify how data has been transformed. To better categorize workflow, Jim Gray's description of NASA data levels is useful [33]. Level 0 (*L0*) represents the raw bits coming from the sensor; level 1 (*L1*) is the cleaned and calibrated dataset, and level 2 (*L2*) is the derived data products from *L1* data. This creates two primary areas of workflow: transformation of *L0* data from the sensor to an *L1* data product, and creation of *L2* data products derived from *L1* datasets. The first type of workflow is often an automatic process or script of the data system based on

agreed upon methods for cleaning data. Baudel's approach of direct manipulation for cleaning data, described above, is an interesting exception [10].

Creating L2 derived data products can be a complicated process that entails several challenges. Workflow systems [9, 11, 22, 56, 103] provide a significant step forward in this regard, striving for several goals simultaneously. First, they attempt to raise the level of abstraction for scientist-programmers, allowing them to reason about their computational tasks visually as data flow graphs instead of syntactically as scripts. Second, workflow systems aim to provide reproducible research. Computational tasks are often more difficult to reproduce than laboratory protocols due to diversity of languages, platforms, user skills, and usage scenarios. Expressed as a workflow, these protocols are easier to share, reuse, and compose than are raw scripts. Third, workflow systems help to abstract away the execution environment, allowing workflow tasks to be executed on a variety of different platforms. For example, the Kepler and Trident systems allow workflows to be submitted to a cluster for execution or evaluated directly in a desktop environment [9, 56]. However, they do not provide for interactive visualization on the client, as workflows are typically executed as batch jobs.

Finding ways to re-use workflow to benefit others, especially non-programmers, is essential to making workflows a more useful scientific tool. To this end, Maechling *et al.* [59] showed that visual interfaces and example workflows make a large difference in the use of workflows by scientists. Silva's VisTrails [11] project showed that capturing workflow creation steps lets users store a seamless visualization creation history and leverage this information to create and optimize multi-view visualizations. A subsequent paper by Scheidegger *et al.* [84] demonstrated how data could be mined to allow querying and creation of visualization by analogy, using a distance metric to determine comparable workflows.

Gobel focused on making workflows easier to share among scientists. In her words, "Workflow is more than just plumbing – workflows are valuable know-how about how to create important derived datasets. They are also protocols that define research results. They are just as important as data." She contends that scientists should be able to describe and

share workflows and other history as easily as people share documents, photos, and video on the Web. Gobel and De Roure documented initial success toward this goal with their myExperiment project, which provided social networking for e-scientists via workflows [30].

2.3 Data Exploration and Visualization Systems

The generation task in Springmeyer's categorization, which involves creating a visual representation of data using systems and tools, is now described by presenting several different data exploration and visualization systems available to oceanographers.

2.3.1 Programming Systems

The most basic systems require users to programmatically create visualizations. Many of these started as research projects and evolved into commercial or open source products for the scientific community. All are now mature and provide an extensive range of visualization techniques for users with the ability to create the necessary programs.

Visual systems were one of the earliest programming solutions, as illustrated in the Application Visualization System (AVS) [107] by Upson *et al.* in 1987. This research defines the basic model still followed by most other visualization systems. Software modules are defined to carry out computation on a set of primitive data types, such as a 3D grid. These objects are then connected into dataflow networks that take in datasets and return visualizations. Changes to the visualization are made by modifying object parameters in the dataflow network. Editing visualizations is supported with a direct manipulation interface to connect objects. IRIS Explorer [26] expanded on the AVS metaphor by adding UI widgets to each object, to easily expose and combine control interfaces to modify visualizations. Abram and Treinish [1] created IBM Open Data Explorer (now OpenDX), which rejected some of the simplifications of these drag-and-drop programming models as inadequate for real-world problems and added more programming structure. In particular, they included conditional execution, iteration, parallelism, and state preservation. Researchers have also shown that successful systems can be built by adhering strongly to comparison principles. Bavoil *et al.*

[11] presented VisTrails, a system that used information from user-entered workflows to create multiple synchronized data views and optimize the underlying system to improve interactivity.

Hibbard pointed out that while these systems offer flexibility, the visual programming approach remains too hard for novices and too limited for experienced programmers [40]. To solve the latter problem, toolkits provide application libraries for developers to build solutions in a compiled environment instead of a drag-and-drop interface. Two such systems are VTK [86] and VisAD [41]. VTK was built on C++ to provide simplicity, portability, and speed. This system also included interpreted language interfaces for simple interactive creation of visualizations. Hibbard *et al.* created VisAD – an extensible, object based toolkit in Java – with a simple interaction model as part of the design. Both have added example viewers to display the capabilities of their respective systems.

2.3.2 Desktop Tools

An easier visualization approach is supplied by desktop data explorers, shown in Figure 2.4. There are several options for basic 2D and 3D plotting. For example, ncBrowse is a system for *NetCDF* files – a self-describing, grid-oriented binary format common in earth science [46]. It supports the selection of parameters and some simple options in the plot layout and decorations. Ferret is a similar plotting and analysis package designed for physical scientists studying global ocean/climate interactions [37]. It provides a semi-automated and flexible environment in which the user can probe large and complex gridded datasets, such as model outputs and observational datasets. Another widely used tool in this category is Matlab, which provides a wide range of data manipulation capabilities and plotting options as well as a variety of libraries to extend its analysis capabilities [61]. Ocean Data View, an oceanography-focused tool, loads station data directly and supports georeferenced 2D plots of data along with other georeferenced objects, such as maps [72].

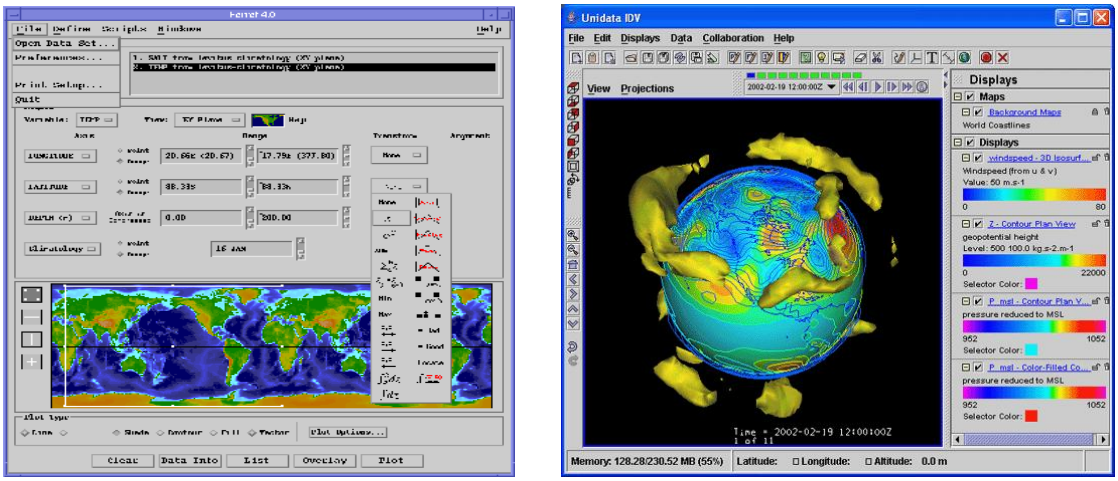

Figure 2.4: Examples of desktop tools. On the left is Ferret which plots data. On the right is the Interactive Data View (IDV), which provides desktop visualization tools for 3D NetCDF datasets.

Two more powerful desktop tools are the Interactive Data Viewer (IDV) [67] and GeoZUI3D [112]. IDV was built for the atmospheric community on top of VisAD. With a strong underlying visualization system, it supports sophisticated 3D model visualization (iso-surfaces, contours, volumetric), terrain maps, and data animation. GeoZUI3D (and its commercial version, Fleidermaus) was designed to integrate multiple 3D bathymetry sets and arbitrary geolocated 2D and 3D objects for oceanographic visualization. It recently added the ability to play back time-varying 3D point data [7]. With the increasing power of these tools, the complexity of the interface has also increased, making it difficult for novices to quickly and easily create data representations. Further, these desktop tools do not provide a unifying model for diverse geolocated data.

2.3.3 Geographic Visualization Systems and Geobrowsers

Geographic Information Systems (GIS), shown in Figure 2.5, provide an integrated model for geolocated data. They consist of a database management system, a set of operations for exploring data, and a graphic display system, all tied to the geospatial analysis of data [80]. ArcGIS, an example of this type of system, was designed for professional map making. For much of the '80s and '90s, GIS and scientific visualization systems developed in parallel with

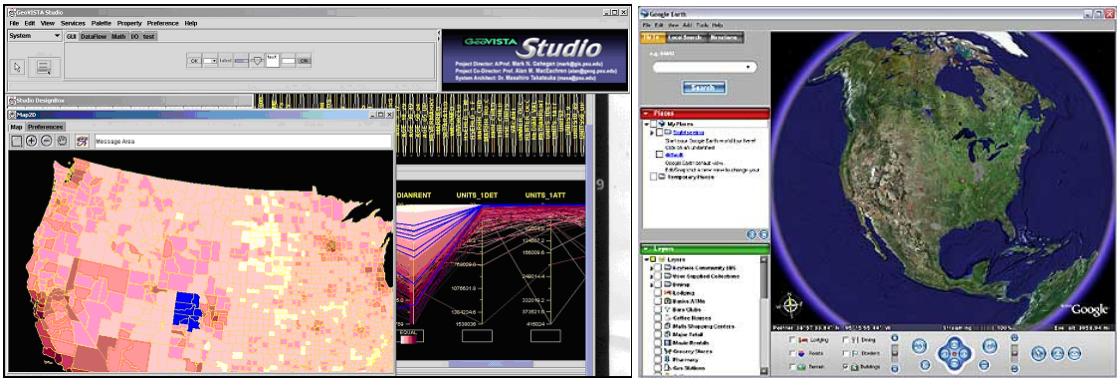

Figure 2.5: Examples of geographic visualization systems and geobrowsers. On the left is the interface for the GeoVista Studio Toolkit. On the right is Google Earth.

little cross-over, but convergence over the last several years has enabled research into systems that support both types of data. GeoVista Studio [100] described an environment for geospatial scientific data, focusing on visual programming for quick application creation and Java for extensibility and deployment.

Georeferencing browsers (*Geobrowsers*), such as Google Earth, shown in Figure 2.5, provide a more accessible interface to geographic data. The first widely available geobrowser was the TerraServer created by Barclay and Gray [8]. It provided a Web-based interface that allowed users to easily zoom in for more detailed satellite images of the earth. Google Earth (originally Keyhole) built on this model by creating a local client providing a 3D earth navigational paradigm [31]. This client supported the ability to easily layer more geolocated images, maps, labels, lines, and objects over the earth images and deliver relatively easy sharing of layers through XML documents. An example of the capabilities of geobrowsers for oceanographic data can be found in the Google Oceans project [32], which also shows the limitations of current systems: ocean terrain is limited to the low resolution data in the Google servers or to 2D images overlaid on the ocean bottom, there are no scientific data display or analysis tools, and there are limited interactive layout capabilities. World Wind from NASA and OssimPlanet are both open source geobrowsers that were modified by researchers to display examples of specific earth datasets [20].

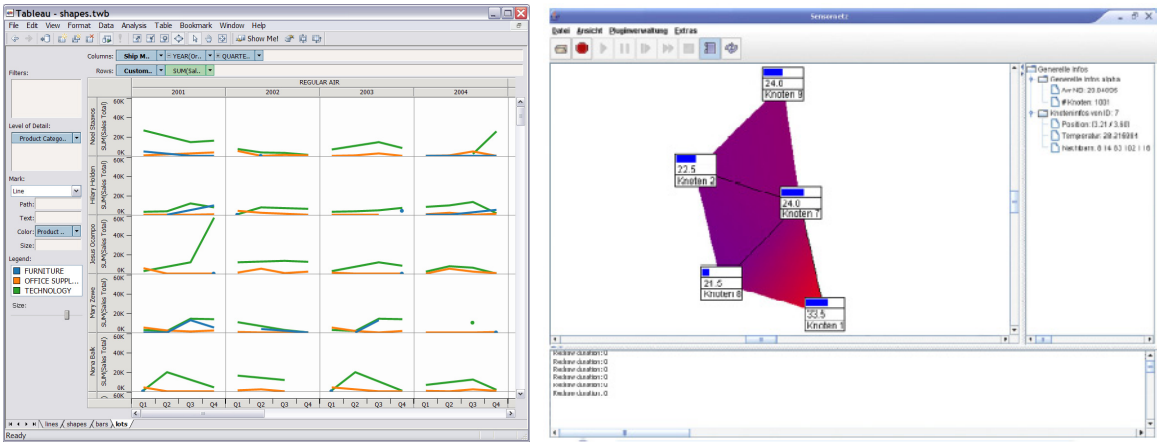

Figure 2.6: Examples of other data exploration systems. On the left is Tableau's information visualization interface. On the right is SpyGlass.

Geobrowsers can also be used to display virtual observatory data, as demonstrated by the EarthScope project [70]. In this system, seismometer locations are displayed across a map, and sensor detail is available by clicking to bring up details of a geolocated object. Ease of use and broad availability of the viewing application are a significant advantage, while limited interface capabilities are the major drawback.

2.3.4 Other Data Exploration Systems

Information Visualization (*InfoVis*) systems, shown in Figure 2.6, support many useful features for viewing observatory data. In particular, while being weak on geospatial and scientific visualization, they are often strong on data comparison and query. Two such systems are Polaris by Stolte *et al.* [95] and Improvise by Weaver [113]. Both systems were designed to support the unpredictable nature of data exploration by allowing rapid changes by the user to quickly generate new visualizations. Polaris used an extension of SQL to generate visual specifications and data transformations directly from the user's interaction with the system. Improvise defined a system for coordinated queries across several visualizations to generate closely synchronized views of data. This let users build highly coordinated visualizations across several windows.

Another type of system is an observatory viewer specifically designed for viewing sensor networks. Because observatories have appeared relatively recently, most of these systems are ad hoc and built from existing tools. Recently, systems have been created to allow observatory-specific interfaces. Spyglass supports viewing of instruments, their connections, and detail about specific nodes [17]. It also allows exploration features, such as interpolation of realtime data values, by extending the base system with new modules. The current system provides no inherent geospatial context for the nodes, but including maps in the view is feasible by creating new modules. The Shu *et. al.* system provides abstract visualization of connectivity between the wireless sensors in a sensor network in order to optimize data flow strategies [90].

2.4 Gap Analysis

I will first consider these classes of systems with respect to their ability to carry out the scientific data exploration tasks described in Section 2.2. Table 2.2 below summarizes the tasks supported by each class of visualization system in order to provide a comparison. Each of the five tasks has several dimensions that were discussed previously:

Examination: Plots (**P**), Geospatial (**G**), Models (**M**), Vectors (**V**), Observatory Layout (**O**)

Orientation: Navigation (**N**), Aids for Navigation (**A**)

Comparison: Layering Data (**L**), Multiple Views (**M**)

Queries: Data Details (**D**), Data Filtering (**F**), Visual Query (**V**)

History: Implicitly Captured (**I**), Explicitly Recorded (**E**), Scientific Workflow (**W**)

The boldfaced letters associated with each dimension listed above will be used in Table 2.2 to provide a finer level of comparison. A capital letter indicates strong support, a lower-case letter indicates adequate support, and an underscore indicates little or no support. For example, "`P g M V _`" as an entry for Visual Programming systems under the Examination task column means that type of system is excellent for Plotting, Ocean Models and Vectors, can be used for Geospatial Data, but has minimal support for Observatory

Layouts. An ideal system will have entries with capital letters across the entire row, while a weak system will show primarily underscores as entries, indicating gaps in the capabilities.

Table 2.2: Data exploration systems evaluated by task area. A capital letter indicates strong support for a capability; a lowercase letter, adequate support, and an underscore, little or no support. (* indicates VisTrails is the only visual programming system with workflow support.)

System	Examination	Orientation	Comparison	Query	History
Visual Programming	P g M V _	N _	L M	D F v	I h W*
Toolkit	P g M V o	N a	L M	D F v	_ _ _
Plotting Tools	P g _ _ _	n _	l _	D f _	i e _
3D Data Tools	P G M v _	N _	L M	D f _	i e _
GIS	P G m _ _	N _	l m	D f _	_ e _
Geobrowsers	_ G _ _ o	N A	L _	d _ _	_ e _
InfoVis	P _ _ _ _	n A	_ M	D F V	I e _
Observatory Viewer	P _ _ _ O	n _	_ _	D f _	_ _ _

Based on the analysis in Table 2.2, one can quickly see that no ideal system exists for ocean observatories. Most systems successfully allow users to examine a representation of either the data or the observatory state, but rarely both. The programming systems and desktop tools provide the most capabilities in this regard, and 3D Data Tools provide a good example of the available capabilities for scientific exploration and visualization. The other systems have more limitations: GIS systems and Geobrowsers have limited data display capabilities, InfoVis systems have limited geospatial support, and Spyglass displays only the sensor network. None of these systems directly supports both observatory and data visualization, but this could reasonably be explored with visualization toolkits.

Orientation in the form of navigation is also reasonably supported in these systems. The 3D systems allow camera control, and the geospatial systems have geolocated objects for visual context. Geobrowsers deliver the most seamless navigation by focusing strongly on

the user interface. Some constrained orientation is also available with Geobrowsers in the form of saved tours and the ability to jump to specific sites.

Comparison is usually possible through layered representation in the systems. Outside the user programmable systems, the systems surveyed afford little inherent support for rich window synchronization. Quite often, one can only drag two windows next to each other to determine differences. The InfoVis systems feature models for either automatically displaying comparison windows or allowing strong synchronization between windows. Beyond providing simple detail on a data point, querying is another area where InfoVis systems are strongest. Their integration with the underlying query language transparently allows sophisticated data queries.

Finally, the history task has little support beyond simple undo/redo or possibly simple annotations. Exploration of interaction histories and creation of annotation trails to document work is not supported. The visual programming systems generate workflows as a necessary artifact of creating visualizations, but re-using and sharing them are limited to reloading saved files. The exception is the VisTrails system, specifically designed to capture visualization workflows from the user.

While Table 2.2 illustrates how data exploration tasks are supported by existing systems, there is more to evaluating useful exploration tools than ensuring capabilities exist. Determining their overall effectiveness for observatory needs requires balancing usability with functionality. In this regard, some researchers have created more quantitative measures to evaluate visualization systems. The approach by Amar and Stasko [5] is based on determining a system's ability to support decision making. To this end, they defined a set of questions to investigate where analytical gaps exist in the system. Similarly, after examining the slow acceptance of new data visualization techniques in the sciences, Van Wijk [110] proposed the following equations to assign costs and the returned value for a given technique:

$$C = C_i + nC_u + nmC_s + nmkC_e, \qquad G = nmW(\Delta K), \qquad F = G - C$$

The first equation describes the time cost, *C,* for data visualization when used by *n* users *m* times, with each carrying out *k* steps to explore it. The total cost is the summation of the initial cost to create the visualization C_i, the cost to learn the software C_u, the cost of each session C_s, and the cost to explore and understand the representation C_e. The return is depicted in the second equation, where *G* is the value of the acquired knowledge *W (ΔK)* times the number of times it is used. The total profit is the difference, $F = G - C$. Intuitively, better visualizations have lower costs at each user step, can be understood more quickly, and have a high return on knowledge for several users over several sessions.

This approach is limited by its difficulty in calculating accurate values, assumption of constant cost, and return across users. However, it provides a model for evaluating the relative merits of visualization systems with respect to the effort required for them to be effective. Van Wijk's main contention is that systems must examine all variables in this equation and not just focus on providing novel tools and techniques. In particular, he sees the per user cost, C_u, as the major impediment in most systems. Scientists' time is valuable, and a clear profit from the visualization tool must be obvious for it to be used [110].

Table 2.3: Data exploration system evaluation based on Van Wjik's equation.

System	**Creation** C_i	**Learning** C_u	**Exploring** C_s, C_e	**Return** G
Visual Programming	●	●	●	●
Toolkit	●	●	●	●
Plotting Tools	•	•	•	•
3D Data Tools	●	●	●	●
GIS	●	●	●	•
Geobrowsers	•	•	•	•
InfoVis	●	•	•	•
Observatory Viewer	•	•	•	•

Table 2.3 applies this methodology to the systems reviewed previously. The second, third and fourth columns represent costs; creation is C_i, learning is C_u, and exploration includes C_s and C_e. The right-most column represents the knowledge returned on observatory datasets and observatory instruments and is based on the analysis in Table 2.2. The magnitudes of costs and return are represented by four different sized indicators. The most successful solution would have small indicators in the three cost columns on the left and a large indicator in the return column, creating the highest profit.

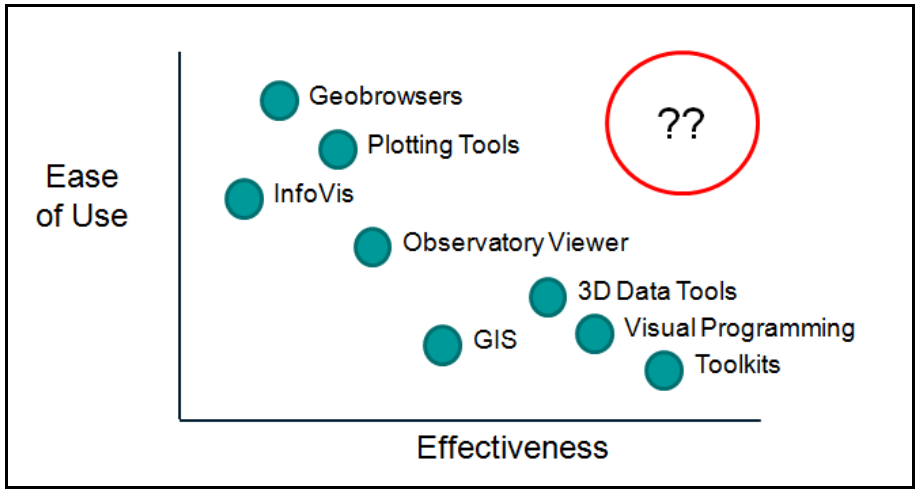

Figure 2.7: Plot of system effectiveness for observatory needs against ease of use.

By plotting system effectiveness for observatory needs against ease of use, one can quickly determine where existing options lie in this space (Figure 2.7). In general, one can see that the tools are high cost/high return or low cost/low return. Visual Programming, Toolkits, and (to a lesser extent) 3D Data Tools provide the greatest possible return but at a significantly higher user cost. Plotting Tools and Observatory Viewers have much lower costs but also a much lower return outside of their area of specialization. Similarly, InfoVis systems have low costs after the database is set up, but they are not well suited to displaying geospatial or observatory data. GIS and Geobrowsers provide a similar amount of value, but Geobrowsers have a much lower user cost. An ideal system in this evaluation would provide the low user cost of tools (such as Geobrowsers) with the high return of Visual Programming, Toolkits, or 3D Data Tools to derive the largest user profit.

2.5 Conclusion

> *"Data analysis tools have not kept pace with our ability to capture and store data. Many scientists envy the pen-and-paper days when all their data used to fit in a notebook and analysis was done with a slide-rule. Things were simpler then; one could focus on the science rather than needing to be an information technology professional with expertise in arcane computer data analysis tools."* Jim Gray [33]

There are many options available for viewing ocean observatory data, but as illustrated in the preceding analysis and commented on by Gray, there are gaps between the needs of the scientists and the solutions available. The primary issue illustrated is that none of the systems investigated provides an easy to use, integrated data exploration and visualization solution across all tasks. Visual Programming systems and Toolkits provide a broad range of capabilities, but they force scientists to become programmers either implicitly or explicitly. 3D Desktop Tools provide nearly as many capabilities, but they have interfaces that are also difficult to learn and use. Geobrowsers have a simple, easily understood model for georeferenced data and navigation, but do not support custom 3D bathymetry, exploration of 3D ocean datasets, or data analysis tools. Other systems investigated provide effective solutions to only a subset of observatory needs.

Oceanographers must therefore use multiple systems to store, analyze, and visualize their data. This leads to inefficiencies in exploration, makes sophisticated analysis pipelines seldom re-useable, and leaves little opportunity for sharing processes or data. This model will be exacerbated as ocean observatories come online with exponentially larger amounts of data. A system that integrates support for all the preceding areas would enhance exploration and sharing by avoiding a plethora of applications and interfaces. This point is reinforced by Springmeyer's interviews with scientists [92] as well as by Bill Hibbard, who noted the need for an earth science system that is comprehensive in tasks, possibly at the expense of techniques [42].

Furthermore, while there are many interesting interactive techniques for exploring 3D ocean models, they are not present in the tools available. Simple slicing planes, contour lines, or transparency are the most common techniques available to scientists today. The 2002 article by Bill Hibbard *et al.* on the state of model exploration [42] discussed the growing need for sophisticated software and the lack of systems to adequately meet those needs. He sees the tool problem as not only the lack of available techniques for model visualization but also poor integration of the techniques into a highly usable environment. In particular, he points out that systems need to provide more effective ways for direct manipulation when exploring this type of data.

Finally, an inherent limitation is the lack of tools for exploration and interaction with the observatory itself. This limitation is not surprising because ocean observatories are in their infancy. With a growing need for the future creation of these scientific systems, such tools are crucial to an observatory's success. A related issue is the ability to monitor and control the current state of the observatory. Given system complexity, recognizing and responding to instrument problems can be greatly enhanced with an interface for viewing data and ocean environments in the context of sensors. For example, one can envision an interactive solution for re-tasking observatory assets for special, one-of-a-kind events, such as earthquakes or volcanoes, in response to data simultaneously collected from the site.

CHAPTER 3

CONTEXTUAL DESIGN STUDY

While a literature review is valuable for revealing gaps in existing systems, it does not provide a clear description of the design requirements for an effective ocean observatory system. To determine these requirements in science settings, I performed field studies to discover which data visualization and exploration techniques are actually being used in the ocean sciences and identify data visualizations commonly in use.

I conducted a ten-week contextual design study with ocean scientists at the University of Washington School of Oceanography and the Monterey Bay Aquarium Research Institute. I met with the scientists on multiple occasions to observe their interactions with other scientists and to gather sample datasets, visualizations, and data analysis practices and tools. I further interviewed nine members of the teams in depth to collect detailed feedback. This chapter describes the methodology used, the key themes identified, and the user interface design guidelines developed from my work with oceanographers.

3.1 Methodology

Several human-computer interaction projects have shown that user-centered design methodologies are important for successful system design involving the science community. Work on Collaboratories [73] and Labscape [6] determined that integration with the daily work patterns of research teams was crucial for accurately assessing needs. An investigation by Schraefel to understand chemists' lab books noted that a mixed approach was needed to complete her team's design work, including both user-centered interviews and ethnographic observational studies [85]. The CINCH system [3], a science interface for 3D neural pathway selection, was developed by spending months with neurologists capturing

automated event logs, carrying out interviews, and involving the scientists in participatory design techniques.

The work of Star and Griesmer [93], and further refinement by Lee [51] on boundary objects and boundary negotiating artifacts, provided a valuable basis for understanding how teams employ artifacts to collaboratively communicate and plan. I also benefited from recent ethnographic research into cyberinfrastructure by Finholt and Ribes, who explored the role of community [81] in large research environments, and work investigating the human infrastructure in a large cyberinfrastructure project by Lee *et al*. [52].

Informed by this research, I carried out a ten-week contextual inquiry study with ocean scientists. Contextual inquiry is based on *Grounded Theory* – a methodology developed in the 1960s by Glaser and Straus [43] to provide an alternative to the dominant forms of sociology research at the time. In particular, it avoids attempting to categorize studied processes ahead of time and instead focuses on collecting and analyzing data to generate a conceptual framework; field notes and artifacts from a site are collected and interviews are conducted with participants in the system being studied. These datasets are then coded to develop theoretical categories that explain the observations, and one or two page memos are written to explore the code categories in more depth. Through iterations over the data that inform further investigations of the system, the conceptual framework is refined to identify key themes observed during the study. These themes may then be used as the basis of user interface guidelines in systems designed for the environment.

I employed this approach at two different ocean science institutions: the Monterey Bay Aquarium Research Institute (MBARI) [63] and the University of Washington School of Oceanography [106]. MBARI is the largest privately funded oceanographic organization in the world and acquires data through fixed and mobile instruments, ship-based cruises, and occasional large-scale, multi-institute projects. At the University of Washington I worked with two groups: one group is building the Regional Scale Nodes (RSN) portion of the NSF-funded Ocean Observatories Initiative, and the other is generating regional-scale simulations of Puget Sound.

During these investigations I visited the research sites two to three times a week over the ten-week period of the study. I first conducted field observations in the participants' offices and during meetings. In most situations, data exploration and visualization were discussed directly or utilized to support the discussion. Each session took one to two hours, during which I recorded notes on user activities. This observation period was followed by nine in-depth interviews with participants: 6 ocean scientists, 2 staff members who generated daily data and visualization products for oceanographers, and 1 research scientist who led a graphics staff that created high-end earth science visualizations. Their professional experience with oceanographic visualization ranged from 5 to 30 years. Each of the interviews was then transcribed and coded along with the initial site observations. As data exploration was observed and discussed in these sessions, visual artifacts were collected in digital or paper form for use with the notes and transcriptions.

3.2 Key Themes

I observed several key themes during the investigation: some of which focused on the data that scientists worked with, some with the tools they used, and some with the processes they employed. All are described below.

3.2.1 Geographic context for ocean data is often essential.

A theme I consistently observed was the importance of geographic context in exploring data. This was often based on the diversity of data being explored. As stated by one subject, "*I would guess we are one of the most complex sciences, with everything from the genome, to plant samples, to water chemistry, to terrain. All are important to tell the story.*" Another important factor was providing a framework for colleagues. Often a scientist would not need to include maps while exploring the data personally, but when displaying it to other scientists, the location and time of data collection was almost always present in at least one of the visualizations.

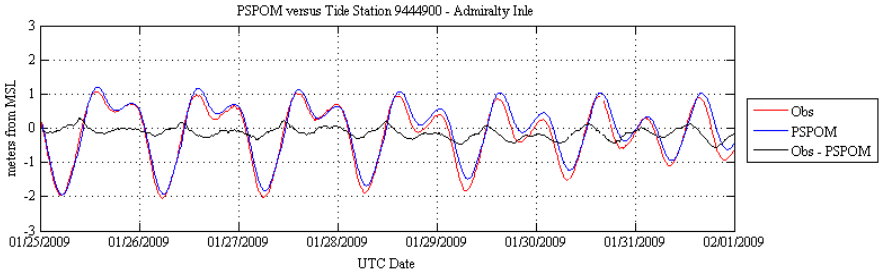

Figure 3.1: A 2D plot with observed and simulated tides and a comparison of the two datasets.

Further supporting the use of geographic context is the growing availability of datasets helpful in depicting a particular location on the ocean floor in more detail. High precision sonar has made it possible to collect detailed terrain maps of the bottom of the ocean, with centimeter-level coverage in some areas. With the recent availability of digital cameras for submersibles, these platforms can shoot video continuously and automatically snap high resolution images every few seconds while on dives. The images can later be stitched into larger continuous images of the seafloor to provide a more comprehensive environment for data representation and experiment planning.

3.2.2 2D tools dominate exploration and visualization.

With respect to artifact formats, 2D plots and graphs are most commonly used by the scientists, as displayed in Figure 3.1. Often, this is because they are looking at only one variable against time or distance. A second primary driver is the typical way results are disseminated, as one participant noted: "*As long as printed journals are the main medium for scientific papers, 2D static images will probably be the main format for scientists.*" Another participant explained that while movies were getting easier to use and forward to colleagues, at a conference he often cannot choose what system will be used for presenting, so he cannot depend on movies for visualizing data. It is safer to design to the lowest common denominator and ensure they can reach their audience.

The most common tools for these scientists were MATLAB, and, to a lesser extent, Ferret, data plotting tools described in Chapter 2. They commented that these tools made it

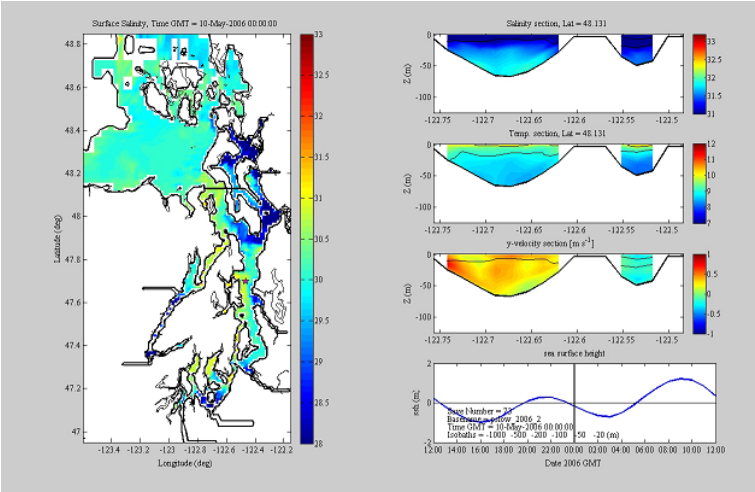

Figure 3.2: Example of 2D plots and 2D geographic images that are combined to create a richer visualization.

relatively easy to manipulate data and get simple plots to quickly visualize a scenario, particularly when exploring data. For making presentation quality visuals, however, most found these tools lacking. Instead, they would use the tools to create the plots and use other tools to finesse adornments, such as typeface and labels. This model was common across the participants. Participants also noted that working with simulated data was often problematic. MATLAB, in particular, was not good at keeping track of geolocation metadata, loaded all the data into memory at the same time, and was hard to operate in an automated fashion.

3.2.3 Data-rich artifacts are not uncommon.

Despite the limitations with reliable output formats and tools, I found that oceanographers were creating rich visual artifacts. They spoke of the complexity of their problems and the need for strong visuals to communicate within and outside their teams. These rich artifacts often consisted of several visuals created with 2D tools that were viewed together to allow quick comparisons of variables at one time point. Another common operation, at least within the modeling community, was to visualize expected output next to actual output. Including methods to look at data in a geographic context was quite common but was usually limited to a top-down, flat map view (see Figure 3.2).

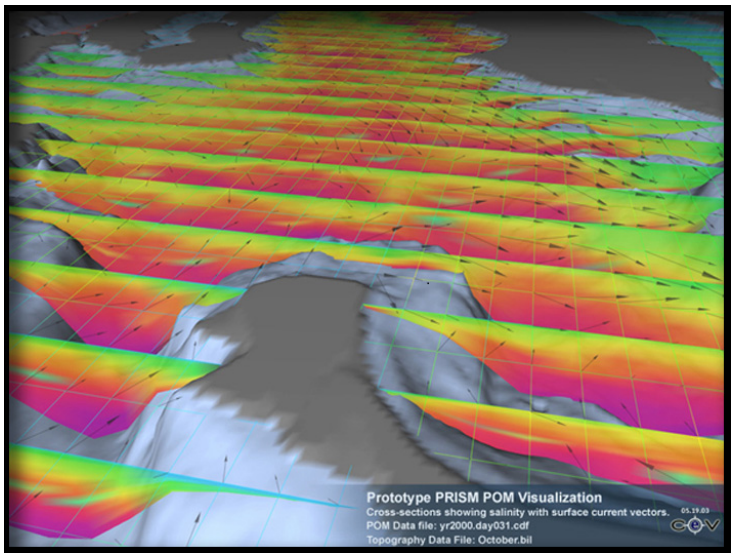

Figure 3.3: A rich 3D visual created to display salinity and currents in Puget Sound.

For viewing data over time, different images for a select set of time steps were often used. For example, a set of 5 to 10 images of ocean surface temperature off the coast of Washington from different times was displayed to highlight changes. A similar solution was employed for model data, where a set of cross-sections was used to visualize the data against the seafloor at several vertical slices in the model. Integration of movies with charts and plots was another solution to visually detect areas of change. In some cases, scientists positioned movies adjacent to each other and viewed them concurrently to see several variables changing simultaneously or a single variable changing in different ways.

There was also often a strong focus on the details of presentation visuals. Participants would make their own color gradients and worry about specific visual issues, asking such questions as: "*Does this image require a flat color or should it be shaded?*" "*Are contours necessary?*" "*How thin should I make this line?*".

3.2.4 3D and 4D visualizations are useful but creating them is too difficult.

Many participants said that they would like to use more 3D geolocated visuals, such as the artifact shown in Figure 3.3, but that it was exceedingly difficult to do using their current

tools. This was exemplified by a 3D movie a scientist scripted in MATLAB that showed vortices due to the tides in Puget Sound. He was pleased with the movie, and he said it was not only good for grabbing people's attention, but that it showed things that would be almost impossible to see in a cross section. He then pointed out that he had made the movie several years earlier but had not made a similar one since: "*It was just too much effort to make in MATLAB and make it look good, especially getting things like lighting, camera angles, and textures right.*"

Google Earth as a 3D platform for science or outreach does not solve these issues or currently appear to be of great interest, a position summed up by a participant's comment: "S*o I think it's a fairly powerful tool, but it's limited. I think it does some things very well, such as images. So if my science data can be put into an image that sits on the surface of the earth, then it works well.*" Scientists found it tedious or infeasible to convert their ocean data into 2D images for Google Earth and even if they did, they would then lose the ability to interact with it further. Regarding the recently addition of bathymetry, another pointed out, "*There isn't that much to show in the interaction between all the detail that Google Earth provides on land and my data in the ocean. And as for the bathymetry, any place that I really care about, I have my own bathymetry for it.*"

3.2.5 Simulated data has a distinct set of problems.

Many of the participants I observed and interviewed either create or work with ocean models. As the scientists described the process, creating a model entails exploration to determine the right data inputs to drive the model, and the setting of internal parameters and model resolution. They then must validate or determine the correctness of the generated model, which has several phases. "*There's a level of just figuring out what a model run did. You get half a terabyte of data, and you just want to know what happened. Then, you want to be able to compare it to your last model run to see if the new run is better, and even more importantly make sure it's accurate, which means a lot of direct comparison between the model and observations.*" This tedious process required many different tools and processes,

and what they desired instead was a system where they could "*make a change and then push a button to spit out a set of plots and images*" to let them quickly validate any model run.

I saw that comparing models to observed data and to other models was difficult because many were set up on non-regular grids. This means determining the grid cell that a data point is in might require extensive manipulation. One participant described frustration at trying to compare two models: "*Then I see that they're on different grids, which means I can't do a direct comparison. So now I'm going to have to pull them into MATLAB or some other tool, figure out how to re-grid them, and then finally do the comparison…which means I probably don't do it.*"

While the sample grid of model data is one issue, another is file size. For many participants, data size is currently an inconvenience, but it will soon become a limiting factor. One participant pointed out that current simulation files are 130 MB for one time slice, and that the final model of Puget Sound will be 10 times bigger, making a dataset at one hour intervals over 300 gigabytes a day. The ultimate goal of their modeling project is to make datasets that cover years at much higher resolution. The scientist hopes to do most calculations on a server to allow just images and movies rather than all data to be downloaded to his system but is unclear about how to accomplish this with current tools.

3.2.6 Current tools do not extend well to the cabled ocean observatory infrastructure.

As explained in Chapter 1, a cabled ocean observatory consists of hundreds of kilometers of cable and hundreds of instruments. The general structure of the layout must first be determined, followed by specific components that will flesh out the design. This was observed to be an iterative process using a variety of tools and techniques. One participant described the process this way: "*It's done a little bit piecemeal. I've got a GIS tool, and I can use it for this. I'm searching on the Internet for other things. A certain amount of it, too, is done on whiteboards and papers and the back of napkins.*" After the broad brush strokes of the design are agreed upon, more levels of detail are added "*to get all the details to flush it out or to support it.*" Internal feedback on observatory design is usually ad hoc, with a

meeting once a week to review key items, synchronize plans, and ensure all constituencies can accomplish their tasks based on the current design.

There were several issues with the current approach noted by the participants. One problem lies with the limitations in exploring and displaying the design in various ways. "*Our current process is pretty limited in what it can represent, so we use a [straight overhead] view almost all the time. Another thing is that you can't change scale quickly. So, if I have a big overview of all the cable and I want to zoom in, well, that's usually a whole different image that has to be generated separately.*" Another problem is the time involved in creating visuals to communicate the design. "*The time depends on the type of image. You can have the back of the napkin right away. You may have a better geo-referenced one within a day. And if you want a highly polished one with other datasets to show outside the group, well that depends on the graphics group.*" The participants also noted the length of time required to modify visuals when the design changed. "*If it's just moving points around, you can do that in hours. The problem is that you have to do it again and again and again because the design keeps changing.*"

3.2.7 There is a growing continuum in the visualization process.

In discussions of data visuals created by the participants, a common view of the visualization continuum emerged. Up front the process was quite fluid and exploratory as hypotheses were generated and refined. As one researcher said to me "*You didn't think it would be this crude, did you?*" There were lots of throw-away visuals in this phase, but they were also looking for visualizations they could use for daily monitoring of the experiment they would be running.

As projects moved beyond exploring many hypotheses to exploring a few in detail, the visualizations tended to become more standardized and moved to an operational phase where they were run daily on new data. One goal was to be more consistent and remove variability, because, as one scientist noted, "*If visualizations are changed it may be necessary to redo several of the past runs in order to be thorough.*" They also talked about the trade-off they faced here: they ended up looking at everything the same way all the time and sometimes felt

like they were "*settling with their current images for analysis*" as opposed to further exploring the data. Besides the desire to minimize changes to the environment, this was also because the images were so hard to update. "*I have to go back to the scripts, remember how they work, and then figure out what it is I need to change.*"

After the experiment was complete (or some phase was complete) the visuals would be used for papers, presentations, and public consumption. Sometimes the text in a paper would drive the visualizations, but often it was the other way around. As one researcher explained, "*I usually start with the figures. I tell the story with figures by just laying them all out and then writing the text around it.*" He said that he thought this was not uncommon in their field. Finally, concerning visualizations provided for public consumption, the researcher noted: "*If we put something on the Web it's almost always just a co-opted version of one of our other visualizations.*"

3.2.8 Framing the story is important.

Another important theme for the scientists was using visualization to help frame "the story" for the audience. The needs here varied based on the audience and the intended outcome. Sometimes, the impetus was to enhance what was communicated to the audience, "*So I can tell a more complicated story if I have the time to do this level of post-processing and intensive visualization.*" Other times, it was to communicate most efficiently. "*I try to just make it really simple. I want them to be able to glance at the figure and in five seconds see what I'm trying to show them, and then they can move on with their lives.*"

For communicating with other scientists, care was often taken to present the work rigorously. "*When it comes to talking to a specialized scientific audience, I want a lot of control over the story, so I can highlight exactly what we did.*" Visualizations were also considered preferable to sharing data from an experiment. This was sometimes viewed as a matter of controlling the message. "*With a graphic you have more control over the story, and also it's harder to repurpose. With the data itself you know they can go write their own paper, but with the figure, they can only show the figure.*" Others saw it as a boundary issue with respect to collaboration. "*If in a talk, for example, you just take a figure from someone,*

put it in your talk, and credit them at the bottom, that's no big deal. To get someone's data, do your own thing to it, make your own visualization of it, and show that in a talk, that's more of an intimate boundary crossing thing."

3.3 Design Guidelines

Based on these eight key themes, I derived the following set of interface design guidelines to address noted deficiencies in current tools and techniques.

3.3.1 Make high-quality geolocated 3D data visualization easier.

The first, third and fourth themes explore characteristics of data exploration and visualization artifacts generated by scientists, and show that creating high quality 3D visualizations with geolocated data is currently difficult and conducted with an ad hoc collection of tools and techniques. There is a need for a data exploration system that makes geolocation a central focus to provide context and that allows scientists to easily integrate other datasets to quickly build rich visuals.

3.3.2 Integrate with existing task-specific tools, when feasible.

The second theme, concerning existing systems, shows that while there were complaints about specific plotting packages and other dedicated tools, most scientists found systems that effectively met their needs for specific exploration tasks. Where they struggled was in finding ways to use these tools together to create rich visuals and 3D geolocated representations. Re-inventing new tool interfaces for all exploration tasks is not a leveraged approach. A better solution is to integrate with existing interfaces and methods in convenient ways.

3.3.3 Add support for ocean modeling processes.

Simulation will continue to grow in importance as processing speeds increase and ocean processes being studied become more complex. The fifth theme indicates that that there is an

unmet need for simple and powerful tools that allow modelers and consumers of models to explore simulated datasets and compare them to observed data and other models.

3.3.4 Provide a flexible and scalable data architecture.

Another element from the fifth theme, and from the previous discussions of ocean observatories, is the continued growth of observed and simulated datasets in size and diversity. To be useful going forward, an exploration system must accommodate this dynamic. For flexibility, providing a toolset for handling new formats is as important as transparently managing current formats. For scalability, it is as important to run across server farms and in the cloud to deal with extreme data size, as it is to run on the local machine to maximize speed.

3.3.5 Make instrument layout interactive and visual.

The sixth theme demonstrates that complex instrument layout is currently performed with non-visual tool formats, such as spreadsheets and text documents, with visualization as a last step to double check the work created. Visual interactive tools enable easy creation of layouts for data and bathymetry and provide immediate feedback on key metrics, such as cost and bandwidth. Such tools facilitate rapid iteration to optimize layouts and communication across diverse science teams.

3.3.6 Provide for a wide range of visualization audiences.

From the seventh theme, we see that data visualization is necessary for a growing number of research needs and audiences. However to accomplish this, data currently needs to travel through many different applications and formats to effectively reach those audiences. To alleviate the friction of moving between different environments, a single system needs to be usable in more of these communication contexts.

3.3.7 Focus on sharing visualizations instead of data.

The eighth and final theme illustrates the importance to scientists of presenting representations with data that help frame the story. I also observed that most scientists are reluctant to share raw data without context. In addition to these user needs, moving data around is fast becoming impractical due to size constraints. Creating systems where interactive visualizations can be shared, rather than the data itself is a solution to this problem.

3.4 Conclusion

This chapter presented insights gained from spending time with ocean scientists to study their data exploration environment. Based on this effort, I developed a better understanding of current data exploration and visualization techniques, which I used to outline a set of user interface design guidelines for systems to address the scientists' unmet needs. The next chapter describes COVE, an interactive exploration and visualization system based on these guidelines.

CHAPTER 4

COVE

After determining the system design guidelines, I carried out a six-month participatory design phase to investigate specific interface solutions with users. I provided working prototypes of the system to facilitate user feedback in the context of the users' daily routines [15]. The preliminary prototype platform was NASA's World Wind open source geobrowser, modified to provide bathymetry and display oceanographic datasets using a variety of techniques. This was soon replaced by an original software code base to afford increased design flexibility and to meet a key user need for availability on non-Windows computers. For each version of the prototype, the goal was to assess effectiveness in real world situations and solicit immediate feedback on specific approaches from users. Issues could thus be discussed directly with users to determine possible solutions, modify the overall design, and alter specific details of the system for further testing. Through this iterative process, user feedback could be quickly assessed, incorporated, and tested to determine optimal solutions.

A key finding in this phase was that integration of capabilities was crucial to the overall system design. Although specific design elements could be found in existing systems, switching between systems to carry out tasks imposed major impediments. For example, when laying out cable for the observatory, repeatedly switching between a GIS package or paper map to check bathymetry and a spreadsheet program to check costs, made observatory design a process with many interfaces, many data conversions, and many opportunities for errors. An integrated environment, on the other hand, soon proved much more effective when doing this task. This approach is also consistent with my gap analysis, which indicated that lack of specific techniques was less a primary impediment in data exploration than user cost associated with techniques [110]. When multiple techniques are necessary to perform a task, the overhead of each system adds to the total cost. In attempting to lower this user cost,

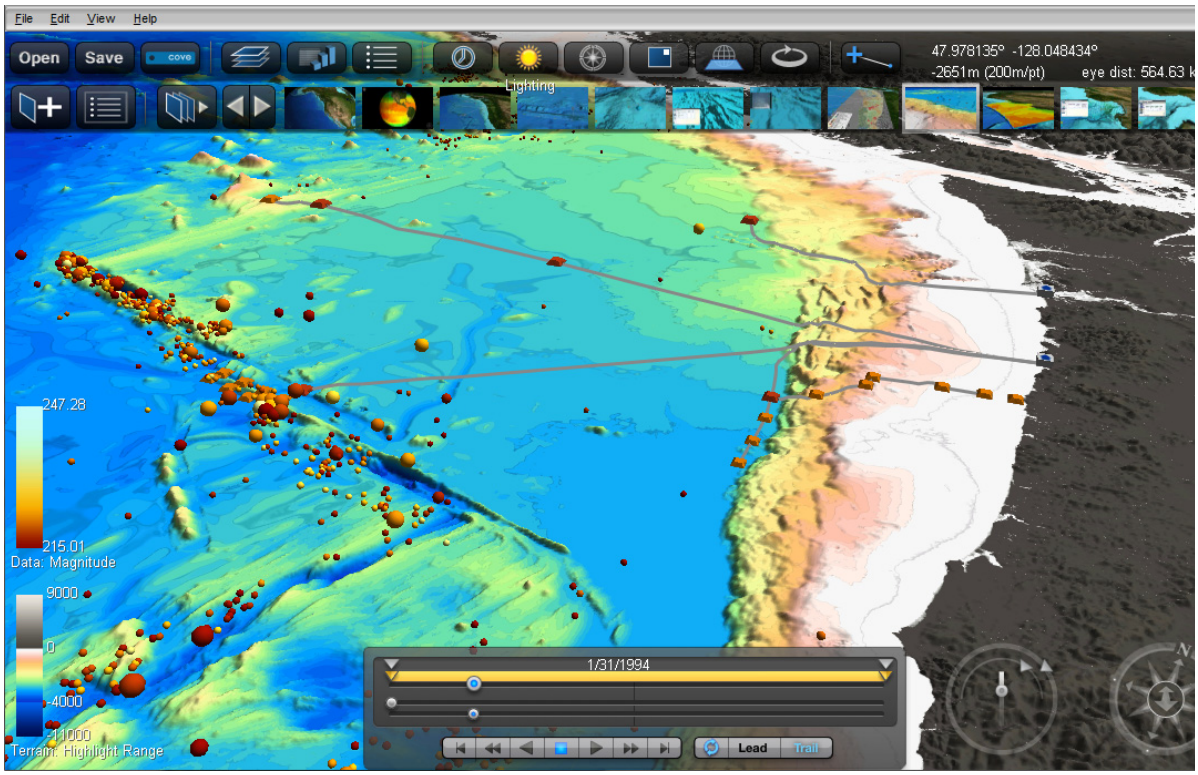

Figure 4.1: COVE displays geolocated scientific data, seafloor terrain, terrain specific color gradients, and instrument layout.

much of the participatory design phase focused on integrating techniques to improve the users' work processes.

The result of this effort is the Collaborative Ocean Visualization Environment (COVE) shown in Figure 4.1, a novel science tool designed from the guidelines outlined in the previous section and long-term collaboration with ocean science users. The user interface is modeled on a geobrowser paradigm much like that of Google Earth and similar systems [31, 65, 68]. A geobrowser provides an intuitive multi-scale interface, a familiar geographic context, and a simple layering metaphor to help organize diverse data. It also lets scientists interact with layouts and datasets ranging from the hundreds of square miles covered by these projects, down to a few meters around a sensor in an experiment.

Existing geobrowser interfaces have many features in common that are available in COVE. A layer tree is provided to organize different items that are geolocated in the main

window. A window menu and button bar provides access to commands, and context menus are available for object-specific commands when in the main earth viewer window. On-screen controls provide support for navigation, timeline functions, and an overview map for quick navigation. Beyond these base features, several new capabilities were designed into COVE based on feedback from scientists. Listed below, these capabilities differentiate COVE from existing geobrowser systems and provide an integrated solution to needs highlighted in the preceding chapter. Capabilities are grouped into four categories and described in more detail in the following sections.

Visualizing and Exploring Data

- Geolocated 3D scientific visualization
- High resolution (sub-meter) custom bathymetry
- Interactive ocean model data exploration

Instrument Layout and Management

- A drag-and-drop instrument management interface
- Layout status tracking and integration
- Swappable instrument and cable libraries

Collaboration and Communication

- User-defined shareable interactive views
- High resolution images and movies
- A Web-based repository for data and visualizations

Architecture and Implementation

- Scripting-based interface for importing new data formats
- A Web-based workflow solution for sophisticated data analysis
- The ability to run across local, server, and cloud environments

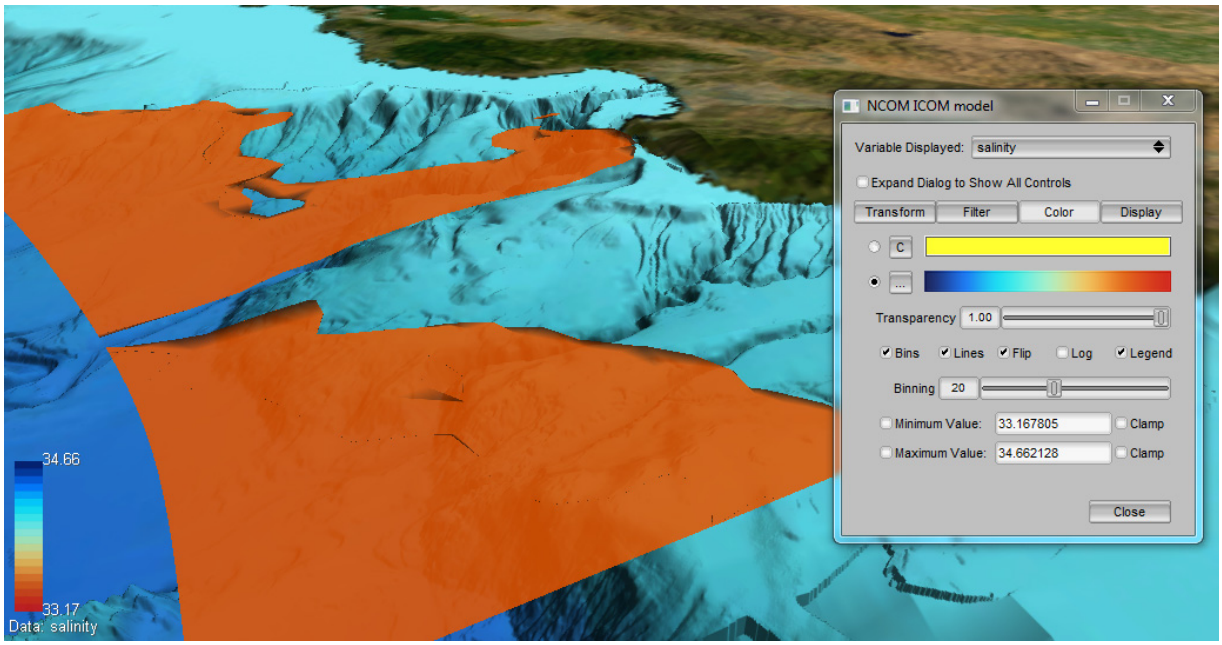

Figure 4.2: COVE is shown displaying a salinity *iso-surface* in an ocean model, providing a variety of coloring, filtering, and alternative display techniques.

4.1 Visualizing and Exploring Data

As noted previously, ocean observatories encompass several different types of data. Contextual data includes high resolution bathymetry (seafloor terrain), geological maps, and site features, such as telecommunication cables and navigation hazards (e.g., ship wrecks). A variety of observed data is collected, which may consist of point data collected by sensors, sonar data, or images and video. Finally, simulated ocean models provide 3D data over time for increasingly large sections of the ocean. Scientific visualization systems provide detailed viewing of specific data types in each of these categories. Existing geobrowsers support limited concurrent viewing of many of these datasets in the form of geolocated images, lines and points. COVE combines these capabilities to provide an integrated visualization of datasets in a familiar geographic context.

4.1.1 Geolocated 3D Scientific Exploration and Visualization

COVE lets users view multidimensional scientific datasets with a variety of interactive visualization techniques not afforded by geobrowsers. In Figure 4.2, COVE displays a salinity *iso-surface* (a 3D surface representing a specific value of a variable) in an ocean model, providing a variety of coloring, filtering, and alternative display techniques. The goal of the capabilities COVE provides is not to deliver an exhaustive selection of specialized scientific visualization packages. Rather, it is to simplify the interface by choosing those features most commonly used by scientists while still allowing a wide range of interactive and visual capabilities. These include:

- Native input and output of geolocated NetCDF files [46] and common text formats
- Data representation as 3D points, paths, vectors, surfaces, and iso-surfaces over time
- Interactive support for data transformations, such as filtering, scaling, and re-sampling
- A wide selection of color gradients with color binning, and logarithmic scaling
- Screen affordances, such as lighting controls, data legends, and 2D data plot overlays

4.1.2 High Resolution Custom Bathymetry

A primary user frustration with existing systems is inherent limitations for displaying custom bathymetry, crucial to many scientists' work. Geobrowsers are limited to images or a selection of datasets publicly shared on servers, while visualization tools are complicated. COVE lets users load arbitrary combinations of bathymetry and assembles them in real-time, showing the highest resolution terrain available for any point. Images of the high resolution terrain can be created for use in COVE or exported for integration with existing geobrowsers, such as Google Earth.

To provide visual cues for the terrain, scientists can apply customized, depth-based color gradients with contour lines and binned color values, as shown in Figure 4.3. Separate color gradients can be applied to the seafloor, land, and a highlighted depth range to accentuate specific bathymetry. Based on user feedback, techniques to provide further visualization

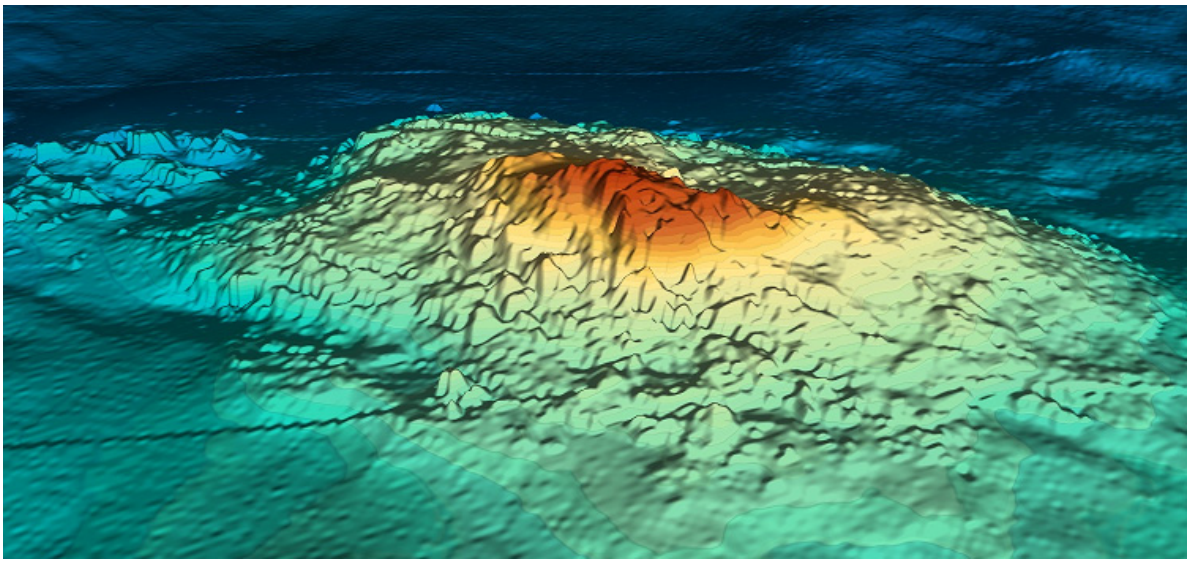

Figure 4.3: An example of COVE's high resolution bathymetry highlighting a seamount.

cues have also been included: extreme bathymetric scaling (up to 100 times), interactive shading and lighting direction, and user control of terrain resolution to provide more detail to aid visualization or less detail to enhance interactivity.

4.1.3 Interactive Ocean Model Data Exploration

Creating interactive tools for exploring data slices and flow in ocean models was one of the more fruitful collaborative efforts. 2D vertical or horizontal sections of model data provide an easier way for users to understand how variables change in the water column or at a specific depth. To support this, COVE can display slices that are part of the model as well as interpolated surfaces at user-selected latitudes, longitudes and depths. The user can also interactively create a custom path through the dataset or view a specific data point or water column (location from the surface to the seafloor) over time. These representations can be plotted in an overlay window and in the geobrowser environment, as shown in Figure 4.4, which displays a custom vertical section for salinity along the main channel of Puget Sound.

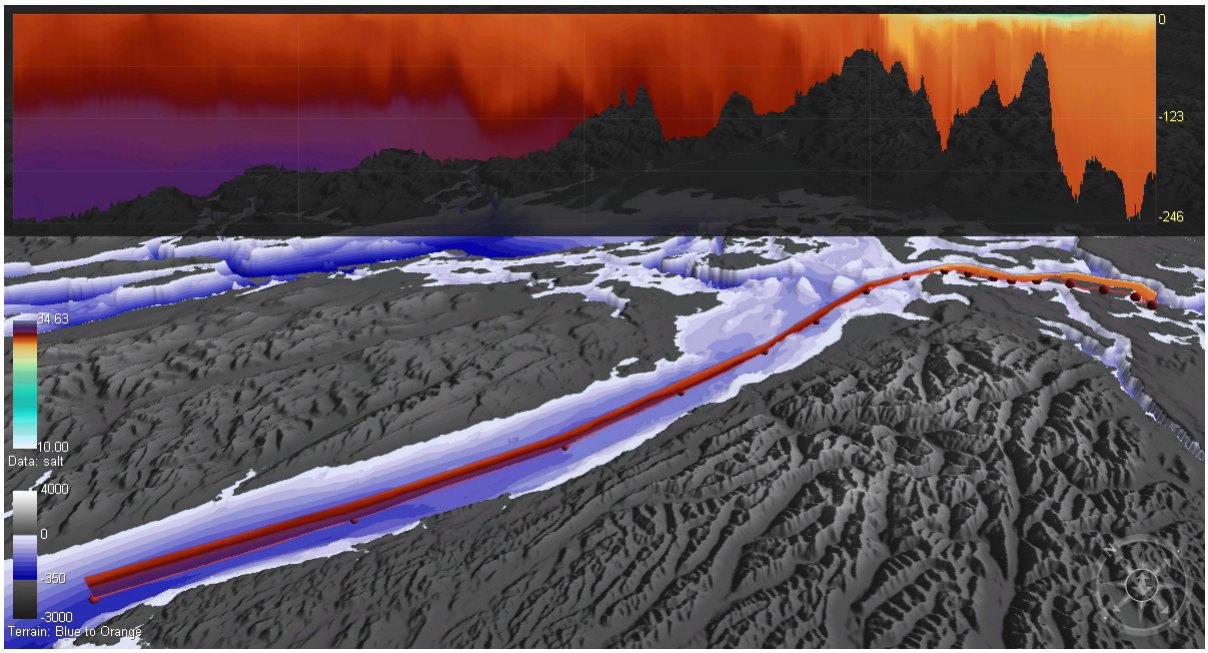

Figure 4.4: An example of interactive ocean model exploration with a custom vertical slice of salinity displayed in both the channel and overlay window.

Particle advection visualizations let users explore how objects change position over time based on model flow variables. Through COVE's interactive particle interface, users can create thousands of particles at arbitrary locations and depths and then drag single particles or groups to watch their paths update dynamically. The particles can be colored based on a range of values, such as total velocity, component velocity, or time released. They can be tracked forward in time to see projected paths, backward in time to see possible origins, or released over time from a point to determine, for example, how an uncapped oil leak might progress.

4.2 Instrument Layout and Management

Many observatory instruments, sensors and delivery vehicles are expensive and scarce resources. Optimizing their use can greatly increase the amount of data collected on a mission, while complications can lead to missed opportunities to collect important data or the loss of instruments. Observatories require the management and monitoring of hundreds to

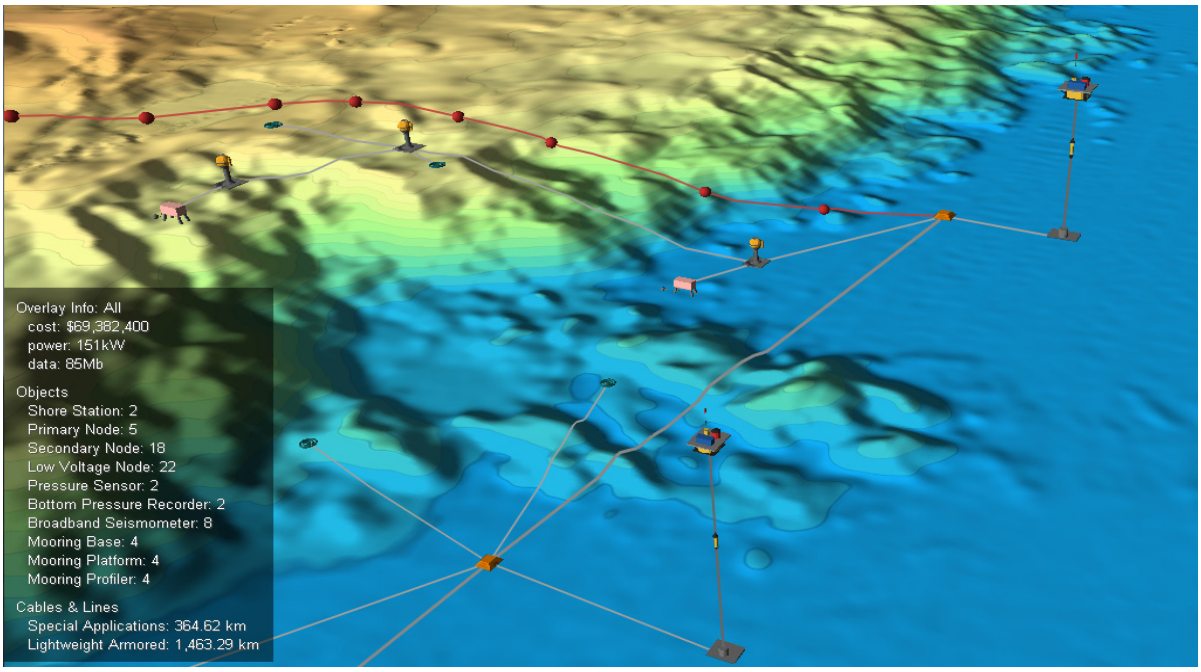

Figure 4.5: Sophisticated instrument layout can be easily created, modified, and shared. On the left a heads-up display provides instant feedback to monitor budgets.

thousands of heterogeneous assets and instruments collecting data at any given time, independently changing state based on detected events, and possibly being completely re-tasked to focus on major ocean occurrences, such as earthquakes, volcanic eruptions, or storms. While geobrowsers normally provide an interface for positioning objects and lines, their interfaces are limited and not designed for connecting objects with cables. Traditional science visualization tools provide no direct support for this process. While image editing tools and 3D modeling packages provide high quality results, they are designed for graphic specialists. COVE provides capabilities that enhance user interaction, on-screen feedback, and customization in the geobrowser interface to enable planning and management of observatory assets and instruments by the scientists themselves.

4.2.1 A Drag-and-Drop Instrument Management Interface

Existing geobrowser systems use direct manipulation drag-and-drop interfaces to add and position lines and objects. Objects can be icons, common 3D shapes, or 3D models, and

information summaries for any object can quickly be viewed as text in a pop-up window or as a detailed Web page in a Web browser. COVE extends this interface by providing an array of cable types that can be added to connect the objects (Figure 4.5). Positioning handles are available for maneuvering cables around obstacles on the seafloor, and a cross-section view is provided to quickly inspect the cable run for extreme climbs or drop-offs. Collections of objects and cables can be grouped into instrument templates to allow easy insertion of complex packages, such as geodetic arrays, containing several connected components. To enhance the user's ability to safely explore different layout possibilities, multi-level undo and redo commands are available on all layout activities. These commands, commonly available in visualization and 3D modeling packages, are rarely in geobrowsers.

4.2.2 Layout Status Tracking and Integration

To track layout status, COVE automatically updates heads-up displays on the screen during editing sessions, providing instant status on characteristics of assets and instruments, such as budget, current cost, and length of cable (Figure 4.5). Having budgets and installation timing readily visible and costs automatically updated with changes lets layouts be quickly refined to meet specific goals. Deployment time and cost can be associated with each instrument and cable to view how the layout evolves as instruments come on line. A table view lets users see all instruments and cables in a specific layout in tabular form, which can be easily exported to spreadsheet applications. Layouts can also be exported and imported using *XML* to integrate with geobrowser systems. Import of layouts in other formats is also possible through the scripting interface, which is discussed in Section 4.4.1.

4.2.3 Swappable Instrument and Cable Libraries

I observed that lists of available layout objects and their properties were often updated due to the availability of new instruments and cables, different staff in the observatory having different needs, or a user's needs changing over time. Therefore COVE's design lets users swap in task-specific libraries of instruments and cables (Figure 4.6). Each object or cable is based on a user-modifiable type, with several settable features available to define appearance,

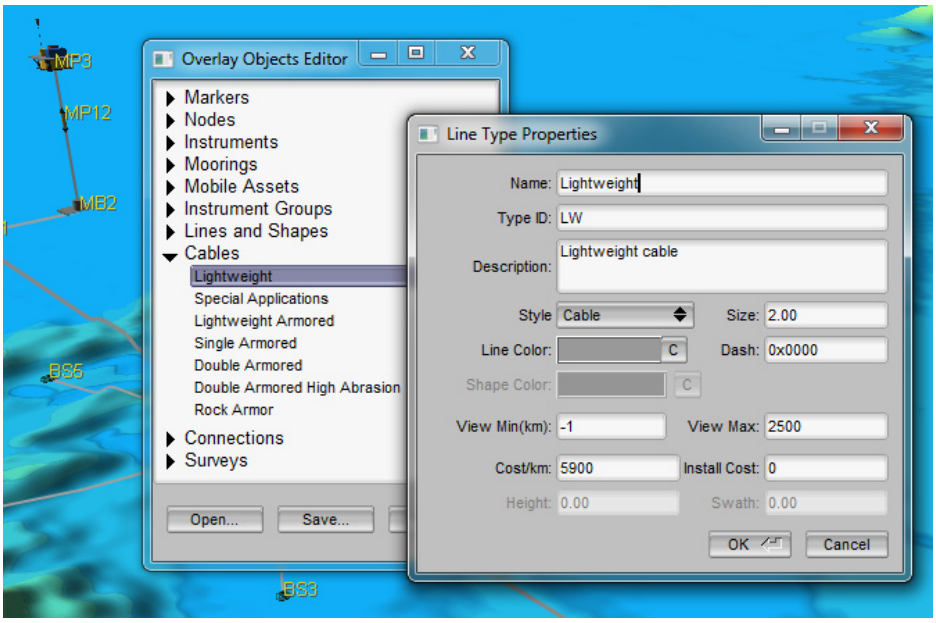

Figure 4.6: Sophisticated instrument layout can be easily created, modified, and shared with collaborators.

cost, and cabling requirements. New types can easily be added, based on existing instruments and cables, or deleted to remove unused options. The libraries can also be created dynamically from a central database containing the most up-to-date working set for a user group.

4.3 Collaboration and Communication

Observatories bring together scientists who may not have previously worked together, and for some projects, scientific fields that may not have worked together. The observatory is thus creating an infrastructure that is designed to facilitate non-oceanographer participation in ocean experiments. Furthermore, all the sciences are finding it more and more important to reach out to policy makers and citizen scientists to affect change, create awareness, or raise funds. Geobrowsers use files to share collections of links, images, and packaged tours through an environment, but they do not provide an extensive range of visual outputs. Visualization tools provide high-quality image and movie outputs, but producing them is time consuming and their static natures make it difficult for viewers to explore environments

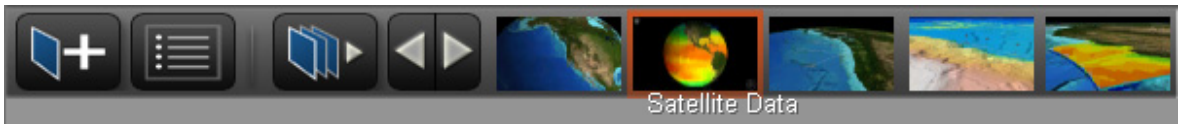

Figure 4.7: The COVE view panel which allows users to quickly create, edit, plan and select a preset visual of a collection of datasets.

in more detail. COVE speeds and simplifies the creation of high quality visuals that promote further exploration for richer collaboration across teams and effective communication beyond the team.

4.3.1 User-defined Shareable Interactive Views

WorldWide Telescope (WWT) demonstrated the incorporation of user-defined views for end user exploration of virtual observatories [39]; a set of thumbnails on a toolbar at the top of the screen provides a link to an image in their database. Similarly, interactive views can easily be created and stored in COVE to save a specific set of camera, layer, data, and visual settings. These views can then be invoked locally for exploration and discussion or shared to a Website. Once shared, other team members can download and select from the view panel to quickly start exploring the data (Figure 4.7). A set of views can also be automated, like a slide show, to provide a guided tour through different aspects of the site and data.

4.3.2 High Resolution Images and Movies

Where interactive use of COVE is not feasible, the user can create high-resolution images and movies to communicate. The output resolution can be set to any dimensions and is only restricted by the size of the graphics memory. The output can also be generated from any collection of existing views, allowing several disparate images to be created via one command or stitched together in a multi-scene movie. These settings can also be saved as a visualization script and later used to generate the visualization from command line or remote server versions of COVE. Thus, once users have established a set of visuals that need to be created repeatedly for analysis, they can be produced automatically at the data source and sent to the user rather than requiring data to be transmitted and stored locally.

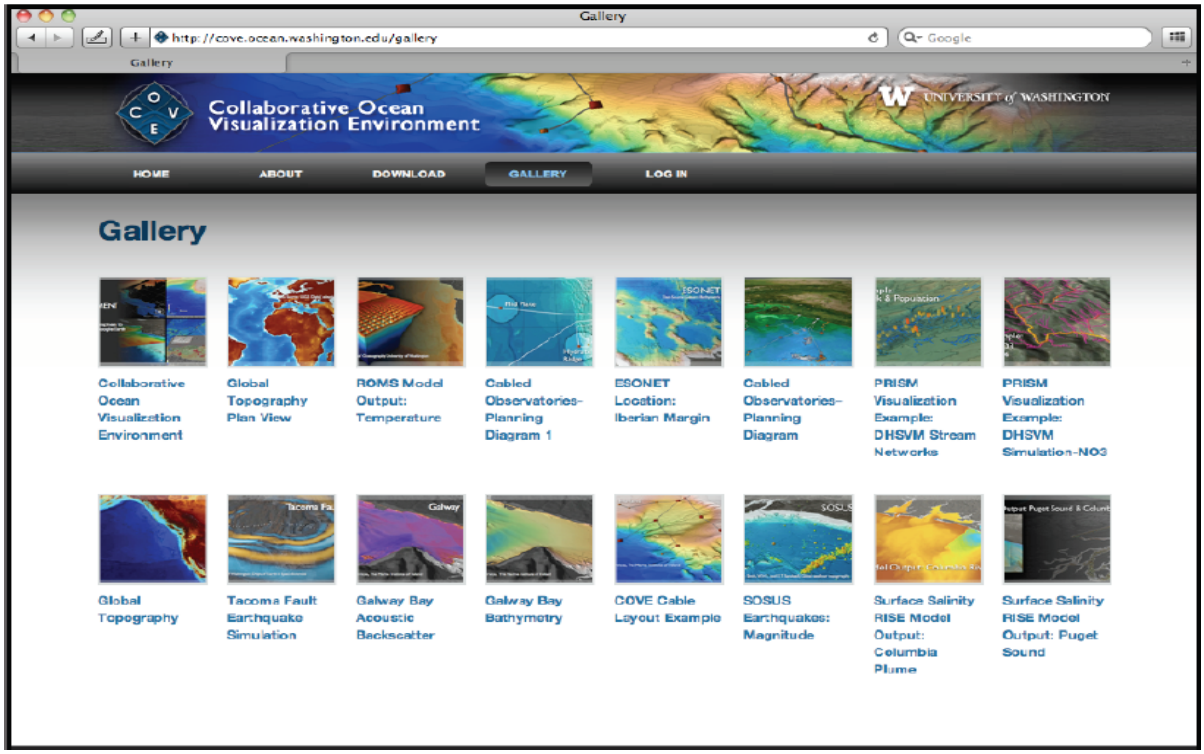

Figure 4.8: Views are easily created in COVE and can then be uploaded to a Website to facilitate sharing visualizations of experiments and data.

4.3.3 A Web-based Repository for Data and Visualization

A server-based data and visualization repository makes it easy to upload and share new views and datasets with the rest of the team. Once shared to the server, an image or movie is displayed with metadata, which provides information such as the author and creation date. Anyone with access to the server can then view the available visuals through a Web-based interface, as shown in Figure 4.8. To inspect the view further, users can download a high resolution version of the image or movie, or, if the author has made it available, the entire COVE file. In this way users can focus on sharing the story they want to tell with the data; sharing the data for exploration by others is an equally straightforward process. From directly within COVE, datasets, bathymetry, and layouts can also be shared to the server for access by the team.

4.4 Architecture and Implementation

The goals for the COVE architecture are: (1) to provide rich cross-platform interactive 3D graphics, (2) to offer a flexible internal design for prototyping new solutions, and (3) to utilize the Web to help scale data handling needs. The implementation details of COVE, while not essential to the design, are concisely provided here for completeness. The prototyping system required both geobrowser and visualization technologies. The original implementation plan was to leverage an existing geobrowser as the basis for COVE, but this was precluded due to limitations in existing APIs, cross-platform support, graphics capabilities (e.g., Web-based solutions), and stability of existing systems. COVE was therefore implemented as a C++ open source project, with cross-platform support provided by the OpenGL graphics interface and the Fast Light Tool Kit (FLTK) interface toolkit [25]. Publicly available libraries were included for accessing NetCDF [46] files, and the NetCDF CF-Metadata naming conventions were used to standardize identification of position and time variables. Below are three specific extensions to the standard geobrowser architecture that are elements of the COVE design.

4.4.1 Scripting-based Interface for Importing New Dataset Formats

A constant user need was the ability to load newly available data into COVE. Due to the diversity of datasets, it quickly became time consuming and infeasible to alter the code for each data format. To resolve this issue, the design was extended to support a scripting language for data input. The specific language chosen was python, a popular open source interpreted language becoming increasingly common in scientific programming [83]. The design provided a set of function calls to the scripting language that securely exposed internal data handling functionality. It also allowed callbacks into the system for simple dialogs to support visual integrity with COVE. With more time and effort, this approach could be used as an all-purpose way of extending the system, but this level of integration was unnecessary for the existing user requirements.

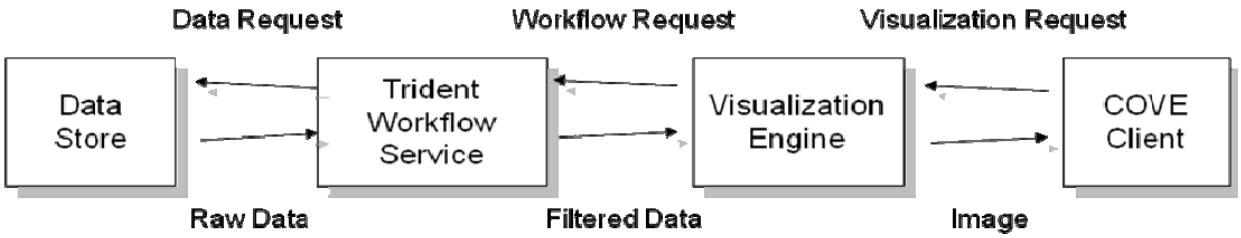

Figure 4.9: The COVE architecture consists of a set of components that can be distributed across local, server and cloud resources to support data exploration needs.

4.4.2 A Web-based Workflow Solution

A Web-based workflow solution allows the use of external resources to address file size, arbitrary formatting, and extensive manipulation often required to explore and visualize data. COVE can initiate, monitor, and download workflow results through simple calls to a Web service. To support this, an extensive workflow library was built to allow re-use of COVE's routines, similar to that provided for scripting described previously. By combining workflow and direct manipulation capabilities, interactive exploration of large datasets becomes possible. Microsoft Trident, a Windows .NET based workflow service, provides this functionality for COVE [9]. An example of the Trident user interface is shown in Figure 7.2 and a more detailed description appears in section 7.1.

4.4.3 The Ability to Run across Local, Server, and Cloud Environments

The COVE architecture consists of a set of layers that can be distributed across local, server, and cloud resources to support a wide array of data exploration and visualization problems (Figure 4.9). Each layer can be tightly bound to the next or accessed through Web service interfaces. For example, the COVE client can be tightly bound with the visualization engine to enhance interactivity (the default configuration) or be Web-based to view results on a variety of platforms. The visualization engine can take advantage of a local GPU or run in software for non-GPU environments, such as cloud services. The data store is file-based to handle legacy datasets and can run in any environment that provides a Web server. The

workflow service and entire architecture will be explored in more detail in Chapter 7, where further analysis will show the importance of this flexible architecture for observatory needs.

4.5 Conclusion

My close collaboration with scientists yielded an extensive set of additions to the standard geobrowser interface to solve issues in data visualization and exploration, observatory design, collaboration and communication, and cross-platform architectures. In the next three chapters, I present evaluations of COVE via a range of methods to illustrate the effectiveness of the original design and determine improvements. The first is a *user study* that investigates data exploration and visualization by users with a wide range of ocean data expertise. The second is a set of *real-world deployments* in science environments that tests instrument layout, collaboration, and communication. The third is a *quantitative analysis* of the COVE architecture to determine its scalability based on a representative set of data exploration and visualization workflows.

It is critical to reiterate that *integration* of COVE's capabilities consistently received the most enthusiastic responses from users. Thus, while the additions to COVE were organized into categories for clarity in this chapter, an important objective in the evaluations was to determine synergies amongst capabilities that add value beyond the features themselves.

CHAPTER 5

USABILITY EVALUATION

This chapter presents usability evaluations I conducted to determine the effectiveness of COVE in two ways: (1) by studying experts and novice users in ocean data exploration and (2) by studying visualization producers and consumers. The first study assessed COVE's data exploration utility across users of differing skill levels as they performed potential observatory tasks, and the second assessed COVE's effectiveness as a tool for creating and viewing interactive visualizations.

In the first study, novice and expert users in ocean data exploration performed prescribed tasks in COVE. The user's execution time for each task was measured, along with his or her level of accuracy. An instrumented version of COVE logged user activity, and participant observations were recorded manually during these sessions. At the end of the session, participants completed surveys, and a subsection was interviewed to collect more detailed responses.

The second study, focusing on visualization producers and consumers, was carried out in concert with the first, but was assessed more qualitatively. I observed a group of producers as they created COVE views that were later used for the interactive environment in the first study. I recorded observations and comments during these sessions and collected feedback from participants after completion of each visualization. To gather information from the consumers, I asked them about the effectiveness of each visualization in helping them complete the prescribed tasks, both during the session and via ratings in a post session survey.

The sections below describe the experiment setup, evaluation session, and results for each usability study. The results assisted in determining which elements of the original design were validated, which needed to be modified, and what open challenges remained.

Figure 5.1: On the left are several groups working on prescribed tasks. On the right is a single group examining a salinity model for Puget Sound as part of the evaluation.

5.1 Ocean Data Experts and Novices: Study 1

With a wide range of scientists and general users exploring observatory data, it is important that a system provides support both for experts and novices in ocean data exploration. This study's goal was to determine COVE's ease of use for both types of users and to collect feedback on the effectiveness of specific capabilities discussed in the previous chapter.

5.1.1 Experiment Setup

The user group for the evaluation consisted of 30 participants who had not previously used COVE. The novice group comprised 24 participants who were asked to carry out the prescribed tasks as part of their Introductory Ocean Science class laboratory. This group worked in teams of 2-3 on 11 Windows workstations, as shown in Figure 5.1. The expert group consisted of 6 participants, who carried out the tasks on their personal systems, which included a mixture of Windows and Macintosh environments. The testing was conducted at the Spatial Analysis Lab in the Ocean Sciences Building at the University of Washington for the novice group and in the offices of the expert users. In each case, COVE and the content file prepared for the assigned tasks were pre-installed on user systems.

All participants were asked to perform 6 general ocean data exploration tasks and answer 4-7 questions about each area. Table 5.1 lists the general task areas and provides a short description of each. The complete set of tasks and questions appears in Appendix B. Participants were given a maximum of four hours to complete all tasks and were instructed to proceed as quickly as was comfortable, but at a pace that allowed them to answer the questions accurately. For each exploration task, pre-created views on COVE's view panel provided a preset interactive environment. An example of two such views is shown in Figure 5.2.

Table 5.1: Task areas covered in usability study 1.

Task ID	**Description**
Bathymetry	Determine depths at locations and compare terrain features.
Surface Data	Inspect surface salinity and temperature values and compare locations.
Cross Section	Explore salinity and temperature ocean model cross sections.
Comparison	Compare model salinity with recently collected salinity data.
Circulation	Determine circulation and current vorticities based on a flow vectors.
Advection	Move particles to interactively determine their source or destination.

Quantitative data was collected from activity logs of user interaction while completing the prescribed tasks, as well as user responses to task questions. The activity logs were created by an instrumented version of COVE which captured user actions and timing information. The logs were primarily used to determine task completion time, but could also be inspected for further analysis. The accuracy of task completion was based on correct responses to task questions.

Qualitative data was also collected in the form of user surveys, which the users filled out immediately after completing the prescribed work. The survey asked 30 questions about their user experience with COVE in various areas. Responses were elicited on a 5-point Likert scale, from extremely dissatisfied to extremely satisfied, with users having the ability

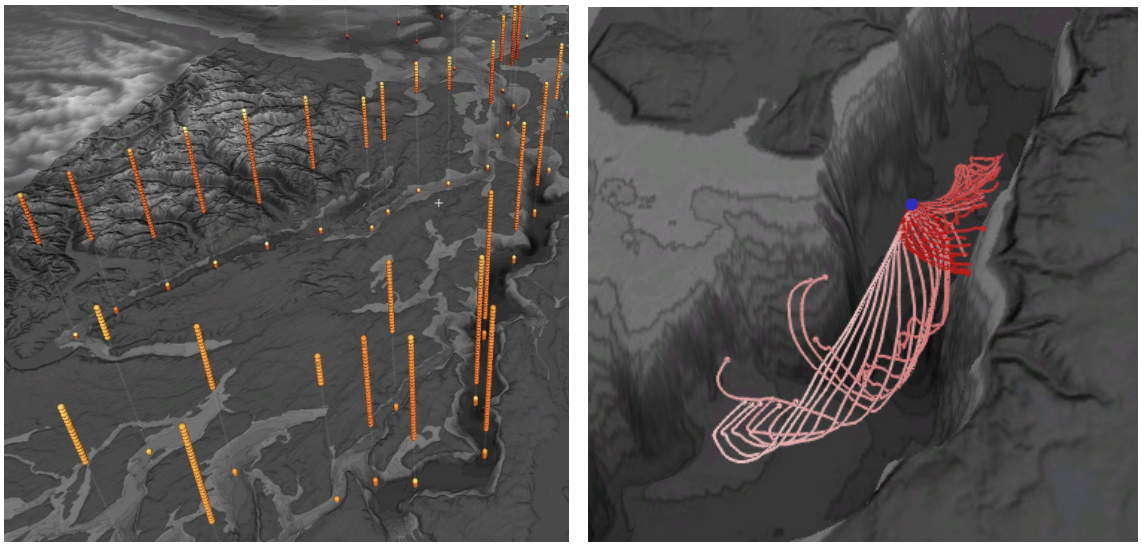

Figure 5.2: An example of the interactive views used to carry out the prescribed tasks. On the left are soundings collected from Puget Sound for the Comparison task. On the right are particles released over time to support the Advection task.

to comment further on an area, if desired. Because a recent class assignment used a combination of Google Earth and ArcGIS, a sophisticated map creation and exploration tool, comparisons of COVE to these tools were also requested from all participants. The areas covered in the survey are listed below.

- User interface elements
- Presentation of bathymetry
- Data exploration capabilities
- Use of interactive views
- General user experience and comparison with similar tools

5.1.2 Evaluation Session

Users were initially given a written overview of the data exploration session and instructions to get started in COVE. The overview compared COVE to Google Earth and described basic navigation, visual controls, dashboard commands and the view panel. A short demonstration of COVE and an overview of the tasks were then provided. Instructions on how to use

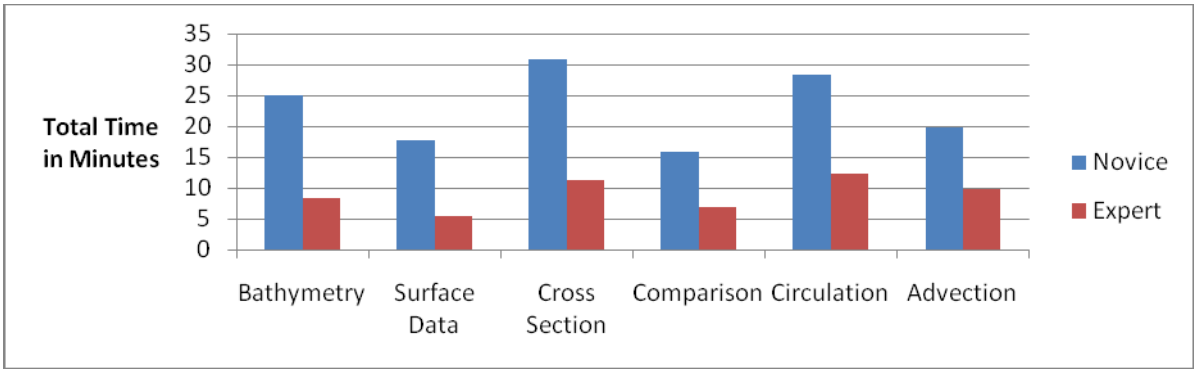

Figure 5.3: The average time for task area completion for each level of ocean data exploration experience.

COVE were also available in a help file on the system. General questions were answered prior to the start of the formal usability evaluation. Specific inquiries concerning the prescribed tasks were answered during the session to clarify the intent of a question. The task areas were designed to be completed sequentially by the users, and each area was designed to take approximately 5-10 minutes by experienced COVE users based on sampling before the study. After completing the tasks, the participants filled out the survey and activity logs were collected from each system.

5.1.3 Results

The average time for task area completion for both levels of ocean data exploration experience is shown in Figure 5.3. This figure shows that the experts were quickly effective at completing COVE tasks, with an average completion time just below 10 minutes per task. Novices took approximately 2.5 times longer per task on average, indicating that experience exploring ocean data was a strong determinant in task completion time. There is also variance shown in average task completion times for both groups. The greater time taken by both groups on the Bathymetry task is probably attributable to this being the first task performed using COVE. The increased time taken on the Cross Section and Circulation tasks is likely related to system performance issues, which will be discussed later.

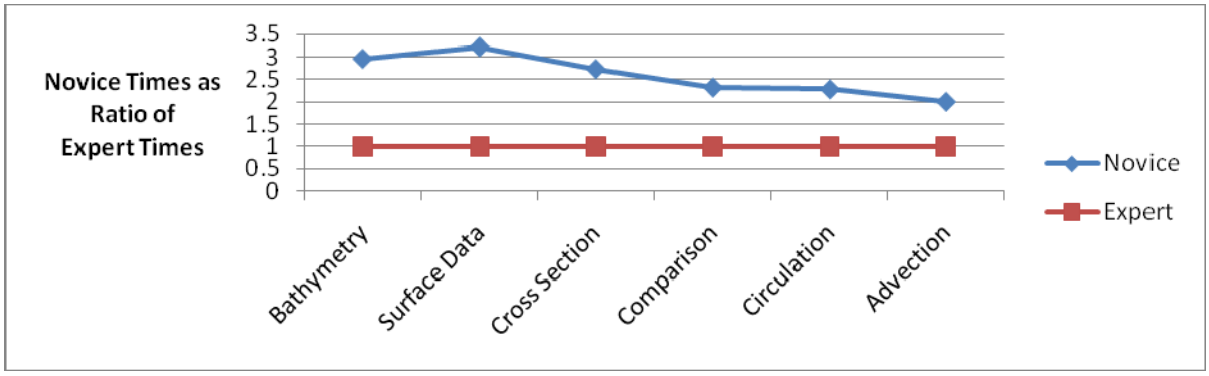

Figure 5.4: The average time for task area completion as a ratio of expert times. Novice users decrease the disparity between their skill level and the experts over time.

As each task required a different amount of time to complete, the participants' times were also analyzed based on the average time for task area completion *compared to the expert users*. This provides a relative measure of task performance between skill levels. These results are illustrated in Figure 5.4. Interestingly, over time, novices progressed from taking more than 3 times longer than experts to complete a task, to less than 2 times as long: as non-expert users became more comfortable with COVE's interface, they became more effective exploring data and thus decreased the disparity between their skill level and that of experts. This result is further substantiated in Chapter 6, where non-experts on real-world science deployments quickly became effective enough to carry out tasks normally confined to experts.

Experts were consistently more accurate in their responses to the 4-7 task questions. There was also no clear correlation between experience using COVE and task accuracy (as was noted with execution time); the correlation seemed primarily task dependent. In discussions with instructors, they noted the accuracy was in line with similar assignments, indicating novice users were no less accurate using COVE. More investigation is needed to determine whether task accuracy can be improved by becoming a more experienced COVE user, by performing similar tasks repeatedly rather than differentiated tasks as in this study.

The user surveys provided substantial feedback on COVE's design via comments and ratings for various capabilities on a 5-point Likert scale. This feedback is categorized into three areas: (1) ease of use, (2) effectiveness of specific capabilities for data exploration task

completion, and (3) comparison to other ocean data tools. In the tables below, specific scores are listed for each rated area, and cumulative results are averaged over all participants and listed to one decimal place in the right-hand column.

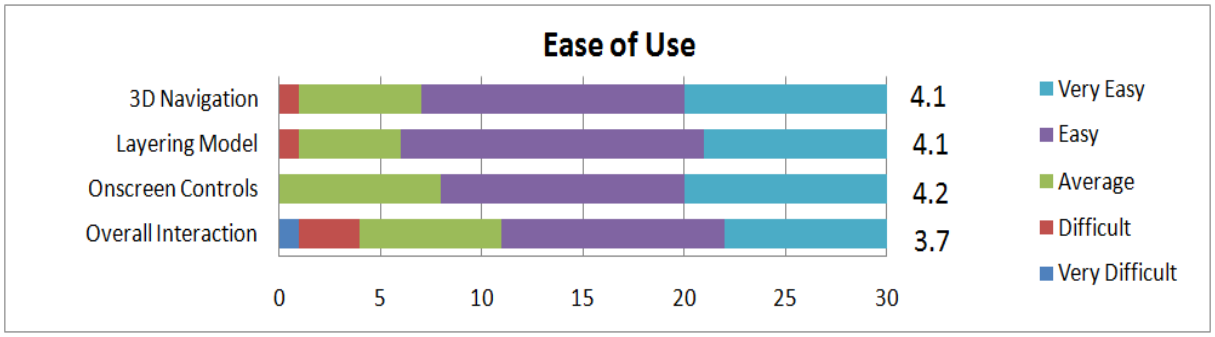

Figure 5.5: User survey results on a 5-point Likert scale, where 1 indicates *very difficult to use* and 5 indicates *very easy to use*, with the number of users selecting each rating as well as the average.

Observation of the users' interaction with COVE revealed their general ease with the interface, and few questions were raised during the session. This was corroborated in the survey, with both 3D Navigation and Layering Model averaging just over 4.1 for ease of use, as can be seen in Figure 5.5. The participants rated their experience with the Onscreen Controls a 4.2 for the Quality of Visuals and 3.7 for its Overall Interaction. Their comments indicated that the lower rating for Interaction was primarily due to system performance, which will be discussed further in the next section.

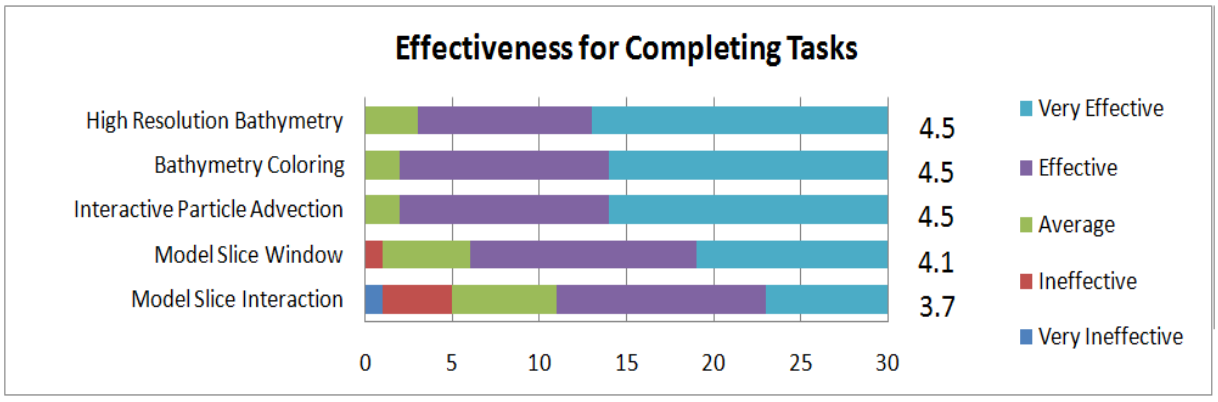

Figure 5.6: User survey results on a 5-point Likert scale specifying the effectiveness of a capability for task completion, where 1 indicates *not effective at all* and 5 indicates *extremely effective*.

The user rating for *effectiveness* of specific capabilities for completing tasks is displayed in Figure 5.6. COVE's High Resolution Bathymetry and Bathymetry Coloring were rated very helpful in understanding the data, which was particularly noted in the comments and in the survey ratings, with an average score of 4.5 in both categories. Interactive model exploration consistently received positive ratings from the users as reflected in the ratings, where Interactive Particle Advection scored 4.5, the simultaneous display of the Vertical Slice Window scored 4.1, and Vertical Slice Interaction scored 3.7. Users indicated the stronger rating for Advection was due to its high level of interactivity and variety of available display options. Conversely, the lower score for Slice Interaction was related to slow system feedback and lack of apparent controls for changing data display.

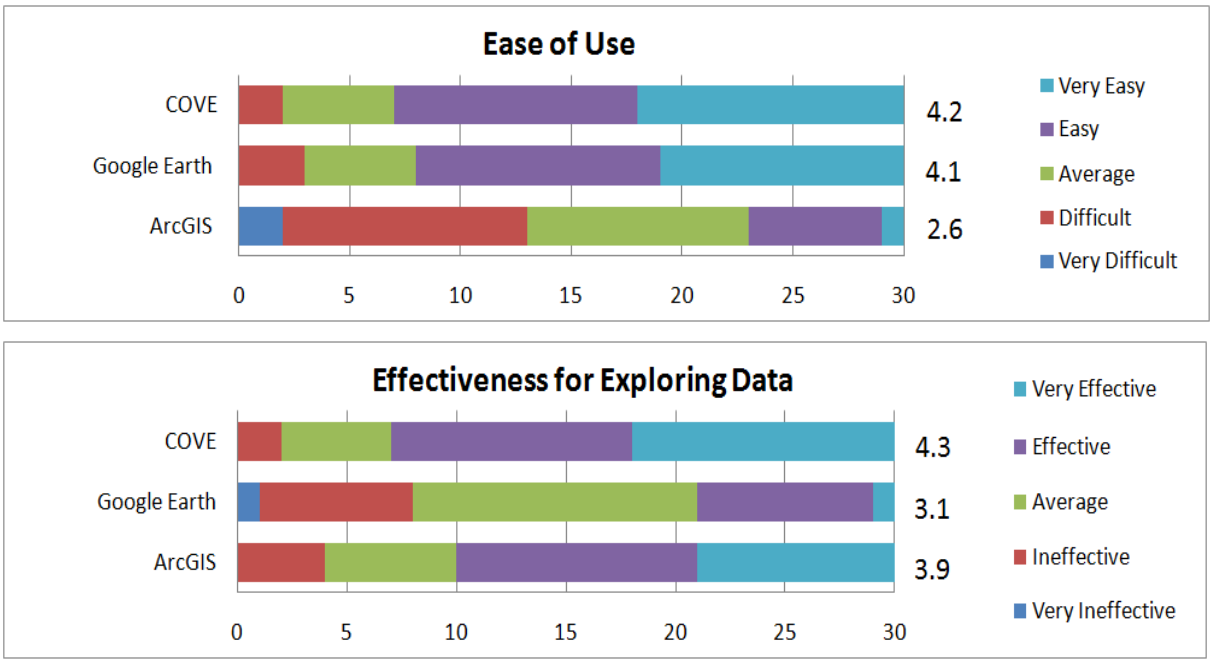

Figure 5.7: Comparison of COVE to alternative ocean data tools on a 5-point Likert scale for overall ease of use and effectiveness for exploring ocean data.

The survey also asked users to rate COVE, Google Earth, and ArcGIS for overall ease of use and effectiveness for exploring ocean data. Results are shown in Figure 5.7. COVE was the most highly rate in both categories, above 4.0 in both. Google Earth rated similar to COVE for ease of use, but much worse for exploring ocean data (3.1). ArcGIS, on the other

hand, rated poorly on ease of use compared to COVE (2.6), and also slightly less useful overall with a score of 3.9. When queried about this last score in more detail, participants noted that the terrain-specific tools in ArcGIS were more powerful, but COVE's ability to easily display a wider variety of data and show it together with the terrain was more helpful.

In order to test differences in these 5-point Likert ratings, I applied a Wilcoxon rank-sum test for both *ease of use* and *effectiveness*. In each case, an omnibus test showed differences across systems. Pair-wise comparisons were then conducted (applying a Bonferroni correction). The tests showed that the difference in *ease of use* between COVE/ArcGIS and Google Earth/ArcGIS was significant ($p = .001$, $p = .0015$), while the difference between COVE/Google Earth was not ($p = .556$). For *effectiveness*, pair-wise comparisons showed that the difference between COVE/Google Earth and ArcGIS/Google Earth was significant ($p = .001$, $p = .003$), while the difference between COVE/ArcGIS was not ($p = .31$). Based on these results, one can see that COVE is unique in providing both *ease of use* and *effectiveness*.

5.1.4 Design Modifications Based on Feedback

As was identified in the activity logs and through observation and feedback, performance was often a hindrance indicating **maintaining interactive performance across large datasets is essential** for effective data exploration. This issue was often specific to particular datasets, which should have been reduced in size for the evaluation. However, this user issue also illuminates how quickly real-world datasets can overwhelm a system. Furthermore, universally reducing display detail, while simple, is not always a suitable solution. Although performance is critical to explore data interactively, displaying detail when communicating is often equally important. This was particularly noted in the deployments described in Chapter 6, where frequent modifications to display detail occurred with large bathymetry sets.

To resolve this issue, the ability to lower terrain resolution through direct user calibration was made more obvious in the interface. A more extensive alteration to the design was made to enable automatic terrain re-scaling based on user activity. As the user moves, terrain resolution is lowered, and, when static, detail is increased. Finally, additional ways to filter

the number of data points being displayed in COVE was provided, along with user prompts to indicate when performance was being impacted by either bathymetry or dataset size.

The repeated changing of facets of data representation, as well as the inability to find some data display alternatives, indicated the need for **more immediate access to all data display options**. The system under evaluation offered data display controls in multiple dialogs that grouped similar controls together. This design required users to change modes several times in order to refine visuals; e.g. repeatedly swapping data coloring and data display format dialogs. In response to this issue, all data display dialogs were merged into a single tabbed dialog that allowed quick, mode-less switching between data display options. Another option was added to expand the dialog to show all tabs simultaneously on large screens, so that the need to switch tabs could be removed, and all options were immediately apparent for any dataset.

5.1.5 Open Challenges

While not a prime component of this study, one area requiring further investigation is the **effectiveness of experienced COVE users**. The reduction of the execution time gap between expert and novice data explorers indicated that, with more practice on specific tasks, this disparity may become negligible. Another study with users executing several tasks in the same subject area would be useful to determine how quickly novices can become as effective as experts, and what capabilities are most important in reducing the difference.

5.2 Visualization Producers and Consumers: Study 2

As discussed in Chapter 3, it is becoming increasingly important to be able to share interactive visualizations instead of datasets, both to minimize data transfer and to help frame and communicate insights. This study investigates COVE's usability for visualization producers as they create interactive visualizations and for consumers who use them to explore data.

5.2.1 Experiment Setup

To evaluate COVE for visualization creation, qualitative data was collected from 4 content producers, each of whom created interactive views for one or two of the general task areas in Table 5.1. All were experienced at creating ocean data visualizations for a variety of audiences, with two having approximately 4 years experience each, and two having over 10 years each.

The consumer group consisted of all 30 participants who completed the data exploration tasks in the previous user study. As the visualizations were the same ones used to complete the prescribed tasks, activity logs were available for all participants at the end of the session. The previously noted user survey also asked consumers to rate each visualization with respect to its effectiveness for task completion.

5.2.2 Evaluation Session

The producer observation session took place in the participant's normal work environment. As each producer created a view in COVE, he or she described aloud what they were doing and why. Answers were provided to specific questions about how to accomplish an action in COVE, but only as a general guide, so that users were left to resolve the issue themselves. Notes were taken as users carried out each task, and each user was informally asked for further feedback upon completion. The sessions were relatively unstructured, as each task was unique and required different COVE capabilities. Producers were encouraged to explore and comment on the complete capabilities of COVE during the session.

Visualization consumers interacted with the created visuals as discussed in the previous study. While carrying out the prescribed tasks, they were polled about effectiveness of the visualizations for task completion to elicit comments. As noted, activity logs and user surveys were then collected at the end of the session.

5.2.3 Results

Saved interactive views and the view panel were used extensively by producers while creating content. They noted how the views made easy it to iteratively compare different approaches and frame the exploration environment for viewers. The views also helped producers work with their fellow producers. "*After I made the first view, I was able to make 6 more that I could go over with [my colleague] to figure out the best one.*" And when it came time to create the final collection for the session, producers found the view panel highly useful. "*The view panel made it so easy to roll up all the views into one file at the end and lead the students through the data the way I wanted to.*"

Figure 5.8 shows the consumer ratings for viewing COVE visualizations. They rated the specific views supplied as highly effective in helping complete prescribed tasks, with an average of 4.2. They also found the View Panel easy for selecting views, with an average 4.3 rating and for Creating and Managing Views with a 4.1 rating. However, as can be seen in the Figure, the latter capability was only used by 50% of the participants due primarily to the fact that it was not necessary for task completion. Overall, consumers found the views and view panel functionality effective, but mostly restricted view panel use to switching between views. Future studies will need to determine how this feature can engage consumers more broadly.

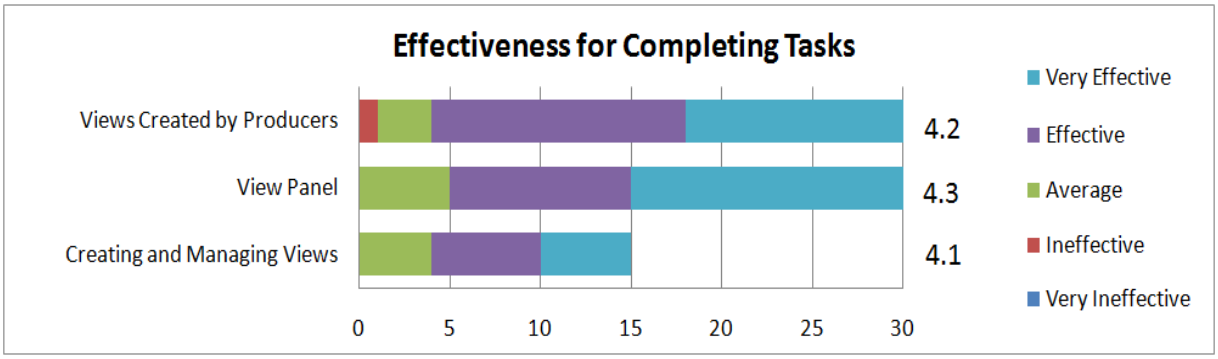

Figure 5.8: User survey results on a 5-point Likert scale, where 1 indicates *very difficult to use* and 5 indicates *very easy to use*, with the number of users selecting each rating as well as the average.

Two of the producers also qualitatively compared COVE with a similar tool for production of interactive visualizations for classroom consumption. They had previously used *Virtual Puget Sound (VPS),* a system that provided a representation of Puget Sound and a 3D navigational model [114]. It afforded slightly better data representation than Google Earth, but had limited capability for interactive data exploration. One producer noted, "*COVE let me ask more detailed questions and questions in completely new areas, like particle advection, so I was able to do a lot more in the lab than last time.*" They also noted that the final visuals were significantly better than those from VPS and superior to other tools previously used.

5.2.4 Design Modifications Based on Feedback

One finding was that **rich data coloring options were necessary** to encourage deeper engagement with the system. The COVE system under evaluation featured a single legend and a limited selection of fixed gradients. While this was a significant improvement over standard geobrowsers, it meant that visualization producers had to trade off elements in what they could show, and consumers had to switch between options to investigate data. To solve this issue, COVE added several new capabilities: a wider selection of built-in gradients, the ability to load custom gradients created in other programs, color binning, logarithmic color scales, and multiple custom data legends for both data and terrain.

Figure 5.9: A visualization producer using COVE in the classroom to communicate to visualization consumers at the beginning of the session.

I observed that visualization producers were constantly fine-tuning details in their work, and **scene lighting options were repeatedly accessed**. Lighting was often crucial to highlight a specific feature on the seafloor. While this functionality was available in COVE, it required several clicks to modify and evaluate results. To solve this issue, a custom lighting control was added to the primary COVE window. This lets users quickly set lighting height and direction as well as shading level, which are immediately reflected in the results. These attributes can also be saved with the view, along with other scene details, to ensure the visualization appears as intended.

For producers, **the programming interface was not a substitute for a good UI** to access data manipulation features. To support workflow and provide flexibility to users, a scripting interface is available to access COVE's extensive data handling library. While working with visualization producers, I quickly determined that requiring any kind of programming effort to use these capabilities would be largely unsuccessful. Many capabilities that required programmatic access were eventually integrated into the UI, and, in many cases, supported through some form of direct manipulation (e.g. the interactive particle advection interface) to eventually expose the desired capability. More investigation is required in this area,

especially regarding the viability of reconfigurable workflows as a solution for more sophisticated data manipulation.

5.2.5 Open Challenges

One request from producers was to take COVE's interactive model exploration capabilities one step further and **allow interactive manipulation of ocean model generation**. This lets consumers explore model output and increase their awareness about how specific inputs affect the final output. Researchers have shown the effectiveness of this approach for teaching purposes [88], and, in collaboration with COVE's visualization and exploration capabilities, this approach could provide an environment to illustrate the full range of ocean model issues. Given COVE's flexible data architecture, this would be a reasonable extension of COVE, but one well beyond the scope of current requirements.

5.3 Conclusion

The key finding from the usability evaluations is that COVE worked effectively in two important ways: for data exploration experts and novices, and for visualization producers and consumers. This is substantiated by activity logs, conversations, and user survey responses. Several capabilities in COVE were added or refined based on user feedback from the two studies.

Both experts and non-experts found COVE easy to use and rated its unique data exploration and visualization capabilities highly, although problems with performance limited some of their interactions. Participants who had used Google Earth for visualizing ocean data found COVE as easy to use and significantly better for exploring datasets. Users who had viewed datasets in sophisticated ocean data tools found COVE's interactive tools significantly easier to use and more useful overall.

Visualization producers were enthusiastic about COVE compared to existing systems, because they could create entire lessons that consumers found more understandable and engaging. Interactive views let them quickly iterate over visualization approaches and create

several options for comparison. Consumers found the views effective for carrying out data exploration tasks and the view panel intuitive and easy to use.

These findings, and the remaining unexamined areas of COVE, are further investigated in three real-world ocean science deployments that involved a broad range of science users interacting with COVE. This is discussed in detail in the next chapter.

CHAPTER 6

COVE IN THE FIELD

Observing the use of systems in real world activities is vital for determining effective design in science environments. To achieve this, I evaluated COVE in three different field deployments. I first observed its use on land in the design of the core infrastructure of the RSN ocean observatory over several months. This was followed by a program of further tests at sea, where I joined two dissimilar, two-week expeditions with multidisciplinary science teams. The first expedition evaluated two primary ocean observatory sites, and the second carried out manned submersible missions to study underwater geothermal vents.

This chapter describes the environment and evaluation methodology used in each deployment, followed by the results broken into three categories: (1) observations that validate COVE's design, (2) design modifications that were subsequently incorporated from feedback, and (3) open challenges for COVE for this type of deployment in the future. The chapter presents research also available in the proceedings of the 2010 IEEE Conference on e-Science [35].

6.1 Observatory Design

Designing an observatory requires several tasks. Apart from viewing observed and simulated data to understand the processes and possible issues at each experiment location, different options must be explored and analyzed to determine optimal layouts for data collection within budgetary and technical constraints. The reasoning for alternative layouts must be presented and discussed with science users to evaluate the effectiveness of various designs. After creating the initial infrastructure, these tasks must be carried out regularly over the life of the system as science teams extend the observatory with new instruments and sensors.

Prior to COVE, the design process used by the RSN team was cumbersome; it included the use of paper and digital maps, geographical surveys, and multiple software applications. When new layouts were created, they were recorded in word processing or spreadsheet documents, with visualizations created offline by a graphics team for presentation. Changes were time-consuming and expensive: there was no automated way to update costing or cabling, comparisons of different models required several documents and spreadsheets, and it was hard to examine designs from different angles or scales.

For this deployment, I employed a long-term embedded approach using participatory design techniques. The main participants were the RSN's lead scientists, engineers, and graphics staff involved in the daily design process. Based on observations of participant interaction with COVE and their feedback, system updates were created and disseminated on a weekly basis. I observed the participants over several months in three primary tasks:

- Individual layout of cables, nodes, and instruments
- Group discussions concerning layout options
- Creation and use of visuals for internal and external presentations

6.1.1 Design Validation

I observed many interactions highlighting the value of COVE in this environment. First, the staff could individually make immediate design modifications based on new needs and new data or bathymetry. Scientists created new layouts in minutes rather than hours or days. They saved multiple views to evaluate results from various angles, particularly between 2D top-down and 3D perspective views. The scientists found the layering model easy to use, and maps of fault lines and surface geology were easily displayed together to facilitate interactive cable layout. The heads-up display was particularly well received for tracking budgets and determining the timing of cable and instrument deployment. One participant explained how COVE improved her design work. "*For me, since I don't have the tools that our graphic artists have, it meant that I can respond much quicker. There was a period of several months where we were getting crisis level, 'you need to respond to this in the next*

Figure 6.1: Scientists using COVE to iterate instrument layouts in real-time and discuss potential designs for a site.

month' to deal with contingencies or inflation or budget changes, and COVE made that possible."

Second, interactive sessions with COVE allowed groups to iterate designs in real-time and consider several alternative designs at the same site (Figure 6.1). After COVE was adopted across the team, it became one of the primary collaboration tools used to create the RSN primary infrastructure. "*As soon as we got access to COVE and had some useable knowledge, from that time on we stopped using anything else. It really was a transformation. From that day on it became the base map for what we do.*"

COVE was used extensively to prepare for NSF design reviews where the team was required to present and defend its design to funding agencies. It allowed different core cabling alternatives to be explored quickly and helped convince the team of the necessity to significantly change the cabling configuration from ring-based to star-based. It was the key tool used to create visuals to explain layouts at design reviews because it quickly created production-level visuals in sync with the most recent designs. The NSF reviews were

considered highly successful, and the RSN team members were particularly vocal in praising COVE as a key contributor to this achievement.

6.1.2 Design Modifications Based on Feedback

I also learned much from the team to improve the initial COVE design. For example, I learned that it was necessary to **support existing practices** for the scientists to comfortably transition to COVE. Although paper maps and rulers had many drawbacks, they were familiar and dependable tools. I therefore added simple 2D top-down views and map views that could be printed, and more extensive import and export capabilities were necessary to support Google Earth and other tools already employed by external groups.

It was also necessary to **provide ways to double check results** within COVE to ensure that distances and positions were accurate before being forwarded to team members. Overlay grids and measuring tools were added to interactively reconfirm distances and locations, as were new map projections to allow direct measuring of distances and comparison with a wider range of maps. Import and export capabilities were also enhanced to allow comparison of positions with known landmarks.

The early COVE design under-estimated the **importance of presentation support**. I initially considered presentation-quality visuals a minor aspect of COVE, but found that because the RSN team had to present and defend its work regularly, high-resolution output formats, text and image overlays and visual highlights were important additions to the collaborative environment. Furthermore, when showing work in COVE to others, several simple presentation features were greatly appreciated: the ability to easily generate multiple views of a site, create an automated slideshow, fade between views, and even simply rotate the current view slowly.

6.1.3 Open Challenges

One disappointing outcome in this deployment was the **limited use of task-specific instrument libraries**, which was due to a combination of factors. First, the COVE implementation was unstable across this capability for part of the deployment. Second, the

observatory design process I observed was in an early stage, which focused primarily on a single core infrastructure across the team, and thus an assortment of instrument and cabling sets was not necessary at the time. As the observatory moves to the next levels of design internally, and is used externally for specific research projects, this capability may be adopted more broadly.

Finally, I originally focused system design efforts on interactivity and rich visualization, spending less time on **data management** aspects of the observatory. Although the visual capabilities of COVE were quickly accepted and widely appreciated, its file-based data access solution limited use by engineering. This was due to the high level of detail they ultimately desired in order to track and manage all observatory assets. They were hopeful COVE could provide a complete solution to this problem, but at the time their lack of specific observatory database requirements made it infeasible to design and implement an adequate solution. In the future, adding database integration to the COVE design would be relatively simple after a more clearly defined data management model is established.

6.2 Mapping RSN Sites

After working with the design team for several months and refining the COVE design based on its feedback, I had the opportunity to test COVE at sea. This next stage of the RSN design process was a two-week ocean expedition to collect extensive data from two major seafloor research sites for the observatory. Also, as the RSN observatory was not yet operational, it allowed evaluation of COVE in the context of real-world ocean research, which could provide a proxy for science teams using the observatory.

The chief goal of the ocean expedition was to verify that the multi-million dollar primary connection nodes were being placed at safe and durable locations. Mapping the seafloor was the most important part of this effort. For this task, sonar is used to collect 3D point sets by bouncing acoustic reflections off the seafloor to create a map of the terrain (Figure 6.2). The ship or AUV carrying the sonar travels back and forth over an area, covering new terrain on each pass. Because the beam scans cover a fixed-angle, the width and data resolution are determined by the depth. For example, the closer the sonar is to the bottom, the thinner the

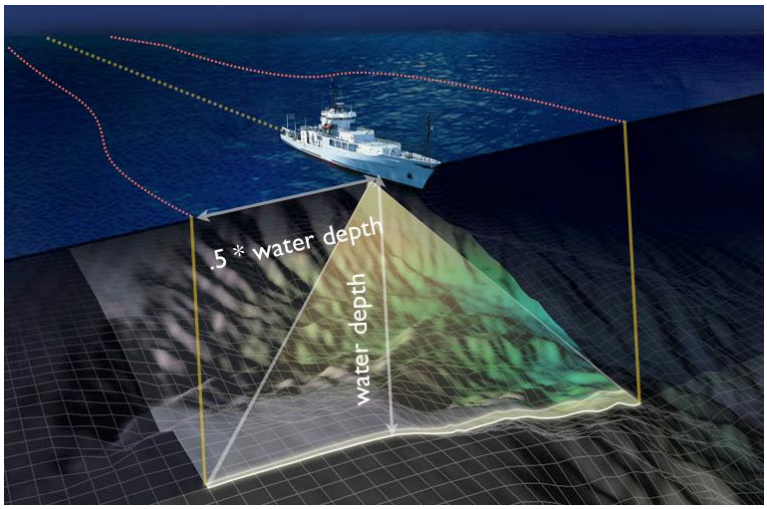

Figure 6.2: An illustration of EM 300 Side Scan Sonar from the research vessel used to gather bathymetric data.

area covered and the more detail collected. These detailed maps of the bottom determined if the node site was flat, solid, and free of landslide danger.

The cruise took place on a research vessel 100 meters in length, with 25 full-time crew and 32 research staff. Hull-mounted sonar was used for lower resolution mapping from the ocean surface. For high resolution mapping, the AUV Sentry sub [23] provided by Woods Hole Oceanographic Institution followed a programmed route 50 to 100 meters above the seafloor for up to 10 hours. The previous technique for determining ship routes involved the team's gathering around a map and placing markers (usually coins) to designate possible waypoints (turn locations) for the ship (Figure 6.3). The AUV was programmed by printing a high-resolution map of the location and using a ruler to determine lengths of runs. Although this approach had been used on many cruises to carry out missions, it suffered from the same drawbacks as those noted with instrument layout for the RSN: routes were hard to create and modify, it was hard to compare alternatives, and different views of the route were difficult to create.

The goal was to replace this approach by using COVE to plan the mapping route, track progress, and then view and analyze the final bathymetry. Because all observatory data was already available in COVE, planning missions in the context of prospective observatory

Figure 6.3: Previously, the scientists required several different data representations and tools to collaborate on decisions.

layouts was possible, and alternative layout plans could be created if necessary. The primary participants still included scientists, but there were also research assistants, ship's engineers, and students. Whereas the methodology in the preceding deployment was long-term, this evaluation was short-term and time-sensitive, and new prototypes had to be created at least daily. The task areas evaluated were similar to those in the previous deployment:

- Individual route planning based on goals from the chief scientist
- Group discussions to evaluate bathymetry and plan missions
- Communicating status and plans at daily science meetings

6.2.1 Design Validation

In the time-sensitive environment on the ship, COVE's interactive mechanisms for route layout were effective in many ways. As pointed out by one scientist, "*On the cruise where we used COVE for the first time, I would say that it increased efficiency several orders of magnitude for laying track lines. And it's much more iterative since it's so easy to move*

Figure 6.4: With COVE, the science staff could share a central forum that captured all needed data and processes, keeping everyone engaged in the discussion.

things around and see the swath width of the sonar." One clear measure of the increase in efficiency provided by COVE was that, by the end of the cruise, bathymetry mission planning was being pushed from experts down to students, as indicated in the usability studies. The scientists also noted that COVE improved data quality as well as efficiency, "*Since we fly over such steep topography, we'd often have holes in the data doing it by hand. With COVE we increased the effectiveness, the quality, and the communication between people.*"

Easy integration with existing bathymetry processing formats and tools made it simple to quickly assess collected bathymetry. Once in COVE, 3D bathymetry visualization tools were all used extensively to quickly explore collected terrain data and discuss site tradeoffs across the team. Because it was cheap to make new routes, several possible plans could be created, compared and discussed to determine effectiveness. It also allowed the team to take advantage of unexpected opportunities, "*The nice part was that because of bad weather or instruments breaking, the normal things that go on out there, sometimes we'd have two more hours of ship time, so we could now use COVE to plan out a track line to fill in a hole and*

use the ship efficiently. We could have done it before by hand, but the ability to respond in a timely manner - I can't tell you how much easier it was."

COVE's interactive visualization interface was used by all levels of the science staff, from students to the chief scientist (Figure 6.4). At the daily science meeting, COVE became a forum to showcase collected bathymetry and other datasets from the expedition. The interactive views and slideshow mechanism allowed a wide range of users to quickly become presenters and story-tellers for their aspect of the trip. I saw great interest not only from the science staff, but also from the ship's engineering staff. By the end of the cruise, the general expectation was that COVE would become part of future mapping expeditions. This is particularly impressive considering the expedition's budgetary parameters and constraints: the research vessel costs $25K per day to operate, and a mission could take months to reschedule if unsuccessful.

6.2.2 Design Modifications Based on Feedback

I learned much in the transition from the land-based environment to an ocean expedition. I had little time to test COVE in this setting before I shipped out and soon noted several stability issues, requiring changes to **operate in a variable network environment**. I eventually had to use portable drives to share data and visualizations, which was acceptable on this occasion but needed to be resolved for future expeditions. To this end, the Web site supporting sharing for COVE has been redesigned to be more portable and provide services over the ship's intranet.

There was also a need to **accommodate an unstable physical environment**. The ship is uniquely designed to be very stable while at sites, but it still moves with the waves and when under power to new sites. To account for this, mouse input was modified to ignore jittery movement based on conditions. In addition, visual output provided extensive object resizing to avoid the necessity of users focusing on a detailed area for an extended period, which could otherwise lead to motion sickness.

An important insight arising from my work with issues of stability in a research environment is that the science staff considered that **software crashes are often forgivable,**

but data preservation is always crucial. Time at sea was limited, and staying on schedule demanded the completion of tasks at each site. If hours of work are lost on a sophisticated bathymetry collection route, for example, it may mean that the cruise is delayed while it is recreated. While this issue is highlighted in the time-sensitive cruise environment, it was also observed in the land-based deployment. As a result, COVE provided a number of features to prevent any such critical losses: more extensive user feedback on risky situations (e.g., low memory), automatic saving of the work environment, and automated testing facilities to validate new releases of COVE.

6.2.3 Open Challenges

A clear request voiced by the expedition's chief scientist was to extend COVE to function as a primary **resource scheduling and communication system for the entire expedition**. Currently, scheduling is carried out with a variety of tools and documents that do not provide a visual way of determining and communicating the daily plan. COVE's ship and AUV routing capabilities could facilitate the scheduling of other expedition events specific to locations on the cruise. Also, collected datasets, mission results and daily reports could be geolocated with the ships position to provide a rich, interactive way of displaying what has been accomplished daily at sea, as well as when reporting on the entire expedition when back in port.

6.3 Exploring Geothermal Vents with ALVIN

After the positive response from mapping observatory sites, I was invited to evaluate COVE in a second ocean-based science environment. This time, I accompanied a team of scientists on a two-week cruise to investigate geothermal vents in the northeast Pacific. This took place with a vessel and crew size similar to the previous mission, but was focused on twelve manned underwater missions in the ALVIN research submarine (Figure 6.5), launched from the ship to collect data on the seafloor [4]. The crew for each mission consisted of one pilot

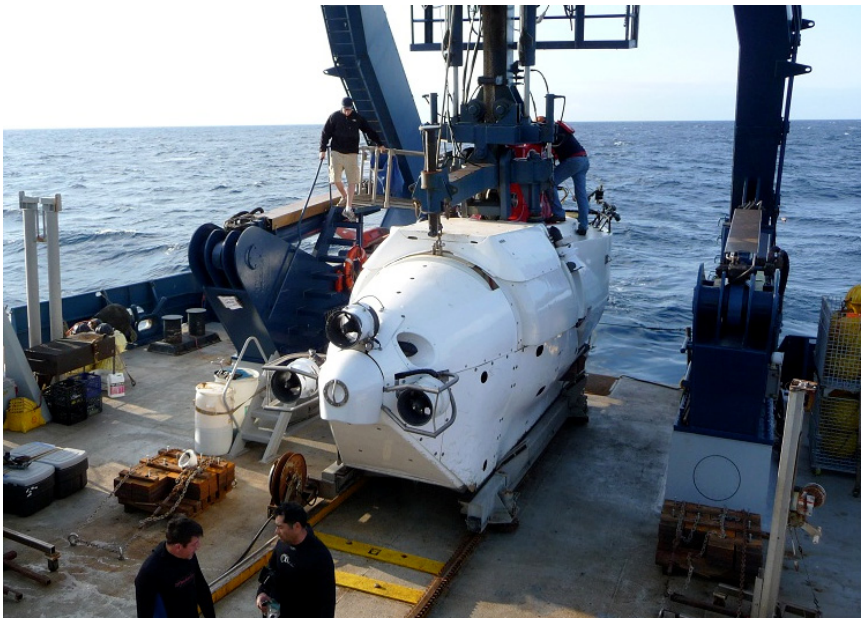

Figure 6.5: The 3 man ALVIN research submarine on the deck of the Expedition Vessel being prepared for mission deployment.

and two scientists and lasted 6-8 hours with 4-6 hours of bottom time based on the depth of the site.

Because it was difficult to communicate with the sub once it was on the bottom, careful planning before each mission was critical. This required collaboration among the scientists to determine the experiments to be carried out, what data was to be collected, and which samples were to be brought up for further study. After the science goals were agreed upon, the crew of the sub and the ALVIN support team met to map the goals against constraints and create the dive plan. This process involved a large collection of paper maps and the expertise of previous crew who had dived at the site.

On the seafloor, it was often hard to navigate due to lack of light and unique landmarks, and the surface team was often unable to provide help due to their limited ability to track the sub's position. It was not uncommon for the sub to lose up to an hour of its bottom time finding locations in the dive plan, and the sub crew would often need to modify the plan mid-dive. When the ship was not on site waiting for the sub to surface, the science crew deployed

other instruments and examined data to find new vent fields. All these tasks were documented at the end of the cruise for submission to the funding agencies.

As with the previous ocean expedition, this evaluation was short-term and time-sensitive, and new releases of COVE were often created daily to respond to user feedback. The primary participants included scientists, research assistants, and ship's engineers, as well as the ALVIN support crew and pilots. Due to the novel nature of this deployment and the limited time to modify COVE, the task areas evaluated varied from those in the previous expedition:

- Dive team planning of ALVIN missions
- Group discussions evaluating collected data
- Determining possible integration of COVE with ALVIN missions

6.3.1 Design Validation

Given the established routine on the ship for handling missions, integrating COVE into the process was incremental, which therefore had much greater impact on later missions. By the end of the cruise, COVE was being used to visually scout each site before the mission. This involved looking at high resolution bathymetry and previous dive tracks to determine new dive locations. Proposed dive tracks were laid out and compared against the sub's power supply, which allowed the dive tracks to be interactively changed to make trade-offs against the science plan. Different views of the track were available to help identify issues, and a set of key views were printed for collaboration with the pilot when on the bottom. "*Having the printed 3D views in ALVIN really helped out. We could compare them against our regular navigation screens and what we could see outside to adjust the dive plan on the fly.*" When the team returned, it could compare the actual dive track with the planned version and combine the track with photos, gradients and data visualizations to present to other scientists.

In addition to being used to plan ALVIN missions, COVE played a key role in investigating a possible new vent field. Anomalous heat readings were noted while steaming between sites, and COVE was used to quickly view the temperature data together with the

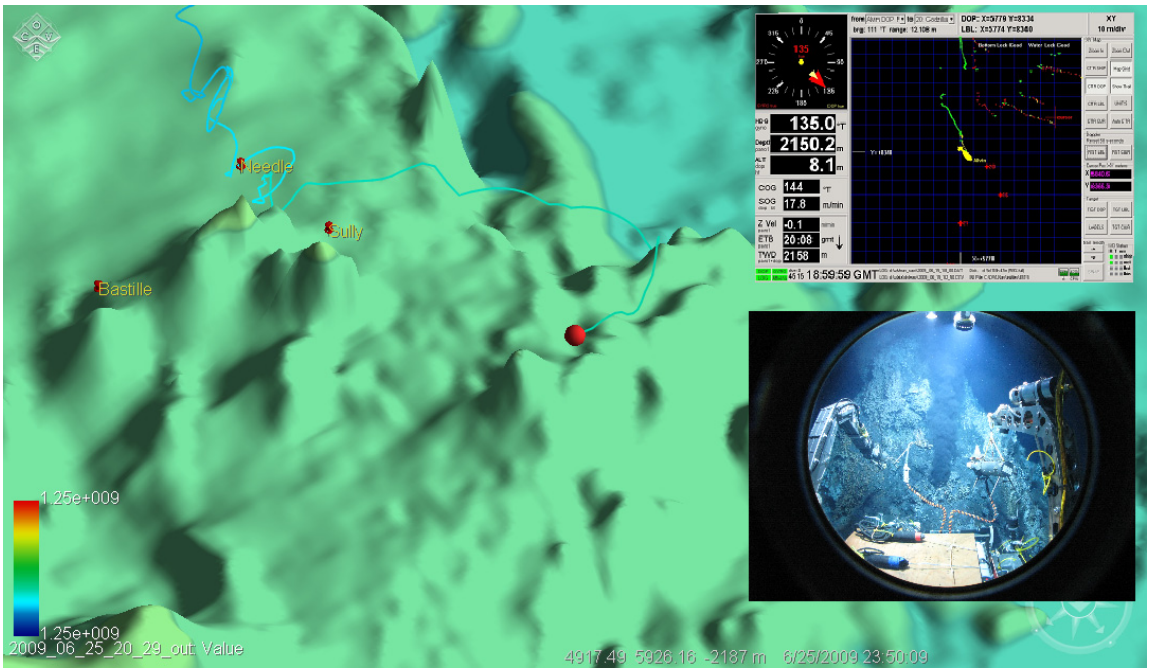

Figure 6.6: The track of the ALVIN sub in COVE, captured after providing a course correction during the mission. On the right are the original ship-side navigation screen and an external view.

area's bathymetry. The scientists were able to compare different views, zoom in to possible candidate locations on the seafloor, and then plan a series of new data collection tracks to test their hypothesis. One participant in the discussion noted, "*The fact that COVE helped us move from detection to a research plan so quickly was great. Everyone could just gather around the screen to talk about the data and figure out our plan.*" By the end of the cruise, we were unable to determine the new vent field's precise location but did amass extensive data for future expeditions.

Finally, I participated in one ALVIN dive to test the effectiveness of COVE's interactive 3D visuals in the submarine itself (Figure 6.7). To do so, COVE was connected to the real-time Doppler location feed from the sub, and input was combined with visualizations of high resolution terrain. This allowed the ALVIN's track to drive the viewer location in COVE and give a real-time view of underwater terrain for the area relative to the sub's location, a view never previously available to the ALVIN pilot. This proved to be highly valuable for

Figure 6.7: At the bottom of the ocean in ALVIN, with COVE providing real-time 3D bathymetry visualization.

the pilot in visualizing the seafloor and determining the best route to complete all mission goals. For example, at one point COVE illustrated that the sub course recommended by the surface team was 180 degrees off from the next station (Figure 6.6). The pilot later observed, "*Without COVE catching that problem, we could have easily wasted a big chunk of bottom time and would have probably had to cut things from the dive.*" As it was, ALVIN was back on track almost immediately, and we successfully achieved all our assigned objectives with time to spare.

6.3.2 Design Modifications Based on Feedback

Crucial to supporting navigation on my ALVIN mission was the addition of **support for real-time data input**. This was accomplished through a polling Web interface that captured data on a parallel thread, then filtered and stored the data for access by data management modules. This feature will also be important for ocean observatory support once operational, where visualization will be critical for detecting and monitoring real-time events, such as volcanoes or earthquakes.

Another important addition was **low power display updating** to extend battery life on the mission. Highly interactive interfaces, such as computer games, are designed to generate new images many times a second. This constant use of CPU and GPU is problematic in an environment with limited power. COVE was modified to track user and system events and generate screen updates only when absolutely necessary, thus minimizing power needs while maintaining an interactive interface.

As with ocean observatories, providing conventional representation is important to communicate across and outside the team. On the ship, one of the most common and trusted formats is a paper map, so **interactive 2D map views and output formats** were added. Perspective is important for providing clues about locations of objects in a 3D view, but also warps distances, so orthogonal projections were provided to remove distortion. Common map affordances – such as borders, legends, and tick marks – were also provided. This offered a comfortable way for users to integrate COVE with accepted practices.

6.3.3 Open Challenges

Based on feedback from the participants, there is a genuine desire to develop COVE into a **fulltime navigation system for ALVIN**. On the most basic level, this would require additional work integrating with the ship and sub systems. A more significant consideration is the need to reassess system architecture to ensure an extremely high degree of stability: ALVIN dives are situations where lives are at stake if systems fail or give incorrect information to the user. More time and close collaboration are required to fully understand the complete set of needs necessary for such a project.

6.4 Conclusion

For observatory design, COVE made instrument and cable layout simpler, faster, and easier to create in the context of collected data, as well as improving communication to a variety of audiences. For the RSN mapping expedition, COVE made ship and sub routes easier to create and provided a platform for discussing and evaluating collected data and node site

selection. And for the ALVIN expedition, COVE provided a platform for new discoveries and a major improvement in ALVIN's navigation. In each deployment, COVE proved to be a transformative and intuitive environment for individuals to explore shared data and conduct interactive tasks, a common space for groups to share discussions and capture important discussion points, and a familiar context for large teams to present progress and findings, both internally and to outside groups.

CHAPTER 7

SCALING TO THE CLOUD

The final evaluation analyzes COVE's architecture based on performance running a canonical set of data exploration and visualization workflows in a variety of configurations.

Virtual observatories are making science in every field more data-intensive, motivating the use of a variety of data management, analysis, and visualization platforms and applications. A comprehensive infrastructure addressing these processes requires cooperation between desktop Graphics Processing Units (GPUs) for immersive, interactive visualization, server-side data processing, and massive-scale cloud computing. The requirement that systems seamlessly span all three platforms, leveraging the benefits of each, is becoming increasingly common.

Moreover, it remains unclear whether application components can be statically assigned to these resources or if specific use cases motivate specific partitioning scenarios. For example, local processing can be an ideal solution for simplicity, to reduce latency, to reduce load on shared resources, and – in the era of pay-as-you-go computing – to reduce cost in real currency. However, as data is increasingly deposited in large shared resources, and as its size grows, reliance on local processing incurs significant transfer delays and may not be feasible.

There exists little research on architecture, principles, and systems for seamless visual data analytics. Current workflow systems are dataflow-oriented [9, 11, 22, 56, 103]; they do not subsume client-side interactive visualization applications such as Google Earth. Today's visualization systems [62, 74, 86] lack data integration capabilities to access and manipulate data from different sources.

However, COVE's architecture not only spans desktop, server-side, and cloud resources, but it allows work to be flexibly allocated across these resources. This chapter explores the design space for architectures that span client, server, and cloud for visual data analytics in the ocean sciences. Together with the COVE client and the Trident Scientific Workflow

Workbench [9], my test system includes the Microsoft Azure cloud computing platform [66]. I compare various design choices, specify tradeoffs in a simple model, and make recommendations for further research.

To inform the analysis, I defined nine visual data analytics scenarios derived from my collaboration with ocean scientists described previously. From these scenarios I distilled a set of common sub-tasks and then implemented a selection of the scenarios as representative visualization workflows in Trident and COVE. I model these visual data analytics workflows as instances of a Data-Workflow-Visualization-Client pipeline, described in the next section, and use this abstraction to derive a simple cost model based on data transfer costs and computation time. I then use this cost model to design experiments that test each workflow in a variety of multi-tier architecture scenarios using typical resources, measuring both computation and data transfer costs at each step.

I find that the role of the client remains critical in the era of cloud computing as a host for visualization, local caching, and local processing. Network bandwidth limitations found in practice frequently dominate the cost of data analytics, motivating the need for pre-fetching and aggressive caching to maintain interactive performance necessary for immersive visualization applications. I also confirm that a GPU is crucial for efficient visual data analytics, suggesting that the generic hardware configurations found in many cloud computing platforms do not provide a complete solution. Finally, I show that there is no one-size-fits-all partitioning of components that satisfies all cases and conclude that seamless Client + Cloud architectures – as opposed to Cloud-Alone or Client-Alone – are an important consideration for visual data analytics applications. These results are also presented in the proceedings of the 2010 International Conference on Scientific and Statistical Database Management [34].

7.1 Architecture Overview

As described in Chapter 2, computer visualization researchers have articulated a standard data visualization pipeline architecture around which the majority of today's visualization systems are designed [1]. This architecture consists of three logical steps: *Filter* (selection,

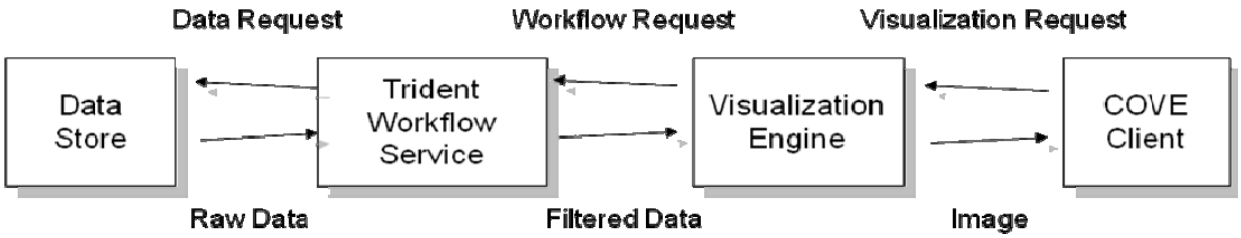

Figure 7.1: The COVE architecture consists of a set of components that can be distributed across local, server and cloud resources to support data exploration and visualization needs.

extraction, and enrichment of data), *Map* (production of a spatial representation of the data using visualization algorithms), and *Render* (generation of a series of images from the spatial representation). Today, the Map and Render steps are typically performed together using high-performance GPUs; going forward, I simply refer to these two steps together as *Visualization*.

Informed by the basic Data, Filter, Map, Render visualization pipeline, COVE's architecture is modeled with four separate software components: *Data Store*, *Workflow*, *Visualization*, and *Client* arranged linearly according to dataflow as shown in Figure 7.1. Each component can be flexibly distributed across physical systems, with communication provided through system I/O services if the components are co-located or by Web application interfaces when distributed. The *Data Store* component can be a remote query service, such as a database on an OPeNDAP server [23]. These services are typically outside the scope of a workflow system, though calls may be issued from a workflow context. The *Visualization* component is distinct from the *Workflow* for two reasons: (1) there are a variety of stand-alone visualization systems found in practice [62, 74, 86], (2) the visualization step occurs last and benefits from access to a GPU, and is therefore measured independently from the rest of the workflow. The *Client* component processes user interaction and issues calls to the upstream pipeline. This model lets the Visualization component seamlessly move from platform to platform based on current graphic processing needs; it can be tightly bound with the client for interactivity, tightly bound with the workflow system to minimize data transfer, or run independently to leverage external graphics resources.

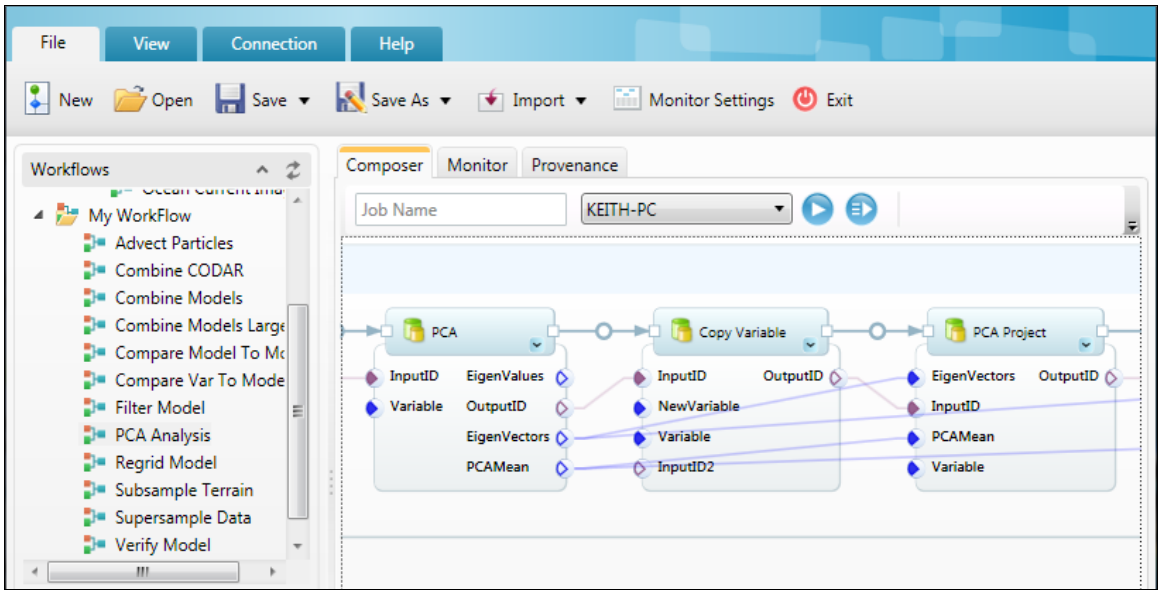

Figure 7.2: An example of Trident's workflow editing interface, which allows the creation of sophisticated workflows by connecting pre-defined components.

To measure system performance across the visual data analytics workflows, I leverage COVE's Visualization and Client components, Microsoft's Trident workflow system for Workflow, and Microsoft's Windows Azure cloud computing service for the Data Store. These platforms, along with the architecture's data model and programming model, are now described in more detail.

7.1.1 Workflow with Trident

The Trident Workflow system, developed by Microsoft Research in concert with COVE, is a domain-independent workbench for scientific workflow that is based on Microsoft's Windows Workflow Foundation. The system supports a high level, component-based view of scientific tasks that offers many advantages over traditional, script-based approaches, including visual programming, improved reusability, provenance, and execution in heterogeneous environments. Unlike many workflow systems, Trident provides automated provenance capture, smart re-execution of different versions of workflow instances, on-the-

fly parameter modification, task monitoring, and support for fault tolerance and failure recovery.

Trident can be executed on the local desktop, on a server, or on a High Performance Computing (HPC) cluster. It currently runs on the Windows OS using the .NET API, using SQL Server for data storage and provenance capture. Interactive workflow editing and management is available through programs in the Trident suite (Figure 7.2). Trident provides cross-platform support using Silverlight, a cross browser and cross platform plug-in for delivering .NET-based applications over the Web.

Cross platform support is also available through a Web service interface developed in collaboration with COVE. This interface allows execution and job control through a REST API. For example, a user can login, select a desired workflow, monitor its progress, poll for created data products, and retrieve data products for local use via straightforward HTTP GET and POST calls. COVE used this communication interface to provide cross-platform access to Trident.

7.1.2 Cloud Services with Azure

Azure is Microsoft's cloud computing platform. In contrast to Amazon's suite of *Infrastructure as a Service* offerings (c.f., EC2, S3), Azure is designed to be a *Platform as a Service* that provides developers with on-demand compute and storage for Web applications running on Microsoft datacenters. Azure's primary goal is to provide a platform on which developers can easily implement *Software as a Service* applications. Amazon's EC2, in contrast, provides a host for virtual machines, but the developer must assume entire responsibility for outfitting the virtual machine with all necessary software.

Windows Azure has three components: (1) a Compute service that runs applications, (2) a Storage service for data management, and (3) an Application Fabric that supports the Compute and Storage services. To use the Compute service, a developer creates a Windows application consisting of *Web Roles* and *Worker Roles* using the .NET API or the Win32 API. A Web Role package responds to user requests, and a Worker Role, often initiated by a Web Role, runs in the Azure Application Fabric to carry out computation.

For persistent storage, Windows Azure provides three storage options: *Blobs*, *Tables*, and *Queues*, all accessed via a REST API. A Blob, similar to a file-like data object, can be retrieved, in its entirety, by name; a Table is a scalable key-value data store, and a Queue simplifies asynchronous communication between Worker Roles. In this investigation, I modeled data sources as Blobs and simply retrieved them for processing by the workflow engine.

7.1.3 COVE's Data Model

COVE's data model is file-oriented. On Azure, each file was stored as a Blob. On the server and local platforms, each file was stored on local disk and referenced with standard filename conventions. In either case, COVE accessed non-local files using HTTP.

The workflow component downloaded files from the data store and cached them. It then accessed files from the local cache. This transfer mechanism could have been optimized to reduce the overhead of local disk IO, but local storage also lets cached files be reused in future workflows. Further, I observe in the Results (Section 7.4) that local IO overhead was small relative to overall workflow cost.

Similarly, the workflow component made locally cached data products available through HTTP using a REST API. Although Trident provided access to a SQL server for data storage, I found the current implementation for serializing and de-serializing large files to the database to be prohibitively slow. Instead, I implemented an alternative multi-threaded, file-based data storage solution that resolved this IO performance issue. All experiments were conducted using this file-based solution.

Trident, by default, loads all data into memory to permit immediate access for computation. Thus large files can exceed physical memory, which can lead to thrashing of the hard drive. For the Trident workflow activities, COVE instead used a lazy loading strategy that loads only a descriptive header when opening a file; it then read in specific sections of the file on demand. This technique reduced the memory footprint and prevented thrashing.

7.1.4 Programming Model

The Trident activities were written in C++ and deployed to a dynamic library for increased performance. The object interface for the library was then wrapped by a set of .NET managed C# Trident activities. Each activity typically accessed a single method in the library. This design also made it easy to expose the same functionality available to the Workflow component to other systems. For example, because Trident was not yet available natively on the Windows Azure service, I created a substitute workflow shell on Azure that executed the workflow activities in each scenario.

Each activity contained a set of inputs and outputs that could be declared explicitly by the user or implicitly through composition with another activity (e.g., the output of the file load activity could connect to the input of the resample activity). The activities could be linked interactively using the Trident Workflow Composer or by editing the workflow specification file. Although the activities could execute either serially or in parallel (according to instructions in the workflow specification), all workflows in my tests operated serially to simplify performance monitoring.

7.2 Experiment Design

To measure the performance of different visualization architectural configurations, a representative suite of real ocean data visualization and analysis scenarios must be obtained. This was generated from data collected during the contextual design study (described in Chapter 3). The final product was a set of 16 data analysis scenarios that spanned a wide range of requirements in oceanographic data visualization and analysis. These scenarios are listed in Appendix C along with collected requirements for data transformation, analysis, and representation. To make the investigation more tractable, I focused on 9 scenarios that had relatively similar workflow needs based on my analysis and discussions with the scientists (see Table 7.1).

Table 7.1: Use case scenarios for visual data analytics in oceanography

Scenario	Description
1. Data Archive Analysis	Analyze existing collections of observed and simulated data
2. Ocean Modeling	Generate more accurate and denser ocean simulations
3. Observatory Simulation	Simulate ocean observatory collected data from existing data
4. PCA Sensor Placement	Determine optimal sensor placement using PCA modeling
5. Hydrographic Analysis	Estimate larger ocean effects based on limited observed data
6. Data Comparison	Compare observed and simulated datasets for integrity
7. Flow Field Analysis	Measure changes over time based on ocean currents
8. Hydrographic Fluxes	Measure changes over time in a specific ocean volume
9. Seafloor Mapping	Generate detailed terrain maps from collected sensor points

From the study, I determined that the scenarios shared several similar underlying tasks but were difficult to categorize as data-intensive, computation-intensive, or visualization-intensive. This difficulty was due in part to the heterogeneous nature of oceanographic data. While ocean simulations are data-intensive, producing multiple terabytes of output, most observed data is significantly smaller because it is expensive to obtain, usually sparse, and requires aggressive extrapolation and interpolation to determine ocean effects. Therefore, data sizes in any one scenario often show extreme variability from task to task. I also found that while large datasets increase computation time, as expected, analytics were not usually inherently compute-bound in these scenarios. Visualization needs varied from simple 2D plots to animations of geographically located datasets with multiple 3D iso-surfaces of ocean parameters. The choice of visualization primarily depended on a current research need rather than on a specific scenario, making it difficult to build an optimized application for all situations.

7.2.1 Concrete Workflows

From this suite of 9 scenarios, I derived 43 re-usable components (called *activities*) in order to recreate the scenarios. These activities were linked together to filter the raw datasets and create visualization-ready data products. Supported activities included: sub-sampling, super-sampling, cropping, filtering, masking, scaling, merging, and re-sampling data to match other data sample points. Some activities were not compute-intensive, while others, such as re-sampling of simulations, were quite compute-intensive due to the use of irregular grids in ocean simulation and the size of the simulated datasets. I also derived ocean-science-specific activities, such as *particle advection* to map currents, and the projection of instrument collected data onto *vertical sections* by super-sampling data points. Details of the complete activity set and usage appear in the standard oceanographic library included with Microsoft's Trident Workflow system [9].

Table 7.2: Representative workflows tested based on ocean science scenarios.

Workflow	**Scenarios**	**Data**	**Computation**	**Visualization**
Advect Particles	1,2,3,6,7	Medium	Medium	High
Combine Data	1,4,5,9	Low	Low	Low
Combine Models	1,2,3,6	High	Low	Medium
Compare Models	1,2,6	High	High	High
Compare Data to Model	1,2,6,7	Medium	Medium	Low
Filter Model	1,2,8	Medium	Low	Medium
PCA Projection	2,4	Medium	High	Medium
Regrid Model	1,2,7,8	Medium	Medium	Medium
Subsample Terrain	3,9	Medium	Low	High
Supersample Data	1,3,5	Medium	Low	Medium
Verify Model	2,6	High	Low	Medium
Vertical Section	1,5,6	Low	Low	High

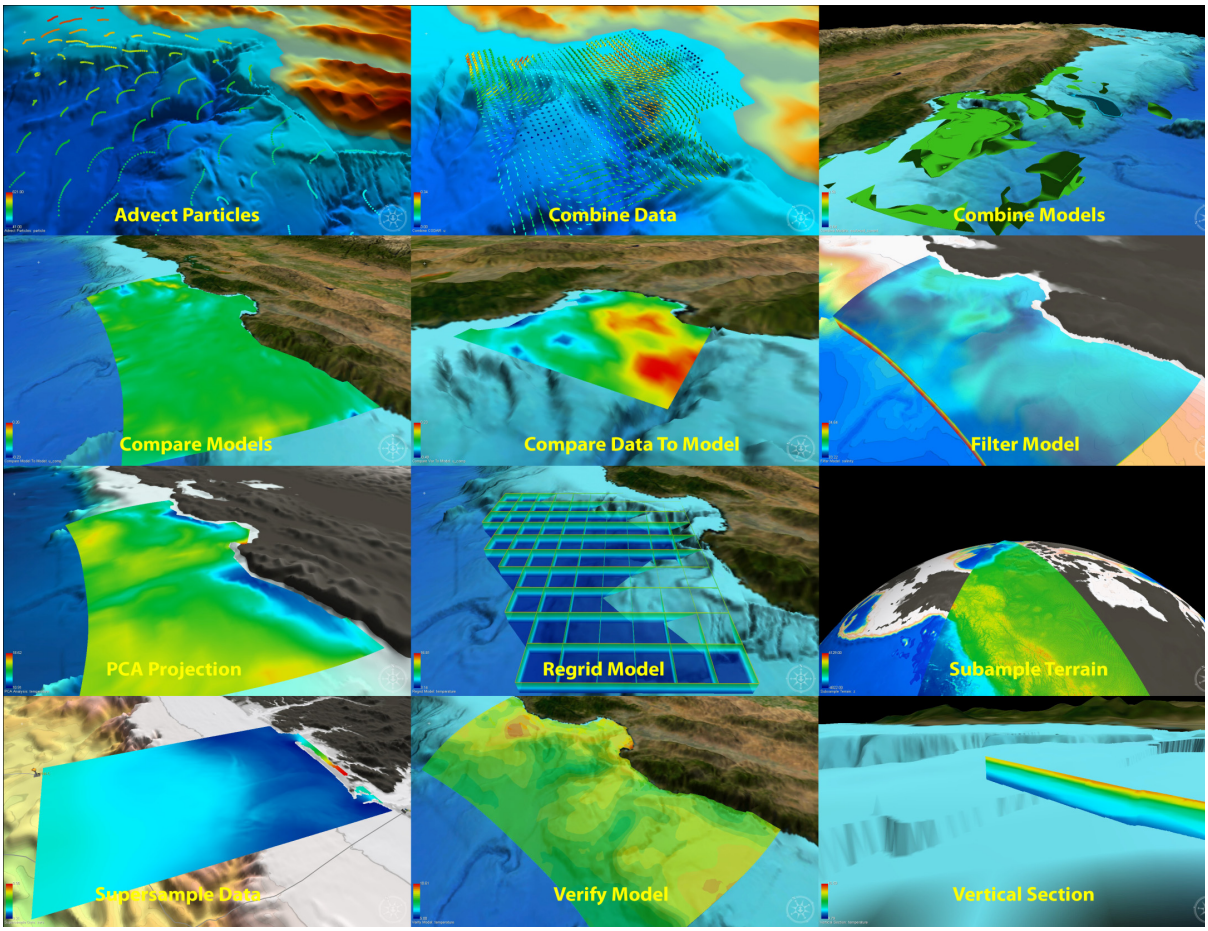

Figure 7.3: Visualization task output from the twelve representative workflows.

From these activities, I created a subset of 12 representative visualization-based workflows. These workflows are listed in Table 7.2, along with the primary scenarios to which they apply and a broad measure of how data-intensive, computation-intensive and visualization-intensive each was relative to other workflows in the sample. Each workflow consisted of 8 to 20 activities and formed the test cases for the visualization architecture described in the next section. The Workflow component loaded necessary inputs from the data store, transformed the input datasets into a new dataset, and then sent the data to COVE's visualization engine to create a time series visualization. The visual output of these workflows is shown in Figure 7.3.

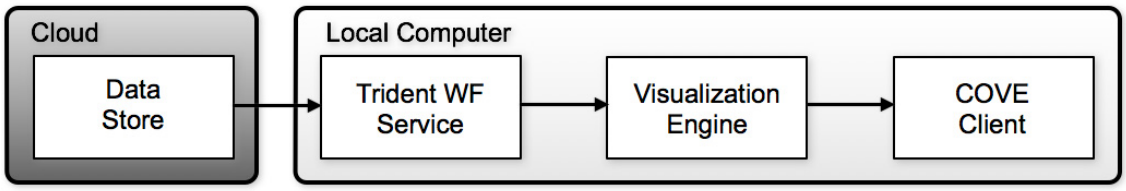

Figure 7.4: An example of a mapping of software components to resources. In this case the data is provisioned in the cloud, and all other tasks are performed on a local computer.

This analysis suggests that there are no obvious or simple patterns in the workload of oceanographic analytics and therefore no obvious or simple systems that can be built to satisfy work requirements. However, COVE's architecture offers a flexible visual data analytics platform that spans these requirements, incorporating workflow, visualization, and cloud-based data access.

7.2.2 Cost Model

I mapped the four logical components of the COVE architecture onto a physical configuration that consisted of three resources: *Cloud*, *Server*, and *Client*. An *architecture configuration*, or simply *configuration*, maps the software components (Data Store, Workflow, Visualization, Client) to hardware (Local, Server, Cloud) in a way that respects data flow order. For example, Figure 7.4 illustrates a configuration that maps the Data Store to the Cloud and the Trident Workflow Service, Visualization Engine, and COVE Client to the local computer.

The cost of each scenario is expressed as the sum of the workflow execution time, the visualization execution time, and the total data transfer cost between each pair of adjacent steps. That is:

$$COST = RAW\_TX + WF\_COMP + WF\_TX + VIS\_COMP + VIS\_TX \quad (7.1)$$

where:

$$RAW\_TX = RAW\_SIZE / BANDWIDTH\_DATA\_WF$$

WF_COMP = *WF_WORK (RAWSIZE) / PROCESSOR_WF*

WF_TX = *WF_SIZE (RAWSIZE) / BANDWIDTH_WF_VIZ*

VIS_COMP = *VIZ_WORK (WF_SIZE) / PROCESSOR_VIZ*

VIS_TX = *VIZ_SIZE (WF_SIZE) / BANDWIDTH_VIZ_CLIENT*

RAW_SIZE is the size in bytes of the input dataset. *BANDWIDTH_DATA_WF*, *BANDWIDTH_WF_VIZ*, and *BANDWIDTH_VIZ_CLIENT* are the transmission rates between the data source/workflow system, the workflow system/visualization system, and the visualization system/client, respectively. *WF_SIZE* and *VIZ_SIZE* are functions of the final output size based on the input data size for the pipeline step. *PROCESSOR_WF* and *PROCESSOR_VIZ* are the processor speeds for the respective machines, accounting for the potentially significant difference between server and client machines. *WF_WORK* and *VIZ_WORK* are functions of data size and return the (approximate) number of instructions required to process their input. These functions can be estimated precisely through curve fitting and sampling, or that can be provided by the user directly [16]. They are typically polynomial in the size of the input data, but I found that even rough linear estimates of the workflows often provided a reasonable estimate.

Although this model captured the cost of the pipeline, it was often not directly useful for prediction or optimization because the *WF_WORK* and *VIZ_WORK* functions were too difficult to estimate *a priori*. Therefore, I retained this model as a reasoning tool in the Results (Section 7.4) but also experiment with a simpler proxy model based *only* on data transfer overhead. This proxy model, although simple, frequently captured the relative cost between different architecture configurations. In this case, the model is:

$$COST = RAW\_TX + WF\_TX + VIS\_TX \quad (7.2)$$

The results (Section 7.4) present experiments that justify this simplification for certain configurations.

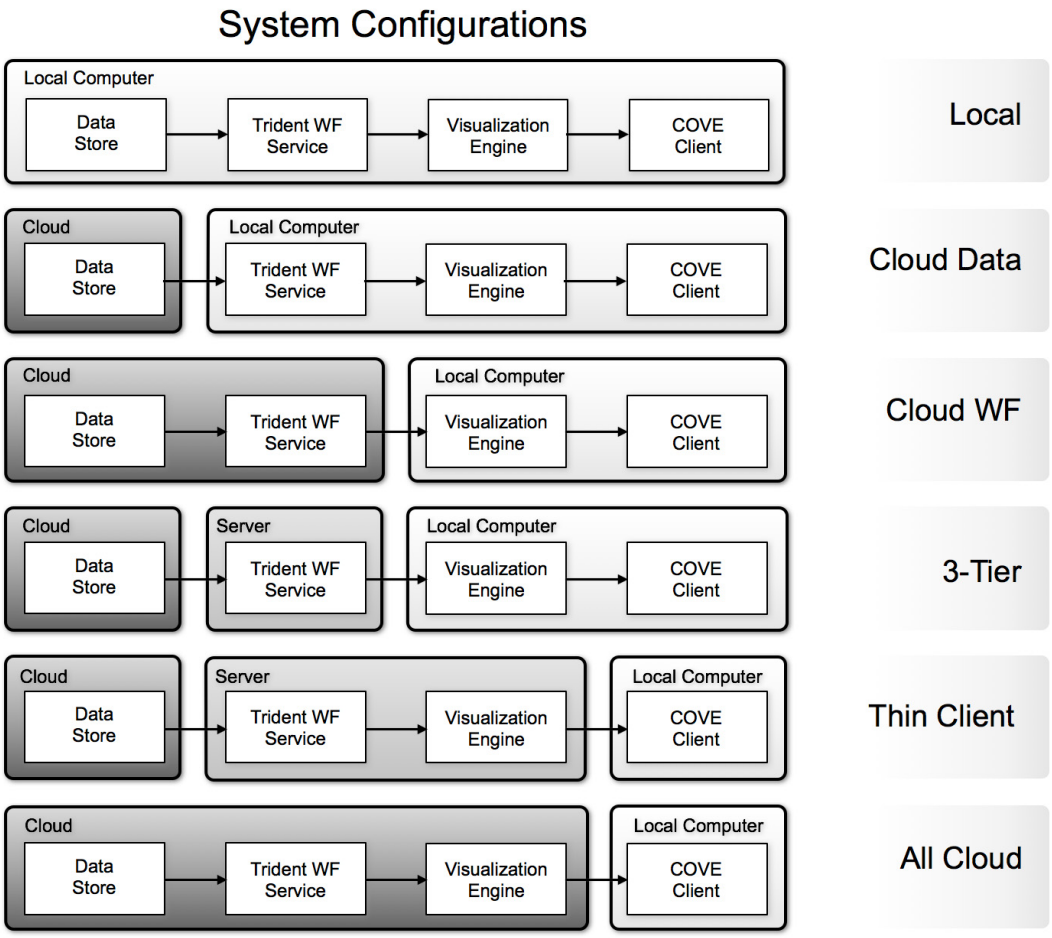

Figure 7.5: The six configurations evaluated in the integrated COVE, Trident, Azure system.

7.2.3 System Configurations

A variety of configurations can be created to run the visualization workflows enumerated in Table 7.2. Figure 7.5 illustrates the six configurations I evaluated.

In the *Local* configuration, all data, workflow, and visualization were handled locally. This was the most common visualization mode scientists used because it avoids network latency and cross-platform communication issues. However, the computation is limited by available cores and data size by local storage capacity.

In *Cloud Data,* data was moved to the cloud. This configuration, which allows larger data sizes and data sharing among researchers, imposes the overhead of downloading all necessary data to the local system for computation.

In *Cloud Workflow,* workflow was moved to the cloud and co-located with the data. This leverages the computational and storage capabilities of the remote platform and removes the overhead of moving raw data. However, this configuration still incurs the cost of downloading the filtered data or visualization.

In *3-Tier,* workflow was moved remotely to a server and data moved to cloud storage. This allows the most flexibility to optimize the choice of physical platform based on cost and needs. It is also the most sensitive to network speeds because raw and filtered data are both transferred across the network.

3-Tier Thin moved the workflow and visualization handling to a server, possibly with fast graphics capability, and placed data in cloud storage. This configuration is useful for a thin client environment, such as a browser or phone interface, but requires a fast connection between the cloud and the server.

Finally, the *All Cloud* configuration moved all data, workflow, and visualization to the cloud, creating a minimum of network overhead because only the final visual product is transferred over the network. However, the cloud environment is usually unspecialized; in particular, it typically does not provide graphics hardware for fast visualization.

7.3 Experimental Analysis

I executed the 12 benchmark workflows (Table 7.2) on each of the 6 system configurations (Figure 7.5) to record the 5 cost components (Equation 7.1). Because the complete dataset is difficult to visualize effectively, I show a summary of overall performance in the results (Figure 7.8), which displays the average time of each cost component across the entire workflow set for each configuration.

Table 7.3: Physical specification of experimental systems.

Machine	Description
Local Machine	• Apple Macbook Pro Laptop running Windows 7 (32 bit) • Intel Core Duo T2600 CPU 2.16 GHz, 2GB RAM, • Radeon X1600, 256 MB memory • Internet Connection: 11.43 MB/Sec in, 5.78 MB/Sec out
Web Server	• HP PC running Windows Server 2008 R2 Enterprise • Intel Core Duo E6850 CPU @ 3.00 GHz, 4 GB RAM • Internet Connection: 94.58 MB/Sec in, 3.89 MB/Sec out
Azure Web Role	• Intel PC running Windows Server 2008 R2 Enterprise • Intel 1.5-1.7 GHz, 1.7 GB RAM, No Video System • Internet Connection: .85 MB/Sec in, 1-2 MB/Sec out

Setup: I instrumented COVE and all of the Trident activities to record wall clock time for each component in the cost model: network transmission (RAW_TX, WF_TX, VIZ_TX), workflow computation (WF_COMP), and visualization creation (VIZ_COMP). The three systems described in Table 7.3 were used as the Local Machine, Web Server, and Azure Web Role in the architecture configurations. For these experiments, only the Local Machine included graphics hardware.

Data sizes: The data sizes used for each workflow appear in Figure 7.6, averaging around 150MB per task. Typical datasets include time steps of an ocean simulation, a set of *glider* tracks from an Autonomous Underwater Vehicle (AUV), or a terrain model for a region. All datasets pertain to the Pacific Northwest or Monterey Bay region and are actively used by scientists in ongoing research. Data size variance across workflows is representative of the breadth of data used by oceanographers.

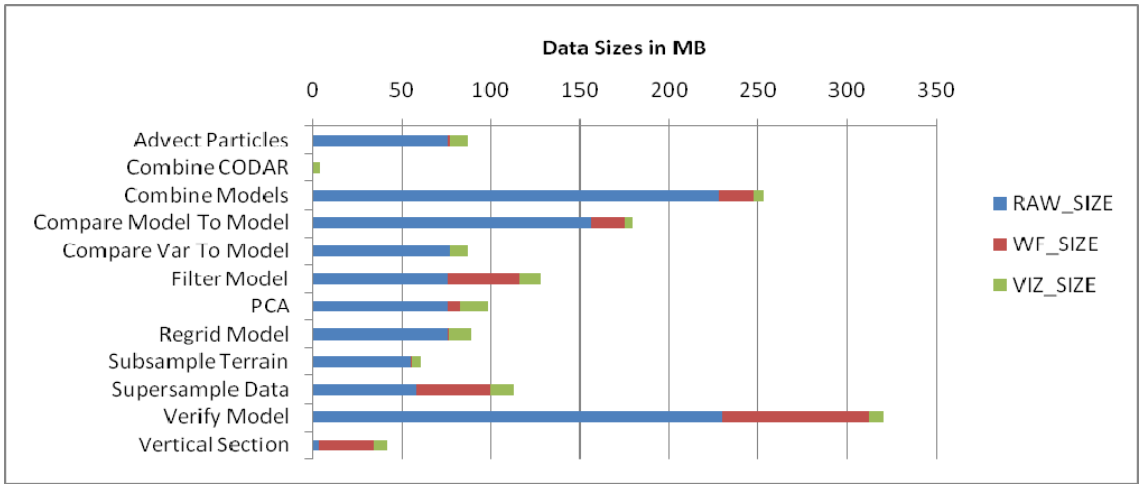

Figure 7.6: Data sizes used in the experiments. Each bar is broken into the raw data size (RAW_SIZE), the filtered data size generated by the workflow (WF_SIZE), and the size of the final result generated by the visualization (VIZ_SIZE).

7.4 Results

Based on the experiments outlined above, I now answer the following questions. (1) Is there a single partitioning of work that is preferable for all visual analytics benchmarks? (2) What role does client-side processing have in cloud and server oriented analytics? (3) Does access to a GPU strongly affect performance for visual analytics workflows? (4) Does the simple cost model derived in Section 7.2 effectively capture performance?

The tests showed the following results. (1) There is no one-size-fits-all partitioning of work – the appropriate configuration depends on workflow characteristics (Figure 7.7). (2) Client-side processing is a crucial resource for performance, assuming data can be pre-staged locally to minimize transfer costs (Figure 7.8). (3) Access to a GPU strongly affects the performance of visual data analytics workflows, meaning that generic, virtualized cloud-based resources are not ideal (Figure 7.9). (4) The simple cost model is sufficient to capture the behavior of these workflows, and the cost is generally dominated by data transfer overhead. I describe each finding below.

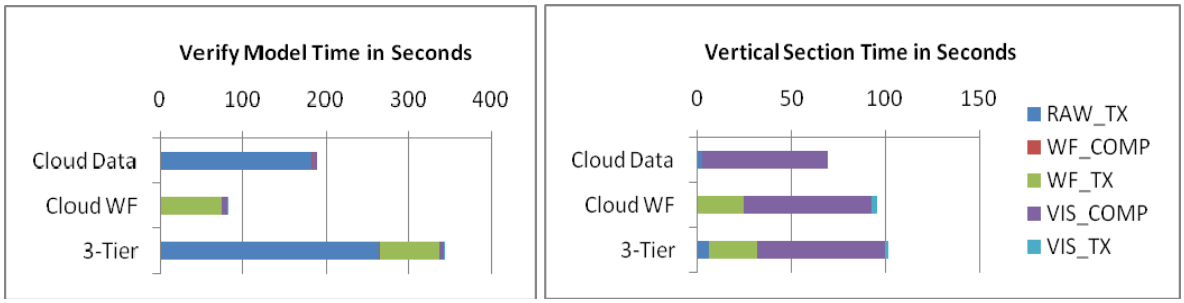

Figure 7.7: Time comparison of a workflow with WF_RATIO < 1 on the left and WF_RATIO > 1 on the right. When WF_RATIO < 1, the preferred strategy is to push the computation to the data and when WF_RATIO > 1, the better strategy is to bring the data to the computation.

7.4.1 There is No One-Size-Fits-All Partitioning of Work

The diversity of workflows in the visual analytics benchmark illustrates that multiple configurations must be supported in practice. Although local processing outperforms other configurations due to its lower data transfer overhead, this configuration is not always viable. Among the alternatives, no single configuration is best in all cases. In the Vertical Section workflow, for example, the output of the filter step is larger than its input, motivating an architecture that pulls data down from remote locations before processing; contradicting the conventional wisdom that one should always push the computation to the data. In terms of the cost model, this distinction is captured by the ratio of data output to data input in the workflow, or WF_RATIO = WF_SIZE / RAW_SIZE.

In Figure 7.7, the time profile for two workflows is displayed: one with WF_RATIO < 1, and the other with WF_RATIO > 1. For WF_RATIO < 1, the preferred (non-local) configuration to minimize transfer overhead is *Cloud WF*, where the data is processed on the same machine on which it resides. However, when WF_RATIO > 1, the preferred configuration is *Cloud Data*, which transmits the data to the local computer for processing. The *3-Tier* configuration in these examples appears to be a poor option regardless of WF_RATIO, but asymmetric processing capabilities between server and client can make up the difference. For example, the PCA workflow is a highly compute-bound scenario and therefore benefits from server-side processing at the middle tier.

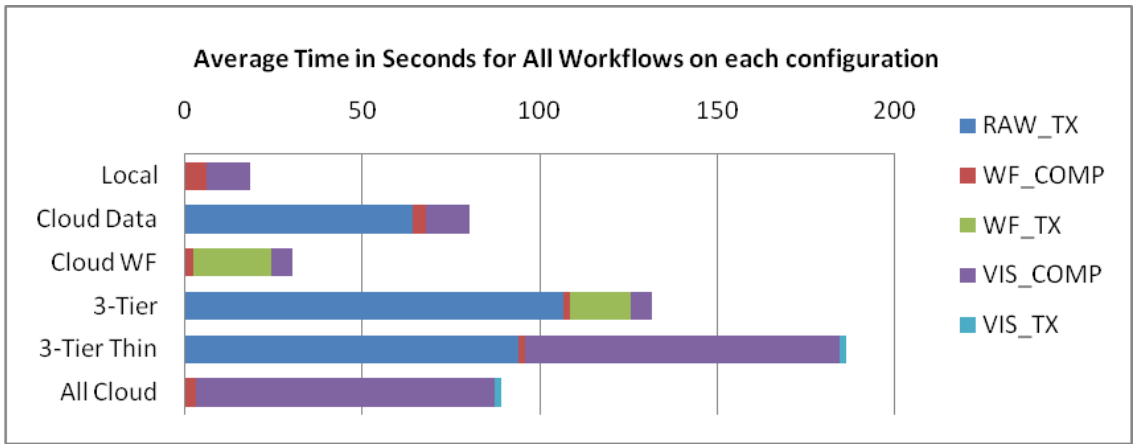

Figure 7.8: The average runtime of all 12 workflows for each of the 6 architecture configurations is dominated by data transfer overhead.

7.4.2 Client-side Processing Improves Efficiency

With these modest data sizes, the local data configuration performed well in all cases. Figure 7.8 shows the average performance across all benchmark workflows. The performance benefits are perhaps not surprising — desktop computers are increasingly powerful in terms of CPU speed and memory size, and they are typically equipped with GPUs to accelerate visualization. However, local processing is appropriate only for small datasets that are either private or have been pre-staged on the user's machine. Because the trend in ocean sciences (and, indeed, in all scientific fields) is toward establishing large shared data repositories, one solution is aggressive pre-fetching and caching on the users' local machines. Another possible solution, given that the *Cloud Data* configuration is the second most effective, is to consider migrating existing computation tools to the cloud.

7.4.3 Visual Analytics Benefit Significantly from GPU-Based Processing

To test whether access to a GPU would improve performance for visual data analytics tasks, the visualization engine was updated to run on Azure. In particular, in the *Thin Client* and *All Cloud* configurations, visualizations were created in software using the Mesa 3D library,

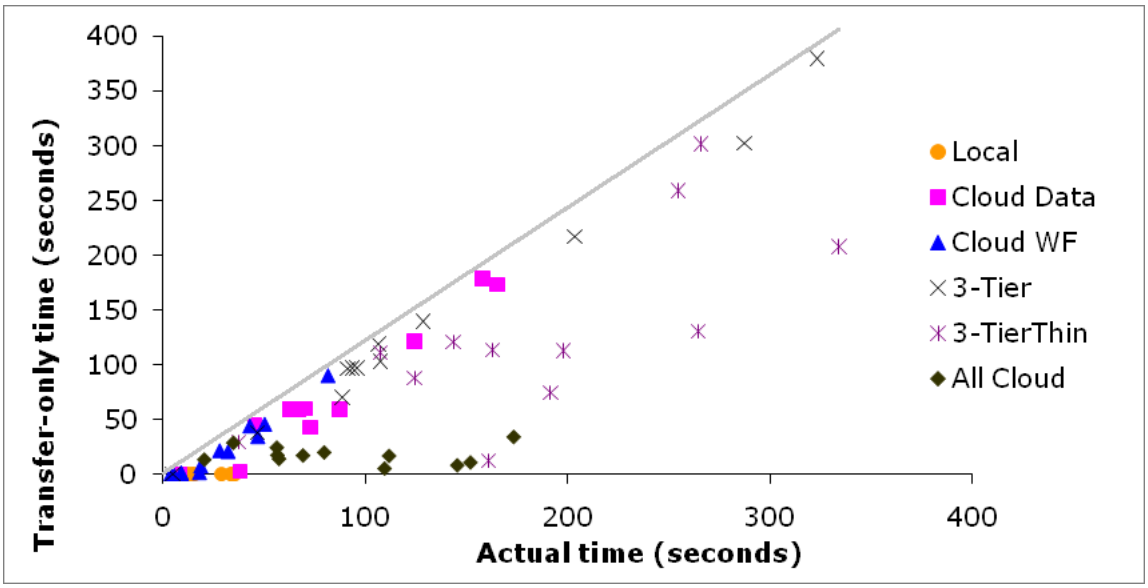

Figure 7.9: Scatter plot of results from Equation 7.2 compared to actual results. Points near the diagonal line indicate a strong relationship between the simplified cost equation and the complete version.

a state-of-the-art OpenGL software renderer, rather than the GPU. I found that on average, the workflow set ran 5 times faster overall with access to a GPU (Figure 7.8). The visualization portion of the work ran 9 times faster. This result suggests that the generic environment typically found on cloud computing platforms may be insufficient for visual data processing. After GPU capabilities are provided, *All Cloud* configurations for these visualization scenarios should be as effective as *Local* options based on the small VIS_TX cost visible on the graph. Another possible benefit of cloud GPUs will be increased computation performance for certain scenarios given their general vector processing capabilities.

7.4.4 A Simple Cost Model Informs Configuration Decisions

The proxy cost model presented in Equation 7.2 is simple, capturing only the data transfer costs between each step. Despite its simplicity, I find that this cost model adequately described several of the computations in configurations where datasets were moved between platforms, suggesting that it could be used to develop a basic architectural configuration

optimizer. In Figure 7.9, I plot the estimated running time against the actual measured times using a model that ignores everything except transfer times. Points plotted near the diagonal line indicate a strong relationship between the simplified cost equation and the complete version. *All Cloud* and *Local* configurations that transfer little or no data match poorly. The remaining configurations match fairly well, except for *3-Tier Thin*, which is dominated by visualization costs on the server that could be rectified with a GPU. The model also underestimates CPU and visualization-heavy workflows for these cases. Therefore, I am exploring an incrementally more sophisticated model based on parameters that can be estimated ad hoc by the scientists.

7.5 Conclusion

Today's cloud-based platforms must be augmented with significant local processing capabilities for maximum performance when processing representative oceanographic workloads. Due to the overhead of data transfer, access to GPUs for high-performance visualization, and the interactive nature of visual data analytics, Client + Cloud architectures maximize resource utilization and best improve performance.

These conclusions are based on my comprehensive, multi-year collaboration with ocean scientists from which I gleaned a suite of representative workflows, a complete visual ocean analytics system with immersive visualization capabilities in COVE and a flexible workflow environment in Trident, and a set of experiments testing each workflow on a variety of client, server, and cloud configurations.

CHAPTER 8

CONCLUSION AND FUTURE WORK

This dissertation has presented the motivation, design, implementation, and evaluation of COVE – a novel science tool for supporting data exploration, layout design, and multidisciplinary collaboration for ocean observatories. This process was carried out in close collaboration with scientists to ensure a robust collection of design requirements, iterative feedback throughout the design process, and evaluation in real-world scenarios.

8.1 Contributions

The list below reiterates the contributions claimed in Chapter 1 and summarizes how each was achieved.

8.1.1 A Gap Analysis Assessing the Capabilities of Existing Interactive Science Tools for Observatories

I first reviewed the domain of ocean data exploration, emphasizing ocean observatories. I then categorized data exploration and visualization tasks by type, conducted a survey of the relevant literature, and discussed applicable data exploration and visualization systems. Finally, based on two separate methods of evaluation, I presented a gap analysis of extant systems, illustrating the need for a new science tool that combines data exploration and visualization, asset management, and collaboration capabilities in an interface that is more intuitive for diverse science teams.

8.1.2 A Set of Design Guidelines for Ocean Observatory User Interfaces

I conducted a ten-week contextual design study with multiple ocean science groups at two oceanographic institutions to determine key user themes. Based on these themes, I presented the following interface design guidelines for an observatory data exploration system:

- Make high-quality geolocated 3D data visualization easier.
- Integrate with existing task-specific tools, when feasible.
- Add support for ocean modeling processes.
- Provide a flexible and scalable data architecture.
- Make instrument layout interactive and visual.
- Provide for a wide range of visualization audiences.
- Focus on sharing visualizations instead of data.

8.1.3 COVE: A Collaborative Environment for the Ocean Sciences

Based on these guidelines, I described the design and implementation of COVE, a new science tool created in close collaboration with ocean scientists. COVE's user interface is modeled on the geobrowser paradigm, which provides an intuitive, multi-scale interface, but with an extensive set of additions to provide an integrated tool for observatory needs. To support data exploration, it provides native geolocated 3D scientific visualization, high resolution custom bathymetry, and tools to interactively explore ocean models. For observatory design, it has a drag-and-drop instrument management interface with on-screen status tracking and swappable instrument libraries. To support scientific collaboration and communication, there are user-defined interactive views, high resolution image and movie output, and a Web-based repository to share data and visualizations. Finally, the architecture is flexible and scalable with a scripting interface to import new data formats, a Web-based workflow component to analyze data, and a layered design to run across local, server, and cloud environments.

8.1.4 Evaluation of COVE for Usability and Real-World Science Impact

I evaluated COVE with three different methodologies to verify its effectiveness. In usability studies I showed that COVE worked effectively in two important ways: for data exploration experts and novices and for visualization producers and consumers. In deployments of COVE in three real-world science environments, I validated the design and determined modifications based on user feedback. In each deployment, COVE proved to be an intuitive environment for individuals to explore data and carry out interactive tasks, a common space for groups to share discussions and capture important points, and a familiar context for large teams to present progress and findings, both internally and to outside groups.

8.1.5 A Quantitative Evaluation of COVE Scalability to Server and Cloud Platforms

To evaluate scalability, I described COVE's distributed data exploration architecture in depth and described a range of possible configurations that spanned client, server, and cloud platforms. I then presented a test suite of representative visual data analytics tasks and experimentally compared these configurations using the test suite. Finally, I reported and analyzed their performance, and concluded that the seamless Client+Cloud COVE architecture – as opposed to Cloud-Alone or Client-Alone – was best suited for visual ocean data analysis needs.

8.2 Future Work

While COVE includes a significant set of capabilities, the domain of ocean data exploration and ocean observatories is large, and this work focused primarily on end user interface needs. To continue to evolve the platform and better serve ocean scientists, the following areas, not addressed in this thesis, provide avenues for future research.

8.2.1 Integrated Visual Search

With the growing number and size of datasets that populate physical and virtual observatories, providing rich user interfaces to search these collections will be vitally

important. While COVE offers a natural environment for such interfaces, it currently provides no search facilities beyond simple ordering and distance filtering. Visual search mechanisms can take advantage of the geobrowser interface to provide interactive query and filtering capabilities to find datasets relevant to a location, time, or data type. Using the information captured in shareable views can also allow searches focused on characteristics of the current visualization.

8.2.2 Observatory Simulation

As the observatory design process progresses, simulation of observatory processes to validate designs will take greater importance to limit post-deployment issues. One task consists of simulating the power and communication that is traveling between the shore station and the instruments, nodes, junctions, and cables in the ocean. Another is simulating the data collection that a certain configuration can expect based on historical and ocean model data, which will allow science teams to best place instruments. Both of these tasks would benefit from the COVE interface to interactively iterate designs based on simulated feedback.

8.2.3 Observatory Operation

Another possible research direction is COVE's use for day-to-day observatory operations. This is a user request that arose during the RSN observatory design deployment. As COVE was already being used to position instruments and cables, it could be used to provide fluid integration from design through deployment to daily monitoring of instruments. It could also provide an interactive mechanism to deploy, simulate, and visualize instruments and platforms during ocean events, such as volcanoes and earthquakes.

8.2.4 Supporting other Earth Sciences

While the primary focus of the investigation into COVE's interface has been with the ocean sciences, its capabilities should scale well to other earth sciences that use geolocated data. Regional atmospheric models of temperature and pressure have been displayed in COVE.

Collaboration is ongoing with hydrological scientists at the University of Washington to display model inputs and simulated outputs, such as stream flows, both locally and internationally (e.g., the Mekong Basin). And there has been a growing interest from environmental meta-genomics for displaying data in Puget Sound.

8.2.5 Disaster Response

In many ocean disasters, the ability to quickly and easily view a variety of datasets together is crucial to successfully handling the emergency. This may include recent events such as the 2010 Gulf of Mexico oil spill or 2011 Japanese tsunami and nuclear crisis, where immediately available interactive simulations of possible outcomes would have been beneficial to those managing the crisis, as well as those affected by it. Other scenarios might extend to the search for a missing sailor, whose chance of rescue depends on quickly determining currents and likely position. COVE could provide an interface to explore data, determine responses, and quickly communicate interactive visuals to the public.

8.2.6 Science Presentations

Presentation of scientific work at conferences and to peers is still primarily done using presentation tools designed for business users. User feedback suggests that COVE could provide an effective alternative to this page-based way of presenting scientific results. COVE already provides many of the capabilities necessary – views or slides, titles, simple screen overlays, transitioning between views – and could be enhanced to provide integration with current tools and, where applicable, replace them altogether.

8.2.7 COVE Public Web Sharing Site

Finally, an exciting area of research is to make the collaborative features of COVE available on a broader basis as a public Web service. COVE currently provides a visualization sharing system that is designed primarily for groups working on a local network and would require more robust solutions for personal and group access, security, and world-wide availability.

With these refinements, COVE could support the increasing democratization of science by enabling the sharing of data and perspectives beyond science teams to reach citizen scientists throughout the world.

8.3 Closing Remarks

Increasingly, news headlines stress the vital importance of our oceans, e.g., declining fish stocks, tsunamis, global warming, energy opportunities, and disasters. These reports also demonstrate how much we have yet to learn about this vast environment. To meet this challenge, ocean observatories offer a new era in ocean science. They support a sophisticated array of instruments, integrate access to resources and data, and enable new scientific avenues of discovery. Where possible, scientists will leverage existing systems for this new science platform, but their success will also depend on software and interface designers, who will need to better understand these new environments and provide new tools and interfaces to meet their needs.

This thesis presents my work designing a system for these new science platforms. COVE was developed from a close collaboration with scientists to understand more about the processes and needs of ocean observatories and deliver an integrated environment for data exploration and visualization, planning, and collaboration. COVE's design was validated through a series of deployments in real-world science environments, which demonstrated that by providing an intuitive common visual environment for science teams, systems like COVE can play a pivotal role in helping ocean observatories fulfill their missions.

BIBLIOGRAPHY

[1] ABRAM, G. AND TREINISH, L. 1995. An Extended Data-Flow Architecture for Data Analysis and Visualization. In *Proceedings of the IEEE Symposium on Visualization*, 1995 IEEE Computer Society, 833845, 263.

[2] AHLBERG, C. AND WISTRAND, E. 1995. IVEE: an Information Visualization and Exploration Environment. In *Proceedings of the IEEE Symposium on Information Visualization*, 1995, 66.

[3] AKERS, D. 2006. CINCH: A Cooperatively Designed Marking Interface for 3D Pathway Selection. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 2006, 33-42.

[4] ALVIN, 2010. http://oceanexplorer.noaa.gov/technology/subs/alvin/alvin.html.

[5] AMAR, R. AND STASKO, J. 2004. A Knowledge Task-Based Framework for Design and Evaluation of Information Visualizations. In *Proceedings of the IEEE Symposium on Information Visualization*, 2004, 143-150.

[6] ARNSTEIN, L., HUNG, C.Y., FRANZA, R., ZHOU, Q.H., BORRIELLO, G., CONSOLVO, S. AND SU, J. 2002. Labscape: A Smart Environment for the Cell Biology Laboratory. *IEEE Pervasive Computing Magazine 1*, 13-21.

[7] ARSENAULT, R., WARE, C., PLUMLEE, M., MARTIN, S., WHITCOMB, L., WILEY, D., GROSS, T. AND BILGILI, A. 2004. A System for Visualizing Time Varying Oceanographic 3D Data. In *Proceedings of the IEEE OCEANS*, 2004, 743-747.

[8] BARCLAY, T., GRAY, J. AND SLUTZ, D. 2000. Microsoft TerraServer: a spatial data warehouse. *SIGMOD Rec. 29*, 307-318.

[9] BARGA, R.S., JACKSON, J., ARAUJO, N., GUO, D., GAUTAM, N., GROCHOW, K. AND LAZOWSKA, E. 2008. Trident: Scientific Workflow Workbench for Oceanography. In *Proceedings of the IEEE Congress on Services*, 2008 IEEE Computer Society, 1439049, 465-466.

[10] BAUDEL, T. 2006. From Information Visualization to Direct Manipulation: Extending a Generic Visualization Framework for the Interactive Editing of Large Datasets. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 2006, 67-76.

[11] BAVOIL, L., CALLAHAN, S.P., SCHEIDEGGER, C.E., VO, H.T., CROSSNO, P.J., SILVA, C.T. AND FREIRE, J. 2005. VisTrails: Enabling Interactive Multiple-View Visualizations. In *Proceedings of the IEEE Symposium on Visualization*, 2005, 18.

[12] BEDERSON, B.B. AND BOLTMAN, A. 1999. Does Animation Help Users Build Mental Maps of Spatial Information? In *Proceedings of the IEEE Symposium on Information Visualization*, 1999, 28.

[13] BEDERSON, B.B., GROSJEAN, J. AND MEYER, J. 2004. Toolkit Design for Interactive Structured Graphics. *IEEE Transactions on Software Engineering 30*, 535-546.

[14] BEDERSON, B.B. AND HOLLAN, J.D. 1995. Pad++: a zoomable graphical interface system. In *Proceedings of the Conference Companion on Human Factors in Computing Systems*, 1995 ACM, 223394, 23-24.

[15] BÖDKER, S. AND GRÖNBÆK, K. 1992. Design in action: from prototyping by demonstration to cooperative prototyping. In *Design at work: cooperative design of computer systems* L. Erlbaum Associates Inc.

[16] BOULOS, J. AND ONO, K. 1999. Cost estimation of user-defined methods in object-relational database systems. *ACM SIGMOD Record 28*, 22-28.

[17] BUSCHMANN, C., PFISTERER, D., FISCHER, S., FEKETE, S.P. AND KRÖLLER, A. 2005. SpyGlass: A Wireless Sensor Network Visualizer. *SIGBED Review 2*, 1-6.

[18] CARD, S.K. AND MACKINLAY, J. 1997. The Structure of the Information Visualization Design Space. In *Proceedings of the IEEE Symposium on Information Visualization*, 1997, 92.

[19] CHI, E.H., RIEDL, J., BARRY, P. AND KONSTAN, J. 1998. Principles for Information Visualization Spreadsheets. *IEEE Computer Graphics and Applications 18*, 30-38.

[20] COUGHLAN, J.C. AND HOGAN, P. 2006. Connecting Virtual Observatories with Visualization and Display Tools, Lessons Learned with NASA World Wind. *AGU Fall Meeting Abstracts*, A811+.

[21] DARKEN, R.P. AND SIBERT, J.L. 1993. A Toolset for Navigation in Virtual Environments. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 1993, 157-165.

[22] DEELMAN, E., SINGH, G., SU, M.-H., BLYTHE, J., GIL, Y., KESSELMAN, C., MEHTA, G., VAHI, K., BERRIMAN, G.B., GOOD, J., LAITY, A., JACOB, J.C. AND KATZ, D.S. 2005. Pegasus: A framework for mapping complex scientific workflows onto distributed systems. *Scientific Programming 13*, 219-237.

[23] DOTY, B.E., WIELGOSZ, J., GALLAGHER, J. AND HOLLOWAY, D. 2001. GrADS and DODS/OPENDAP. *Proceedings of the 17th International Conference on Interactive*

Information and Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology, American Meteorological Society Albuquerque, NM 385.

[24] EDGINGTON, D.R., DAVIS, D. AND O'REILLY, T.C. 2006. Ocean Observing System Instrument Network Infrastructure. In *Proceedings of the IEEE OCEANS*, 2006, 1-5.

[25] FLTK: The fast light toolkit, 2010. http://www.fltk.org.

[26] FOULSER, D. 1995. IRIS Explorer: A Framework for Investigation. *SIGGRAPH 29*, 13-16.

[27] FREW, J. AND BOSE, R. 2001. Earth System Science Workbench: A Data Management Infrastructure for Earth Science Products. In *Proceedings of the Conference on Scientific and Statistical Database Management*, 2001, B. RAJENDRA Ed., 0180-0180.

[28] GAITHER, K. AND II, R.J.M. 1995. Visualizing Vector Information in Ocean Environments. In *Proceedings of the MTS/IEEE*, 1995, 1907-1914.

[29] GALYEAN, T.A. 1995. Guided Navigation of Virtual Environments. In *Proceedings of the Symposium on Interactive 3D Graphics*, 1995, 103-ff.

[30] GOBLE, C.A. AND ROURE, D.C.D. 2007. myExperiment: Social Networking for Workflow-Using e-Scientists. In *Proceedings of the Workshop on Workflows in Support of Large-scale Science*, 2007, 1-2.

[31] Google Earth, 2010. http://earth.google.com.

[32] Google Oceans, 2010. http://earth.google.com/oceans.

[33] GRAY, J., LIU, D.T., NIETO-SANTISTEBAN, M., SZALAY, A., DEWITT, D.J. AND HEBER, G. 2005. Scientific Data Management in the Coming Decade. *ACM SIGMOD Record 34*, 34-41.

[34] GROCHOW, K., HOWE, B., LAZOWSKA, E., BARGA, R. AND STOERMER, M. 2010. Client + Cloud: Evaluating Seamless Architectures for Visual Data Analytics in the Ocean Sciences. In *Proceedings of the 22nd International Conference on Scientific and Statistical Database Management*, Heidelberg, Germany, 2010 Springer Verlag.

[35] GROCHOW, K., STOERMER, M., FOGARTY, J., LEE, C., HOWE, B. AND LAZOWSKAA, E. 2010. COVE: A Visual Environment for Multidisciplinary Ocean Science Collaboration. In *Proceedings of the IEEE Conference on eScience*, Brisbane, Queensland, Australia, 2010 IEEE.

[36] GROTH, D.P. AND STREEFKERK, K. 2006. Provenance and Annotation for Visual Exploration Systems. *IEEE Transactions on Visualization and Computer Graphics 12*, 1500-1510.

[37] HANKIN, S., HARRISON, D.E., OSBORNE, J., DAVISON, J. AND O'BRIEN, K. 1996. A Strategy and a Tool, Ferret, for Closely Integrated Visualization and Analysis. *The Journal of Visualization and Computer Animation 7*, 149–157.

[38] HEER, J., VI\'EGAS, F.B. AND WATTENBERG, M. 2007. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2007, 1029-1038.

[39] HEY, T., TANSLEY, S. AND TOLLE, K. 2009. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research.

[40] HIBBARD, B. 2000. Confessions of a visualization skeptic. *SIGGRAPH Comput. Graph. 34*, 11-13.

[41] HIBBARD, B. 1998. VisAD: connecting people to computations and people to people. *SIGGRAPH 32*, 10-12.

[42] HIBBARD, W., B\"OTTINGER, M., SCHULTZ, M. AND BIERCAMP, J. 2002. Visualization in Earth System Science. *SIGGRAPH 36*, 5-9.

[43] HOLTZBLATT, K. AND JONES, S. 1993. Contextual inquiry: A participatory technique for system design. *Participatory design: Principles and practices*, 177-210.

[44] HORNBÆK, K., BEDERSON, B.B. AND PLAISANT, C. 2002. Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Transactions on Computer Human Interaction 9*, 362-389.

[45] HOWE, B.M. AND MCGINNIS, T. 2003. Sensor Networks for Cabled Ocean Observatories. *Scientific Use of Submarine Cables and Related Technologies, 2003. The 3rd International Workshop on*, 25-27.

[46] JENTER, H. AND SIGNELL, R. 1992. NetCDF: A public-domain software solution to data-access problems for numerical modelers. *Preprints of the American Society of Civil Engineers Conference on Estuarine and Coastal Modeling*, 72.

[47] JUL, S. AND FURNAS, G.W. 1998. Critical zones in desert fog: aids to multiscale navigation. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 1998 ACM, 288578, 97-106.

[48] KAPLER, T. AND WRIGHT, W. 2004. GeoTime Information Visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, 2004, 25-32.

[49] KREUSELER, M., NOCKE, T. AND SCHUMANN, H. 2004. A History Mechanism for Visual Data Mining. In *Proceedings of the IEEE Symposium on Information Visualization*, 2004, 49-56.

[50] LANGLEY, P. 2000. The Computational Support of Scientific Discovery. *International Journal on Human Computer Studies 53*, 393-410.

[51] LEE, C.P. 2007. Boundary Negotiating Artifacts: Unbinding the Routine of Boundary Objects and Embracing Chaos in Collaborative Work. In *Proceedings of the Computer Supported Cooperative Work*, 2007 Kluwer Academic Publishers, 1265457, 307-339.

[52] LEE, C.P., DOURISH, P. AND MARK, G. 2006. The human infrastructure of cyberinfrastructure. In *Proceedings of the Computer Supported Cooperative Work*, Banff, Alberta, Canada, 2006 ACM, 1180950, 483-492.

[53] LI, P., CHAO, Y., VU, Q., LI, Z., FARRARA, J., ZHANG, H. AND WANG, X. 2006. OurOcean: An Integrated Solution to Ocean Monitoring and Forecasting. In *Proceedings of the IEEE OCEANS*, 2006, 1-6.

[54] LI, W., RITTER, L., AGRAWALA, M., CURLESS, B. AND SALESIN, D. 2007. Interactive Cutaway Illustrations of Complex 3D Models. *ACM Trans. Graph. 26*, 31.

[55] LI, W., RITTER, L., AGRAWALA, M., CURLESS, B. AND SALESIN, D. 2007. Interactive Cutaway Illustrations of Complex 3D Models. *ACM Transactions on Graphics 26*, 31.

[56] LUDÄSCHER, B., ALTINTAS, I., BERKLEY, C., HIGGINS, D., JAEGER, E., JONES, M., LEE, E.A., TAO, J. AND ZHAO, Y. 2006. Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice and Experience 18*, 1039-1065.

[57] MACEACHREN, A.M. 1994. *Visualization in Modern Cartography: Designing a Visualization User Interface*. Elsevier Science Inc., New York, NY, USA.

[58] MACKINLAY, J. 1986. Automating the Design of Graphical Presentations of Relational Information. *ACM Transactions on Graphics 5*, 110-141.

[59] MAECHLING, P., CHALUPSKY, H., DOUGHERTY, M., DEELMAN, E., GIL, Y., GULLAPALLI, S., GUPTA, V., KESSELMAN, C., KIM, J., MEHTA, G., MENDENHALL, B., RUSS, T., SINGH, G., SPRARAGEN, M., STAPLES, G. AND VAHI, K. 2005. Simplifying Construction of Complex Workflows for Non-Expert Users of the Southern California Earthquake Center Community Modeling Environment. *ACM SIGMOD Record 34*, 24-30.

[60] MASSION, G. 2006. Ocean Observing Systems: Vision and Details. In *Proceedings of the IEEE OCEANS*, 2006, 1-6.

[61] MATHWORKS 1992. *Matlab Users Guide*.

[62] MayaVi, 2009. http://mayavi.sourceforge.net/.

[63] Monterey Bay Aquarium Research Institute, 2010. http://www.mbari.org.

[64] MCGUINNESS, D.L., FOX, P., CINQUINI, L., WEST, P., GARCIA, J., BENEDICT, J.L. AND MIDDLETON, D. 2010. Enabling Scientific Research using an Interdisciplinary Virtual Observatory: The Virtual Solar-Terrestrial Observatory Example. *Association for the Advancement of Artificial Intelligence*.

[65] Microsoft Virtual Earth, 2010. http://www.viawindowslive.com/VirtualEarth.aspx.

[66] Microsoft Windows Azure Platform, 2010. http://www.microsoft.com/windowsazure/.

[67] MURRAY, D., MCWHIRTER, J., WIER, S. AND EMMERSON, S. 2003. The Integrated Data Viewer—a web-enabled application for scientific analysis and visualization. In *Proceedings of the IIPS for Meteorology, Oceanography and Hydrology*, 2003, 8-13.

[68] NASA World Wind, 2010. http://worldwind.arc.nasa.gov/manual.html.

[69] NATH, S., LIU, J. AND ZHAO, F. 2007. SensorMap for Wide-Area Sensor Webs. *Computer 40*, 90-93.

[70] NEWMAN, H.B., ELLISMAN, M.H. AND ORCUTT, J.A. 2003. Data-intensive e-science frontier research. *Communications of the ACM 46*, 68-77.

[71] ORION's Ocean Observatories Initiative Conceptual Network Design: A Revised Infrastructure Plan, 2007. http://www.ooi.washington.edu/.

[72] Ocean Data View, 2007. http://odv.awi.de.

[73] OLSON, G.M., ATKINS, D.E., CLAUER, R., FINHOLT, T.A., JAHANIAN, F., KILLEEN, T.L., PRAKASH, A. AND WEYMOUTH, T. 1998. The Upper Atmospheric Research Collaboratory (UARC). *Interactions 5*, 48-55.

[74] ParaView, 2010. http://www.paraview.org/.

[75] PLAISANT, C., HELLER, D., LI, J., SHNEIDERMAN, B., MUSHLIN, R. AND KARAT, J. 1998. Visualizing medical records with LifeLines. In *Proceedings of the ACM Conference on Human Factors in Computing*, Los Angeles, California, United States, 1998 ACM, 286513, 28-29.

[76] PLAISANT, C., MILASH, B., ROSE, A., WIDOFF, S. AND SHNEIDERMAN, B. 1996. LifeLines: visualizing personal histories. In *Proceedings of the ACM Conference on Human Factors in Computing*, 1996, 221-ff.

[77] PLALE, B., ALAMEDA, J., WILHELMSON, B., GANNON, D., HAMPTON, S., ROSSI, A. AND DROEGEMEIER, K. 2005. Active Management of Scientific Data. *IEEE Internet Computing 9*, 27-34.

[78] POOK, S., LECOLINET, E., VAYSSEIX, G. AND BARILLOT, E. 2000. Context and interaction in zoomable user interfaces. In *Proceedings of the Proceedings of the working conference on Advanced visual interfaces*, Palermo, Italy, 2000 ACM, 345323, 227-231.

[79] PRAGER, E.J. AND EARLE, S.A. 2000. *The Oceans*. McGraw-Hill, New York, NY, USA.

[80] RHYNE, T.M. AND MACEACHREN, A. 2004. Visualizing Geospatial Data. In *Proceedings of the SIGGRAPH Course Notes*, 2004, 31.

[81] RIBES, D. AND FINHOLT, T.A. 2008. Representing community: knowing users in the face of changing constituencies. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, San Diego, CA, USA, 2008 ACM, 1460581, 107-116.

[82] ROBERTS, J.C. 2000. Multiple View and Multiform Visualization. In *Proceedings of the SPIE*, 2000, 176-185.

[83] ROSSUM, G.V. 1995. Python tutorial. *Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI)*.

[84] SCHEIDEGGER, C.E., VO, H.T., KOOP, D., FREIRE, J. AND SILVA, C.T. 2007. Querying and Creating Visualizations by Analogy. In *Proceedings of the IEEE Symposium on Visualization*, 2007.

[85] SCHRAEFEL, M.C., HUGHES, G.V., MILLS, H.R., SMITH, G., PAYNE, T.R. AND FREY, J. 2004. Breaking the book: translating the chemistry lab book into a pervasive computing lab environment. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2004, 25-32.

[86] SCHROEDER, W., MARTIN, K. AND LORENSEN, B. 2003. *The Visualization Toolkit*. Kitware.

[87] SEMTNER, A. 2000. Ocean and Climate Modeling. *Communications of the ACM 43*, 80-89.

[88] SHEN, Y., CROUCH, J.R., AUSTIN, J.A. AND DINNIMAN, M.S. 2007. Interactive Visualization of Regional Ocean Modeling System. In *Proceedings of the Graphics and Visualization in Engineering*, 2007.

[89] SHNEIDERMAN, B. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the IEEE Symposium on Visual Languages*, 1996, 336.

[90] SHU, L., WU, C., ZHANG, Y., CHEN, J., WANG, L. AND HAUSWIRTH, M. 2008. NetTopo: beyond simulator and visualizer for wireless sensor networks. *SIGBED Rev. 5*, 1-8.

[91] SNAVELY, N., SEITZ, S.M. AND SZELISKI, R. 2006. Photo Tourism: Exploring Photo Collections in 3D. In *Proceedings of the SIGGRAPH*, 2006, 835-846.

[92] SPRINGMEYER, R.R., BLATTNER, M.M. AND MAX, N.L. 1992. A Characterization of the Scientific Data Analysis Process. In *Proceedings of the IEEE Symposium on Visualization*, 1992, 235-242.

[93] STAR, S. AND GRIESEMER, J. 1989. Institutional ecology,'translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social Studies of Science*, 387-420.

[94] STOLTE, C., TANG, D. AND HANRAHAN, P. 2002. Multiscale Visualization Using Data Cubes "InfoVis 2002 Best Paper". In *Proceedings of the IEEE Symposium on Information Visualization*, 2002, 7.

[95] STOLTE, C., TANG, D. AND HANRAHAN, P. 2002. Polaris: A System for Query, Analysis, and Visualization of Multidimensional Relational Databases. *IEEE Transactions on Visualization and Computer Graphics 8*, 52-65.

[96] STOLTE, E., VON PRAUN, C., ALONSO, G. AND GROSS, T. 2003. Scientific Data Repositories: Designing for a Moving Target. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2003, 349-360.

[97] SUTHERLAND, I.E. 1968. A Head-Mounted Three Dimensional Display. In *Proceedings of the Fall Joint Computer Conference*, 1968, 757-764.

[98] SVERDRUP, K.A. AND ARMBRUST, V. 2006. *An Introduction to the World's Oceans, 9th ed.* McGraw-Hill Science, New York, NY, USA.

[99] SZALAY, A. AND GRAY, J. 2006. 2020 Computing: Science in an Exponential World. *Nature 440*, 413-414.

[100] TAKATSUKA, M. AND GAHEGAN, M. 2002. GeoVISTA Studio: A Codeless Visual Programming Environment for Geoscientific Data Analysis and Visualization. *Computational Geosciences 28*, 1131-1144.

[101] TAN, D.S., ROBERTSON, G.G. AND CZERWINSKI, M. 2001. Exploring 3D Navigation: Combining Speed-Coupled Flying with Orbiting. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 2001, 418-425.

[102] THURMAN, H. 2007. *Essentials of Oceanography, 9th ed.* Prentice Hall, New York, NY, USA.

[103] The Triana Project, 2010. http://www.trianacode.org.

[104] TUFTE, E.R. 1990. *Envisioning Information*. Graphics Press.

[105] TUFTE, E.R. 1997. *Visual Explanations: Images and Quantities (2nd edition)*. Graphics Press.

[106] University of Washington School of Oceanography, 2010. http://www.cofs.washington.edu/.

[107] UPSON, C., THOMAS FAULHABER, J., KAMINS, D., LAIDLAW, D.H., SCHLEGEL, D., VROOM, J., GURWITZ, R. AND VAN DAM, A. 1989. The Application Visualization System: A Computational Environment for Scientific Visualization. *IEEE Computer Graphics and Applications 9*, 30-42.

[108] VALÉRIA, M.B., CAVALCANTI, SCHIEL, U. AND BAPTISTA, C.D.S. 2006. Querying Spatio-Temporal Databases using a Visual Environment. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, 2006, 412-419.

[109] VAN WIJK, J.J. 2002. Image Based Flow Visualization. In *Proceedings of the SIGGRAPH*, 2002, 745-754.

[110] VAN WIJK, J.J. 2005. The Value of Visualization. In *Proceedings of the IEEE Symposium on Visualization*, 2005, 11.

[111] VIOLA, I., KANITSAR, A. AND GROLLER, M.E. 2004. Importance-Driven Volume Rendering. In *Proceedings of the IEEE Symposium on Visualization*, 2004, 139-146.

[112] WARE, C., PLUMLEE, M., MARTIN, S., WHITCOMB, L.L., WILEY, D., GROSS, T. AND BILGILI, A. 2001. GeoZui3D: Data Fusion for Interpreting Oceanographic Data. In *Proceedings of the MTS/IEEE*, 2001, 1960–1964.

[113] WEAVER, C. 2004. Building Highly-Coordinated Visualizations in Improvise. In *Proceedings of the IEEE Symposium on Information Visualization*, Washington, DC, 2004 IEEE Computer Society, 159-166.

[114] WINN, W.D., WINDSCHITL, M., FRULAND, R. AND LEE, Y. 2002. A virtual environment designed to help students understand science. *Proceedings of the International Conference of Learning Societies*.

[115] WOOD, J., DYKES, J., SLINGSBY, A. AND CLARKE, K. 2007. Interactive Visual Exploration of a Large Spatio-temporal Dataset: Reflections on a Geovisualization Mashup. *IEEE Transactions on Visualization and Computer Graphics 13*, 1176-1183.

APPENDIX A

OBSERVATORIES AND SCIENCE DATA PORTALS

Reprinted from Virtual Observatory Working Group: International Geophysics Year: 2007-2008

Earth Observing System Initiatives

ACOS - Advanced Coronal Observing System	http://mlso.hao.ucar.edu/
AEO - Alliance for Earth Observations	http://www.strategies.org/Alliance/default.htm
CEOS - Committee on Earth Observation Satellites	http://www.ceos.org
WGISS - Working Group on Information Systems and Services	http://wgiss.ceos.org
ESONET - European Sea Floor Observatory Network	http://www.abdn.ac.uk/ecosystem/esonet/index2.htm
IEOS - Integrated Earth Observation System	http://www.noaa.gov/lautenbacher/oceanology.htm
IGOS - International Global Observing Strategy	http://www.igospartners.org
IOOS - Integrated and Sustained Ocean Observing System	http://www.ocean.us
ION - International Ocean Network	http://seismo.berkeley.edu/seismo/ion
IGGOS - Integrated Global Geodetic Observing System	http://www.gfy.ku.dk/~iag/iggos_prop_june_03.htm
GCOS - Global Climate Observing System	http://www.wmo.ch/web/gcos/gcoshome
GOOS - Global Ocean Observing System	http://www.ocean.us
GOSIC - The Global Observing Systems Information Center	http://www.gosic.org
OOI - Ocean Observatories Initiative	http://www.orionprogram.org/OOIl

Virtual Observatories

ASTROGRID Virtual Observatory	http://www.astrogrid.org/project
AVO - Astrophysical Virtual Observatory	http://www.eurovo.org/, http://www.aus-vo.org/
CARISMA - Canadian Array for Realtime Investigations of Magnetic Activity	http://www.carisma.ca
WTF-CEOP - The WGISS Test Facility for CEOP	http://jaxa.ceos.org/wtf_ceop
CoSEC - Collaborative Sun Earth Connector	http://cosec.lmsal.com/
Earth Observatory	http://earthobservatory.nasa.gov/
EGSO - European Grid of Solar Observatories	http://www.egso.org/
GAIA - Global Auroral Imaging Access	http://gaia-vxo.org
ICESTAR - Interhemispheric Conjugacy Effects on Solar-Terrestrial and Aeronomy Research	http://www.siena.edu/physics/ICESTAR/
IVOA - International Virtual Observatory Alliance	www.ivoa.net
NVO - US National Virtual Observatory	http://www.us-vo.org/
NVODS - National Virtual Ocean Data System	http://www.nvods.org
VSN - Virtual Seismic Network	http://eqinfo.ucsd.edu/vsn/
VGMO - Virtual Global Magnetic Observatory	http://mist.engin.umich.edu/mist/vgmo/vgmo.html
VHO - Virtual Heliospheric Observatory	http://vho.nasa.gov/
ViRBO - Virtual Radiation Belt Observatory	http://virbo.org/
VITMO - Virtual Ionosphere-Mesosphere-Thermosphere Observatory	http://omniweb.gsfc.nasa.gov/vitmo/
VSO - Virtual Solar Observatory	http://virtualsolar.org/
VSPO - Virtual Space Physics Observatory	http://vspo.gsfc.nasa.gov
VSTO - Virtual Solar-Terrestrial Observatory	http://vsto.hao.ucar.edu/
VMOs - Virtual Magnetospheric Observatories	http://lwsde.gsfc.nasa.gov/VxO_05_Selections.pdf

Other Data & Information Portals

AEON - Australian Earth and Ocean Network	http://www.aeon.org.au/
BIOS - Biological Innovation for Open Society	http://www.bios.org
BlueNet - The Australian Marine Science Data Network	http://www.utas.edu.au/cms/news_events/bluenet.html
CANRI - Community Access to Natural Resource Information,	http://www.canri.nsw.gov.au/
CIG - Computational Infrastructure in Geodynamics	http://geodynamics.org
CEDARweb - Coupling, Energetics and Dynamics of Atmospheric Regions Data systems	http://cedarweb.hao.ucar.edu
EarthScope	http://www.earthscope.org
ECHO - Earth Observing System Clearing House	http://www.echo.nasa.gov/
EDNES - Earth Data Network for Education and Scientific Exchange	http://www.ednes.org/
EOSDIS - Earth Observing System Data and Information System	http://spsosun.gsfc.nasa.gov/eosinfo/Welcome/index.html
GEON - Global Earth Observing Network (national Geosciences Cyberinfrastructure Network)	http://www.geongrid.org
IGS - International GNSS Service	http://igscb.jpl.nasa.gov/
IgeS - International Geoid Service	http://www.iges.polimi.it/
IERS - International Earth Rotation and Reference Systems Service	http://www.iers.org/
ILRS - International Laser Ranging Service	http://ilrs.gsfc.nasa.gov/
IRIS - Incorporated Research Institutions for Seismology	http://www.iris.edu/

APPENDIX B

USABILITY EVALUATION TASKS

During this lab exercise we will explore a computerized model of the Puget Sound Basin. We will use the Collaborative Ocean Visualization Environment (COVE), which is being developed at UW, to visualize various oceanographic datasets. We will explore oceanographic properties in Puget Sound such as temperature, salinity, and circulation. The interface allows you to choose a wide range of views of simulated data that were generated previously by a numeric model.

Our goal for this lab is to understand how these physical parameters interact to create oxygenated versus anoxic conditions in the water column. Specifically, we are interested in understanding the physical oceanographic mechanisms responsible for the low levels of dissolved oxygen in Hood Canal. By comparing different sites throughout the Sound, you will be able to visually comprehend how water stratification and flushing rates (the two major physical causes of the O_2 problem) affect these properties.

The geology and bathymetry of Hood Canal play a large role in the water quality and dynamics of how the water moves. The entrance to the canal is a relatively shallow sill, but just south of the entrance the canal becomes very deep. This sill at the entrance creates a condition in the canal that doesn't allow deep water to flow or exchange very easily with the changing tides and seasons. Flushing rate in the canal is on the order of 1-2 years, while outside the Canal in the Main Basin the flushing rate is 2-3 months.

Introduction to COVE data visualization software

Click on the "COVE" icon on the desktop to load the program.

i) Navigation in COVE is similar to what you're used to with Google Earth Pro. Click and hold the left mouse button to move your location on the map (pan).

ii) To zoom in and out use the scroll wheel on your mouse or click and hold the two sided arrow in the center of the compass rose (bottom right). You can change the North/South orientation as well as the angle of your view by clicking and holding the scroll wheel on the outer ring of the compass rose. To return to a normal overhead view push the zero key

iii) To control the lighting intensity and direction of shading, adjust the circular slider tool to the left of the compass rose.

iv) The dashboard at the top of your screen has many useful tools, such as a distance measuring tool, the ability to view the earth as a spherical or flat image, and the ability to toggle UTM or Lat/Long coordinates. The "Data Layers" tool allows you turn on/off different layers as we did in Google Earth Pro. The "Settings" tool allows customizing how data is visualized.

v) On the second row of dashboard tools you will find a list of different "Views." These are pre-made views of certain data and locations. You can create your own view by clicking the "Save new view" button if you'd like to be able to easily go back to a certain region of interest. Before starting the lab exercise, familiarize yourself with navigating thru COVE. ***IF YOU ENCOUNTER ERRORS CLOSE AND RESTART THE PROGRAM***

Part I: Exploring Puget Sound Bathymetry

To begin, select the view labeled "Bathymetry" from the COVE dashboard. This will display a data layer containing moderately high resolution bathymetry for the seafloor. As you zoom in to Puget Sound, the view will become more detailed. There is a legend showing you the scale of depths on the lower left corner of your screen. Also, when you scroll your mouse over an area, you are given the lat/long and also the depth or altitude of that location on the upper right corner of your screen.

1) Where are the shallowest regions of Puget Sound (i.e. what basin)?

2) Where are the deepest regions of Puget Sound (i.e. what basin)?

3) Identify the latitude/longitude and depth of the sills at
   a) Hood Canal:
   b) Main Basin:
   c) South Sound:

4) What is the distance in kilometers from the entrance to Puget Sound to its most southerly point following the channel?

Part II: Surface Salinity and Temperature of Puget Sound

1) Select the "Salinity Surface" view from the COVE dashboard. This data layer displays surface salinity in the Puget Sound basin and coastal ocean. The data has been generated by a numeric ROMS model (Regional Ocean Model System) and represents a snapshot of surface properties for January 6, 2005.

    a) In what areas of Puget Sound is the surface salinity the highest?

    b) In what areas of Puget Sound is the surface salinity the lowest?

    c) What 2 factors drive these differences in salinity?

2) Now click on the "Data Explorer" tool on the COVE dashboard. In the pop-up box click on the "Color" subheading. This tool allows you to change the colors used to display data as well as change the maximum and minimum values used to scale the data.

    a) In the "Data Explorer" select the "Bins" box under the "Colors" heading. How does this change your view of the data?

    b) Change the maximum and minimum values of the scale to 0 and 25. With this scale what major features can you see more prominently, and why?

3) Repeat this exercise using temperature in place of salinity. To change to the surface temperature data layer, open the "Data Explorer" tool and click on the drop down menu labeled "Variable Displayed" at the top of the pop-up box. Select "temp" from the drop down menu. Notice that the scale will still be set to what you had for salinity, and the data may not show up properly. To have a scale automatically generated, uncheck the boxes next to "Minimum Value" and "Maximum Value" under the "Color" subheading.

    a) In what areas of Puget Sound is the surface temperature the highest, and what is the approximate temp (scale is in degrees Celsius)?

b) What is the range of temperatures across the entire Hood Canal basin?

Part III: Salinity and Temperature Cross Sections

1) Select the "Salinity cross-section" view from the COVE dashboard. This view allows you to look at the vertical profile of salinity or temperature across a section of your choice. The default cross section goes thru the Straits of Juan de Fuca into the Main Basin. You can make the 2D side view window larger or smaller by clicking and dragging the bottom right corner of the window.

   a) Using the default cross section, at what depths of Puget Sound is the salinity generally the greatest? What causes this salinity difference?

   b) Open the "Data Explorer" tool, and change the Variable Displayed to temperature (You may need to uncheck the min/max value boxes under the "Color" subheading). Using the default cross section, at what depths of Puget Sound is the temperature generally the greatest? What causes this temperature difference?

2) To customize the path of the cross section, open the "Data Explorer" tool and click on the "Display" subheading. Click on the "Edit Path" button, which will bring up a new window. You should see spheres along the line of your cross section. Left click and drag these spheres one by one to change the path of your cross section.

   a) Create a cross section of the whole Hood Canal basin. Show your instructor your cross section to get credit for this question.

   b) Where are the highest temperatures in the Hood Canal basin (i.e. general location and depth)?

Part IV: Vertical Salinity Profiles at PRISM Cruise Stations

1) Select the "Salinity Profiles" view from the COVE dashboard. This view displays salinity profiles collected during the December 2004 PRISM cruise. Note that these stations include the same ones that you visited during the Thompson cruise. The data

that you collected on your cruise can be visualized in the same manner. By clicking on the colored profile you can get exact salinity values for each depth of the profile.

a) Find the station at the entrance of Hood Canal (lat/long: 47.896500, -122.603667). What is the range of salinities here (i.e. exact values from surface and bottom)?

b) Find the station in the middle of the North Main Basin (47.812667, -122.453833). What is the range of salinities here (i.e. exact values from surface and bottom)?

c) Find the station near the north end of the Whidbey basin (48.238667, -122.556833). What is the range of salinities here (i.e. exact values from surface and bottom)?

d) Based on the above results, rank the 3 stations from most stratified to least stratified.

e) What factors make these stations more or less stratified relative to one another?

Part V: Puget Sound Circulation and Current Velocities

1) Select the "Surface currents" view from the COVE dashboard. This view displays modeled current velocities as vectors over a 24-hour tidal cycle. The direction of the vector indicates the direction of flow. Warmer colors (red) indicate higher current velocities; once velocities reach zero, the vector becomes transparent and disappears. Using the timeline on the bottom of the screen, you can either manually move thru time by dragging the timeline or simply push the play button. The spatial extent of this dataset has been reduced to limit the memory demand on the computers.

   a) In this dataset, where are current velocities generally the highest, why?

   b) Where are current velocities generally greater throughout the tidal cycle, the entrance of Hood Canal or the entrance of the Main Basin?

c) What 2 main factors drive circulation (i.e. currents) in Puget Sound?

2) To have an easier time determining the direction of currents you can display the vectors as arrows (warning: this will slow down the computer, so be patient). Make sure your timeline is stopped, then open the "Data Explorer" tool. Under the "Display" subheading, click on the "Show Arrows" box. To make the arrows nicer to view, change the length setting to 0.12 and width setting to 0.24

a) Zoom in on the Straits of Juan de Fuca, near where this dataset ends (where you can still see some arrows) and manually move the timeline to observe how the direction of current movement changes throughout the day. What time do the first high and low tides of the day occur? Remember there are 2 highs and 2 lows, so only look at the first half of the day to fill out the following table:

	Ebb	1st Low tide (slack)	Flood	1st High Tide (slack)
Time				

3) Select the "Currents Different Levels" view from the COVE dashboard. This view displays current velocities at three depths. The red vectors are surface velocities, yellow is mid-depth, and blue is near the bottom. As velocities reach zero the vectors become more transparent. To view the different depths you will need to angle your view so that you are looking into the basin.

a) Zoom in so that you can see the current velocities at all 3 depths simultaneously near the entrance to the main basin (48.056976, -122.624195). During an incoming (flood) tide, does the water move landward at all depths?

b) Explain any observations you have about the magnitude timing of the currents at each depth during the flood tide (i.e. does one depth start incoming sooner/later or weaker/stronger than the others?)

c) During an outgoing (ebb) tide, does the water move seaward at all depths?

d) Explain any observations you have about the magnitude timing of the currents at each depth during the ebb tide.

Part VI: Particle Advection

1) Select the "Advection" view from the COVE dashboard. Similar to dropping dye or beads into the physical model, this view shows you the path/destination of neutrally buoyant particles dropped at a certain location based on the currents you observed in Part V. The default color scale relates to the speed of the particle (i.e. the speed of the currents that the particle is in). Each particle starting location has many paths coming off of it, which show particles dropped at different starting times.

   a) Open the "Data Explorer" tool. From the "Display" subheading, click on the drop down box labeled "Value." Select "Start time" instead of speed. This will color the paths based on when the particle was released (blue released first, yellow released last. What do you notice about the direction of movement of the particles released first vs. last? Why is this?

   b) From the "Data Explorer" tool, click on the "Edit Particles" box, this allows you to move the origin of the particles by clicking and dragging the blue circle connected to the particle paths. Drag a particle origin out to the straights of Juan de Fuca. What is the net direction of movement of all particles released over the 24 hour tidal cycle? Why is this? (hint: these particles are released on the surface).

   c) (0.5 pts) From the "Data Explorer" tool, uncheck the "Release Particles Over Time" box. This lets you look only at the first particle dropped at the beginning of the tidal cycle. If you dropped your hat off of the Thomas G. Thompson at exactly lat/long 47.664249, -122.455843, where would your hat wash ashore (lat/long and name of location)?

   d) (0.5 pts) If you fell off of the Thomas G. Thompson at exactly 48.319864,-123.211784, how far would you float by the end of the day (distance from start to end, not total distance travelled)?

APPENDIX C

COLLECTED WORKFLOW SCENARIOS

Scenarios	Data Discovery, Access &Transformation	Data Analysis & Computation	Data Exploration & Visualization
Hydrographic Survey Cross-Sections	•Metadata cataloging •Collection indexing •Data calibration •Data transformations •Re-gridding •Re-projection	•Profiling and cross-section comparisons •2D interpolation of scalar and vector fields of models	•Contours •Cutting planes •Platform tracks •Standard plots •Textured topography
Observation and Ocean Model Comparison	•Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation •Data probes to time-series	•Data validation via observations •Profiling and cross-section comparisons •4D interpolation of scalar and vector fields of models	•Flow fields •Iso-surfaces •Cutting planes •Mobile platform tracks •Textured topography
Flow Field Visualization & Particle Advection	•Data transformations •Re-gridding •Re-projection	•4D interpolation of scalar and vector fields of models	•Flow fields •Iso-surfaces •Cutting planes •Stream-lines •Advected textures •Particle systems •Textured topography

Scenarios	Data Discovery, Access &Transformation	Data Analysis & Computation	Data Exploration & Visualization
Atmospheric Model Output	•Data transformations •Re-gridding •Re-projection	•4D interpolation of scalar and vector fields of models	•Flow fields •Iso-surfaces •Cutting planes •Particle systems •Textured topography
Operational Mooring Data Streams	•Metadata cataloging •Collection indexing •Data calibration •Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation •Data probes to time-series	•Profiling and cross-section comparisons	•Flow fields •Iso-surfaces •Cutting planes •Particle systems •Mooring platforms •Textured topography
Principle Component Analysis & Mooring Locations	•Data transformations •Re-gridding •Re-projection	•Principle Component Analysis	•Flow fields •Iso-surfaces •Cutting planes •Textured topography
Hydrographic Fluxes Through a Volume	•Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation •Data probes to time-series	•4D interpolation of scalar and vector fields of models •Computation of advective fluxes through defined surfaces •Computation of budgets of defined volumes	•Flow fields •Iso-surfaces •Cutting planes •Textured topography

Scenarios	Data Discovery, Access &Transformation	Data Analysis & Computation	Data Exploration & Visualization
Fluid-Rock Interactions and the Sub-seafloor Biosphere	•Data calibration •Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation •Data probes to time-series	•Detection, classification and localization of earthquakes •Time-series analysis	•Earthquakes •Time series •Sub-seafloor cross-sections •Sensor networks •Textured topography
Habitat and Organism Distributions	•Metadata cataloging •Collection indexing •Data transformations •Re-gridding •Re-projection •Thematic aggregation	•Temporal and spatial comparisons •4D interpolation of scalar and vector fields of models	•Flow fields •Iso-surfaces •Cutting planes •Stream-lines •Particle systems •Textured topography
Seafloor Mapping	•Metadata cataloging •Collection indexing •Data transformations •Re-gridding •Re-projection •Data format translation	•2D interpolation of scalar fields •Data fusion	•Mobile platforms & tracks •Textured topography •Video textures
Acoustic Propagation & Noise Fields	•Data transformations •Re-gridding •Re-projection	•Acoustic transmission modeling •Ambient noise modeling •Signal excess calculations	•Iso-surfaces •Cutting planes •Textured topography
Ocean Observatory Simulation	•Metadata cataloging •Data transformations •Re-gridding •Re-projection •Thematic aggregation	•DataStream filtering •Power and bandwidth calculations	•Flow fields •Point clouds •Iso-surfaces •Cutting planes •Time-series •Sensor network representations •Textured topography

Scenarios	Data Discovery, Access &Transformation	Data Analysis & Computation	Data Exploration & Visualization
Hydrology, Soils and Vegetation Simulation	•Metadata cataloging •Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation	•Stream chemistry •Soil transport •Water supply from snow water equivalent •Climate change scenarios	•Flow fields •Stream networks •Flow levels •Time-series •Data displacement maps •Textured topography
Data Archives	•Metadata cataloging •Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation	•Time-series analysis •Spatial analysis •Interpolation for missing data	•Flow fields •Point clouds •Iso-surfaces •Cutting planes •Time-series •Sensor representations •Textured topography
Seismic Activity on Tectonic Plates	•Metadata cataloging •Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation	•Detection, classification and localization of earthquakes •Time-series analysis	•Earthquakes •Time-series •Sub-seafloor cross-sections •Sensor networks •Textured topography
Ecosystem Modeling & Dynamic Information Framework	•Metadata cataloging •Collection indexing •Data calibration •Data transformations •Re-gridding •Re-projection •Thematic aggregation •Data format translation	•Earth system models •Simulation resource planning •Data coverage and interpolation	•Flow fields •Point clouds •Iso-surfaces •Cutting planes •Time-series •Textured topography •Data coverage •Data graphs and plots

VITA

Keith Grochow received his Bachelor of Science degree in Computer Science from the University of Washington in 1990. He spent the next several years working in the high tech industry before returning to study Computer Science at the University of Washington where he completed a Master of Science in 2007 and a Doctor of Philosophy in 2011. In addition to the research fields covered in this thesis, he spent several years investigating computer analysis of human motion and has several publications in this area.