End-to-End eScience: Integrating Workflow, Query, Visualization, and Provenance at an Ocean Observatory

Bill Howe[1] Peter Lawson[2] Renee Bellinger[3] Erik Anderson[4]
Emanuele Santos[4] Juliana Freire[4] Carlos Scheidegger[4] António Baptista[1] Cláudio Silva[4]

[1]Oregon Health & Science University, Center for Coastal Margin Observation and Prediction
[2]National Marine Fisheries Service, Northwest Fisheries Science Center
[3]Oregon State University, Hatfield Marine Science Center
[4]University of Utah, Scientific Computing and Imaging Institute

Abstract

Data analysis tasks at an Ocean Observatory require integrative and and domain-specialized use of database, workflow, visualization systems. We describe a platform to support these tasks developed as part of the cyberinfrastructure at the NSF Science and Technology Center for Coastal Margin Observation and Prediction integrating a provenance-aware workflow system, 3D visualization, and a remote query engine for large-scale ocean circulation models. We show how these disparate tools complement each other and give examples of real scientific insights delivered by the integrated system. We conclude that data management solutions for eScience require this kind of holistic, integrative approach, explain how our approach may be generalized, and recommend a broader, application-oriented research agenda to explore relevant architectures.

1 Introduction

Computing has led to an information explosion in all fields of science. Scientific inquiry is now regularly conducted *in silico* using complex computational procedures assembled from loosely-coupled resources, specialized libraries, and services on the grid and on the web. The heterogeneity of data sources, analysis techniques, data products, and user communities make it difficult to design a system that is flexible enough to accommodate broad requirements but specialized enough to be of daily use to scientists, policy makers, students, and the general public.

Databases, workflow systems, and visualization tools are collectively crucial but individually incomplete. Databases provide algebraic optimization and physical data independence, but offer poor support for complex data types (meshes, multidimensional arrays) and are change-intolerant. Workflow systems are very flexible, but even skilled programmers have trouble operating them effectively. Visualization tools are typically designed to efficiently "throw" data through the graphics pipeline, but offer little support for data integration and manipulation.

In this paper, we argue that typical data analysis tasks at an Ocean Observatory require techniques from all three tools, sometimes domain-specialized. To support these tasks, we describe a platform developed as part of a collaborative cyberinfrastructure at the Center for Coastal Margin Observation and Prediction (CMOP) [5] integrating a provenance-aware workflow system, 3D visualization capabilities, and a remote query engine for large-scale ocean circulation models, in addition to access routines for local files and web services. We conclude that data management solutions for eScience require this kind of holistic, integrative approach and recommend a broader, application-oriented research agenda for the community to study appropriate architectures.

Motivation: Understanding Coastal Margins. The Center for Coastal Margin Observation and Prediction (CMOP) is a multi-disciplinary institution with a mandate to transform ocean science research, education, and policy leveraging expertise in geochemistry, microbiology, oceanography, and computer science. A key enabler of the transformation is unfettered access to data. To this end, CMOP maintains the SATURN observatory: a network of heterogeneous observation platforms coupled with large-scale simulations of ocean circulation.

In this environment, cyberinfrastructure (the software, hardware, and data) serves as substrate and catalyst for effective collaboration between stakeholders: scientists, educators, students, policy makers, legislators, and the general public. *Data products*, consisting of data delivered through interactive visualizations, convey scientific messages and are the currency of communication in this multidisciplinary community. For example, a research paper can often be inaccessible to non-scientists, but a compelling, accurate, and well-documented data product can be understood, applied, and reused by nearly anyone. An important goal for computer scientists in this domain, then, is to provide tools that augment the ability of both experts and non-experts to create and organize data products.

At CMOP, there are two basic sources of data: simu-

978-0-7695-3535-7/08 $25.00 © 2008 IEEE
DOI 10.1109/eScience.2008.67

lations and observations. Simulation results are generated by two systems: a suite of daily forecasts targeting specific estuaries, and long-term *hindcast* databases, where the simulations are re-executed using observed data as inputs. The best hindcast databases predict oceanographic features with useful realism: the spatial, tidal, seasonal, and inter-annual variations of water level, currents and salinity; locations of estuary and plume fronts; and responses of the plume to wind shifts. The observational assets incorporate sensors for physical variables (temperature, salinity, velocity, irradiance, and optical properties) with sensors for chemical and biological variables (oxygen, nitrate, chlorophyll) into a variety of fixed and mobile platforms: permanent stations, vertically mobile profilers, research vessels, and, soon, autonomous underwater vehicles. In addition to in situ instruments and platforms managed by CMOP, we ingest data from third-party assets over the Internet, including remote sensing platforms such as satellites and shore-based high-frequency radar installations.

The ocean sciences, and environmental science overall, are progressing into the computational and informational stages of domain sciences [11] and are demanding tools to facilitate the shift. The hallmark of this progression is that the rate of data acquisition begins to outpace scientists' collective ability to analyze them. In response to this data avalanche, we argue that environmental science has become crucially dependent on the advancement and successful integration of three areas of CS research: workflow, databases, and visualization.

Workflow. Traditional ad-hoc data exploration using, e.g., Perl scripts, has serious limitations. Analysis naturally generates a variety of auxiliary digital artifacts (scripts, intermediate data files, log files, data products, notes) but offer no support for using them to streamline analysis, investigate problems, compare results, or other meta-analysis activities. Workflow systems with integrated provenance models have therefore grown in popularity within the scientific community [2, 8, 9, 19, 20, 22]. Workflow systems not only support the automation of repetitive tasks (the original focus), but they also systematically capture provenance information for derived data products [6].

Databases. Workflow systems provide a substrate in which to manipulate data, but do not provide logical and physical data independence — meaning that when the organization of the data changes, your program need not. Data independence is the core salient feature of relational database management systems (RDBMS), but their success with business data has not been transferred to the complex data types and complex operations of eScience. Separation between the logical and physical worlds opens the door to all other results from the database community—declarative query languages, transparent indexing, algebraic cost-based optimization, parallel query evaluation. Such techniques collectively allow the system to fit the computation to the data rather than fitting the data to the computation.

Operator	Description
bind	Associate data with an existing grid.
restrict	Cull cells that do not satisfy a predicate.
cross	"Multiply" one gridfield by another; analogous to set cross product. (written as the infix operator $\otimes$).
merge	Combine multiple gridfields over the intersection of their grids.
accrete	"Grow" a gridfield by incorporating neighboring cells.
regrid	Map one gridfield onto another, aggregating as needed.
fixpoint	Allow recursive execution of recipes.

Table 1. List of gridfield operators and their descriptions

Visualization. As databases scale up, so do the average size of query results. For example, a few facts about ocean state (the latest salinity and temperature measurements from a single buoy) become millions of facts (time-varying fields of temperature and salinity from a regional forecast). Thousands of pages of results for sequential browsing are not helpful: some form of aggregation or additional filtering must be applied before the returned data can be used to inform scientific decisions.

1.1 Data Model and Algebra

Visualization offers a different and powerful form of aggregation: millions of facts are rendered into a single (interactive) scene, harnessing the visual acuity of the human eye to convey complex scientific messages with fewer bytes. Visualization systems typically emphasize efficient manipulation of an individual dataset. However, this mode of operation does not scale with the rate of scientific data acquisition encountered in practice. We can no longer afford to move the data to the computation by downloading a file for desktop visualization. Instead, we must move the computation to the data, a hallmark capability provided by the query engines of database systems.

Contributions. In the context of an ocean observatory, we present a method of integrating workflow, visualization, and database-style query evaluation using the VisTrails platform [9, 26] — a provenance-enabled workflow and visualization system — augmented with remote access to a server powered by GridFields — an algebraic language for manipulating simulation results in the physical sciences. Specifically, we show how the visual programming, change-based provenance, cache-oriented evaluation strategy, and built-in visualization features of the VisTrails workflow system can be integrated with algebraic optimization and remote pro-

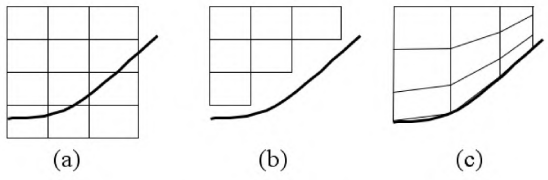

Figure 1. The CMOP grid evolved over time, requiring significant changes to the analysis pipeline. The bold line indicates the bathymetry of the river. (a) The original CMOP grid extended below the bathymetry. (b) To save space on disk, these invalid values were culled, complicating the file format. (c) A newer model uses a σ grid, meaning the depth levels follow the contours of the bathymetry.

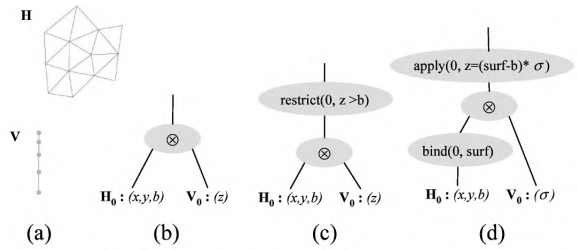

Figure 2. Each CMOP grid can be expressed succinctly as a gridfield expression that highlights the structural differences. (a) illustrates the form of the horizontal and vertical grids. Plans (b), (c), and (d) construct 3D analogs of the 2D grids in Figure 1, respectively.

cessing provided by GridFields. Further, we demonstrate how real problems were solved using this platform, and show that these problems were crucially dependent on all three components.

2 Querying Simulation Results: GridFields

Analysis tasks at CMOP typically involve both simulation output and observed sensor measurements. While sensor measurements usually exhibit a tabular structure suitable for storage and manipulation using a relational database, the simulation results require a different approach, for several reasons: First, the fundamental data model is one of 3D mesh (not tables) making common queries (e.g., interpolation and regridding) difficult to express in SQL. Second, explicit foreign keys can triple the storage required for these array-like data structures. Third, these results are never updated, wasting the overhead incurred by the transactional guarantees that databases are known for.

In previous work, we addressed these problems by developing an algebra of *gridfields* for manipulating the results of simulations in the physical sciences, especially simulations involving unstructured grids [15]. Although unstructured grids can be modeled directly as a collection of polyhedra [13], binary representations of meshes provided greater flexibility and better performance [14, 15].

2.1 Data Model and Algebra

The fundamental data structure of the algebra is the gridfield. A gridfield is a pair (G, F), where G is a *grid* and F is a *field*. A grid is constructed from sets of *cells* of various dimension connected by an incidence relationship. We extend the concept of k-simplices to cell structures, defining k-cell a cell of dimension k [4]. These geometric interpretations of cells guide intuition, but a grid does not explicitly indicate its cells' geometry.

Instead, geometry and other information is captured in one or more *attributes*: functions (represented as arrays) mapping cells to data values. Attributes may be associated with cells of any dimension. For example, attributes named x and y may be associated with 0-cells to represent geometric information, while an attribute named *area* may be associated with the 2-cells to represent the area of the geometric polygon associated with the abstract, topological cell.

The core algebra consists of six operators to manipulate the gridfield data structure. Some gridfield operators are analogous to relational operators, but grid-enabled. For example, the restrict operator filters a gridfield by removing cells whose bound data values do not satisfy a predicate. However, restrict also ensures that the output grid retains certain properties. Other operators are novel, such as regrid. The regrid operator maps data from one gridfield to another and then aggregates mapped data to produce one value per cell.

The operators in the core algebra are shown in Table 1. Compositions of these operators (*recipes*) express constructions of complicated gridfields from primitive components or act as queries to extract smaller, simpler gridfields suitable for direct rendering[1].

For example, Figure 1 illustrates the evolution of the grid structure used in the CMOP observatory [1] over different versions of the simulation program, simplified to two dimensions for clarity. The earliest grid had the form of Figure 1(a). This grid extends below the bathymetry; a distinguished value (-99) is stored for these underground positions to mark them invalid.

Each of these grids, in their full 3D form, can be succinctly expressed as gridfield expressions, as in Figure 2. Figure 2(a) is an illustration of the 2D horizontal grid H and the 1D vertical V. These cross product of these two grids expresses a complete 3D grid. The recipes in Figure 2 (b), (c), and (d) produce the 3D analogs of the 2D grids in Figure 1 (a), (b), and (c), respectively. The fact that these diverse situations can be expressed uniformly with GridFields demonstrates improved physical data independence. These

[1] We avoid the term query plan to emphasize that these expressions are written by the programmer rather than generated by the system.

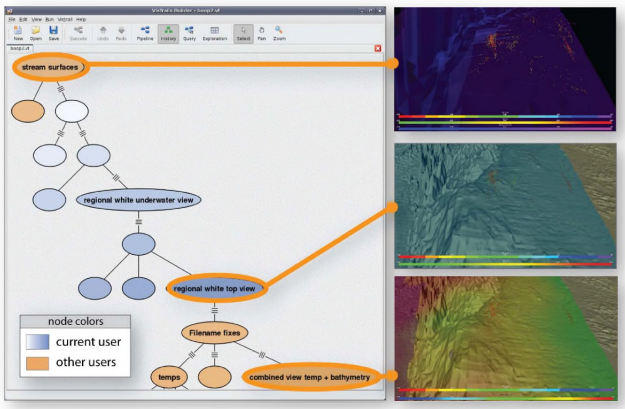

Figure 3. Users collaborate to generate visualizations. VisTrails captures all adjustments made to a workflow, producing provenance history that represents the workflow's evolution. Derived visualizations at each step are shown at right.

recipes can capture the logical and physical change in the grid, insulating downstream programs.

2.2 Execution Model

Programmers build recipes by calling operator constructors with the appropriate parameters and composing them. The core data structures and operators are written in C++, but we provide a Python interface to facilitate recipe manipulation and rapid development.

A gridfield recipe can be evaluated in two ways: *query mode* and *interactive mode*. In query mode, applications construct a tree of operators and submit the tree to the gridfield server for evaluation using the fetch operator. The server is free to transform the recipe prior to and during evaluation. Once evaluated, the results are shipped back to the application.

In interactive mode, the application may request the result of any operator in the tree at any time. That is, the programmer can construct a recipe to, say, display the average salinity for Monday, then repeatedly reparametrize the recipe to see the results for Tuesday, Wednesday, and Thursday.

The term "mode" suggests that the programmer explicitly declares which mode he or she will be using. In fact, the situation is simple: execution proceeds in interactive mode except in the following circumstances: (1) The programmer *optimizes* a recipe, or (2) the recipe root is a fetch operator.

To optimize a recipe, the programmer calls the optimize function, passing the root operator of the recipe tree as the only argument. This function analyzes the recipe for opportunities for algebraic rewrites that result in a less expensive recipe. For example, a very common rewrite borrowed from relational query optimization is to commute restrict operators down the tree as far as possible. Since the restrict operator decreases the size of the gridfield, we want to evaluate it as early as possible to lower the memory footprint and reduce downstream work.

The return value from this function is a recipe different from the input in two ways. First, the optimize function may have applied algebraic optimizations, and second, the operators, when evaluated, may destructively update previous results. The optimized recipe is still composed of operators — as opposed to having been compiled to an object language — but it is considered opaque to the programmer.

The fetch operator is used in a recipe as any other operator. A logical noop, the fetch operator accepts an input operator tree and a url. When the operator is evaluated, it marshals the operator tree and ships it to a remote server for evaluation. The gridfield server receives the message, compiles the recipe, evaluates it, and ships the result back to the caller in a packed binary format. Any other fetch operators in the transmitted recipe are evaluated similarly, with the initial server acting as the client. Simple distributed evaluation of gridfield recipes is therefore possible. The fetch operator also allows simple distributed optimization: operator instances can be injected into a recipe automatically to balance load between client and server.

3 Provenance, Workflow, Viz: VisTrails

To integrate and visualize data, complex pipelines combining multiple data sources, libraries and visualization tools need to be assembled. We achieved this by employing the VisTrails workflow management system. We chose VisTrails as our scientific workflow platform for several reasons. VisTrails uniquely integrates features of both workflow and visualization systems. The Visualization Toolkit is available as a core package, providing sophisticated 3D algorithms to workflow programmers. Complex 3D scenes can be constructed individually but analyzed collectively using a spreadsheet metaphor — each cell contains a separate visualization. Further, a series of related visualizations can be expressed in one operation using *parameter explorations*. Both features help users quickly navigate the enormous parameter space involved in 3D visualization — a critical challenge when entraining new users such as fisheries biologists. The system transparently captures all the steps in the pipeline design process: the provenance for both data products and analysis pipelines. Similar to a source code version control system, users can make modifications to analysis pipelines and check in their changes to a central repository. This repository becomes a sharing point, allowing others to check out (and merge) these changes. VisTrails also provides intuitive user interfaces that allow users to explore the information in the repository, including a visual difference interface that allows users to compare different pipeline versions side by side [9]; a mechanism for querying pipelines by example; and the ability to refine pipelines

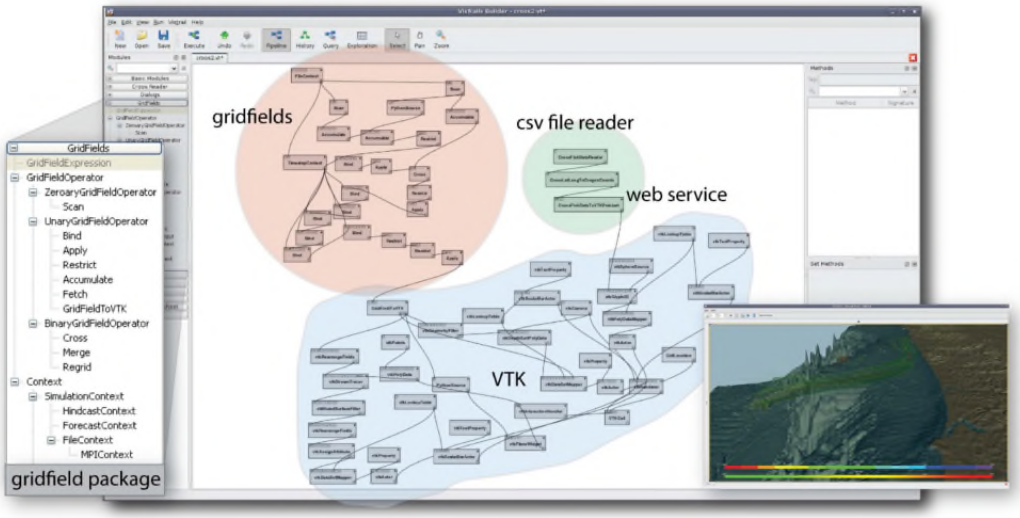

Figure 4. VisTrails is used to integrate the gridfields with different tools. The modules are grouped by kind. On the left, a list with the gridfield operators implemented as modules. The visualization generated by this workflow is shown on the right.

by analogy [24]. Figure 3 shows a version tree with different pipeline versions that were created in the process of analyzing CMOP data. Not only does this tree allow us to navigate through all the different versions of the pipelines, but it also simplifies collaborative efforts.

We integrated the GridField library into VisTrails as a package using the plug-in mechanism. Figure 4 illustrates a workflow that executes a gridfield expression, extracts data from a local source, requests a web service to convert the data to the gridfield's coordinate system and uses VTK to generate 3D visualizations. The corresponding visualization is shown on the right. The GridField package provides access to the operators as VisTrails modules, as shown in the left-hand panel of Figure 4.

The natural execution model for GridField operators in a workflow system is interactive mode (Section 2.2) — each operator is executed sequentially. However, this naive integration of the two systems precludes remote execution, algebraic optimization, and efficient destructive updates. Instead, GridField modules are evaluated lazily, incrementally constructing a tree of closures rather than eagerly manipulating actual data. The expression is only evaluated when 1) the result is needed by another package (e.g., the Visualization Toolkit [17] for on-screen rendering), or 2) a fetch operator is encountered, in which case the expression is shipped to a server for evaluation.

This strategy delivers a form of logical and physical data independence for workflow systems: VisTrails is used to manage "workflow crafting" tasks by providing provenance tracking and querying features, a rich visual interface, the visualization spreadsheet, parameter explorations, and more. However, the GridField system remains free to evaluate expressions however it sees fit: it can apply algebraic optimizations, ship sub-expressions to a remote server, and destructively update data structures in memory as necessary. Further, lazy evaluation provides fine-grained control over the VisTrails cache, which stores intermediate results to reduce unnecessary re-computation [3].

Figure 4 shows a workflow that executes a gridfield expression, extracts data from a local source, requests a web service to convert the data to the gridfield's coordinate system and uses VTK to generate 3D visualizations. The visualization generated by this workflow is shown on the right.

Using VisTrails and the GridField Algebra, we developed two applications involving remote access to CMOP simulations and observational data.

4 Examples

Using VisTrails and the GridField Algebra to combine simulation and observational data, we were able to help fisheries biologists and ocean modelers realize new scientific insight from existing datasets. For the modelers, we provided a a 3D analysis tool for simulation results intended to augment or replace the 2D analysis procedures employed in the status quo. For the fisheries biologists working on the Collaborative Research on Oregon Ocean Salmon (CROOS) project [**?**], we integrated fisheries data with oceanographic observations and CMOP model results to link physical variables to the ocean ecology of fish populations in the Pacific Northwest.

Example: Simulation Skill Assessment. There are several potential benefits of 3D visualization in this domain: The physics of complex physical processes can be directly experienced in real-time rather than interpreted indirectly from 2D plots. For example, the shape of the freshwater plume

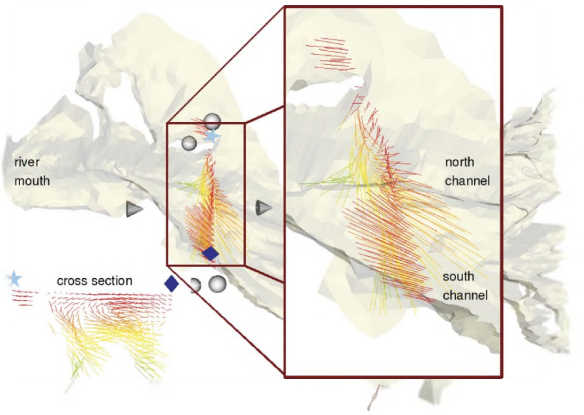

Figure 5. The source of the upstream salt flux in the River estuary model was elusive using 2D methods. A plane swept through the 3D field interactively instantly exposed the relevant region instantly: the tidal flats, in between the two river channels.

can be an indicator of the skill of an ocean circulation simulation. To ascertain shape in 2D, an ensemble of indirect metrics must be calculated: plume volume, surface area, the presence or absence of sharp fronts, etc. However, replaying an animation of the development of the plume bubble immediately conveys the accuracy of the simulation to a trained oceanographer.

On the opposite side of the estuary-ocean interface, the salinity intrusion into the estuary is called the *salt wedge* and is a difficult feature to model accurately due to its nuanced and highly non-linear dependence on bathymetry, river discharge, and tidal influences.

Salinity intrusion length as computed by the model was observed to be shorter than suggested by observation. The typical procedure for investigating the cause is to visualize 2D slices of the salt flux (i.e., salinity concentration multiplied by velocity). Locations from which to extract a slice are found by trial and error, informed by domain expertise. As an alternative, we developed an interactive 3D tool that calculates salt flux and allows the user to sweep a plane through the field arbitrarily, directly visualizing the salt flux vectors as barbs. The answer to the question was exposed immediately: the salt flux was lower than expected at the bottom of the river channels, but higher than expected in the tidal flats between the channels.

In Figure 5, the Columbia River Estuary is bounded on the left (West) by the ocean interface (the ocean itself is omitted for clarity). The cutting plane is oriented vertically, parallel to the left edge of the page and extruded down into the page. In the lower left inset, the reader is looking down the mouth of the river and can see cross section defined by the plane. The barbs represent residual salt flux vectors, colored by depth. The salt flux vectors are tidally-averaged over a spring neap cycle from a two-week period in 1999. Darker barbs are near the surface of the water and lighter barbs in the middle and bottom. The star and diamond align positions in the cross section with positions in the background: star indicates north channel, diamond indicates south channel. The inset at right details tidal flats between the two channels. We can see darker surface barbs pointing downstream (towards left) corresponding to the direction of river flow. We can also see lighter near-bottom barbs pointing upstream (towards right). These features indicate that salt is flowing out at the surface and in at the bottom, which is expected considering the higher density of brackish water. What is unexpected is that the salt flux is of greater magnitude in the tidal flats between the channels than it is in the channels themselves. This feature was immediately exposed by sweeping the plane from left to right against the river's flow and watching the barbs change magnitude and direction.

The shift from static 2D to interactive 3D provided instant insight into the problem, even though the fundamental visualization technique was similar: arrow glyphs representing vectors on a plane.

Example: Fisheries Oceanography The continental shelf of Western North America is one of the most productive fisheries in the world. Yet, the relationship between the ocean environment and fish distribution, growth, and survival is understood only in general terms. Increasingly, the focus is on understanding the fine-scale mechanisms of fish ecology in the ocean. Project CROOS was founded on the idea that fishermen, and commercial salmon trollers in particular, could collect sample data during the normal course of fishing. Since 2006 they have been collecting GPS tracklog of fishing activity and location and depth of each Chinook salmon caught, along with fin clips and scales for genetic stock identification (GSI) and aging. The result is a rich data set with high spatial and temporal specificity presenting unique challenges for data visualization and analysis.

The 3D visualization of fishery data combined with CMOP simulations of oceanographic conditions helps explore relationships between ocean conditions and fish catch distributions. We constructed a 3D scene by overlaying fish catch positions from September and October 2006 colored by stock of origin with streamlines of ocean current from the simulation data fetched using gridfields.

Figure 6 is a view from beneath the ocean surface looking onshore from the shelf break, revealing the depth distribution of the fish. Inspection reveals fish throughout the water column, including some at the surface, signifying an absence of depth data, and some fish beneath the sea floor (not visible in this view), clearly out of bounds. Adult Chinook salmon are usually associated with the sea floor unless environmental conditions dictate otherwise. Here we see there is a well-defined maximum depth for the fish that

is some distance above the sea floor. During the time period these data were collected there were unusual anoxic conditions near the sea floor. This visualization suggests that we explore the relationship between dissolved oxygen and Chinook vertical distribution. The anoxic "dead zone" is a recent feature of the nearshore ocean that is likely caused by new wind patterns resulting from global warming. Aided by these 3D visualizations, we expect to link warming, winds, dead zones, and fine-scale Chinook salmon distributions, gaining insights into the ocean ecology of these fish.

There are many other species of fish and invertebrates that can be sampled at fine spatial scales using adaptations of the techniques developed by Project CROOS. By combining this information with expanded simulations, expanded ocean observation, and the analytic power of 3D visualization we hope to "see" the ocean ecosystem in fundamentally transformative ways.

5 Related Work

Ocean observatories have received increased attention in recent years (c.f., [10]). One tool developed in this domain is the Trident Scientific Workflow Workbench for Oceanography [2], which provides a collection of workflow *activities* for processing oceanographic data from netCDF files, relational databases, and web services. Trident interoperates closely with the COVE oceanographic visualization system [12]. We distinguish our approach by a focus on unstructured grids, a data type for which no standard file formats, database systems, or exchange protocols exist. Further, change-based provenance leads to a variety of novel features [24, 26].

Other workflow systems separate expression from execution in Grid environments [7, 20], but none support algebraic optimization or allow destructive sharing of intermediate results. These features require a formal data model in which safe rewrite rules can be defined. The GridField system provides such a data model for a broad class of scientific data types.

Gray managed unstructured grids from finite element simulations in materials science with Microsoft's flagship SQL Server database product coupled to IBM's OpenDX visualization platform [13]. Indexes provide efficient access to data subsets, and OpenDX renders the results into a manipulable scene allowing inspection of non-trivial simulation features such as crack propagation. However, oceanographic simulation results often exhibit an array-like structure that is difficult to model efficiently in a relational database [15].

A different approach to distributed visualization is to provide access to the virtual desktop on a remote computing system [16, 18, 23, 25]. Here the data exist entirely on the server and only images or graphics primitives are transmitted across the network. Other applications, such as VisIt [18] and ParaView [21] provide a scalable visualization and rendering back-end that sends images to a remote client.

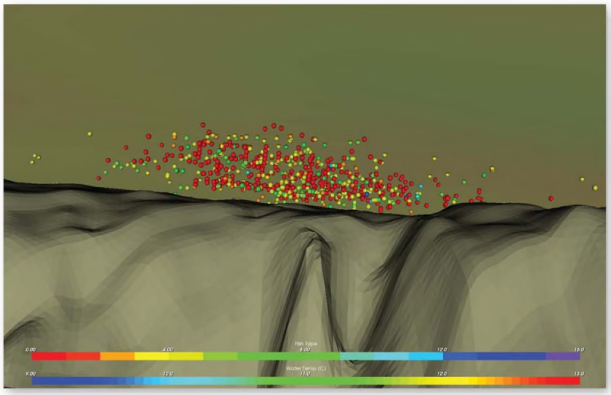

Figure 6. View from beneath the ocean surface looking onshore from the shelf break, revealing the depth distribution of the fish. Dots represent individual salmon colored by stock of origin; water is colored by temperature.

Several systems have coupled visualization systems to relational databases. The Polaris project [27] provides a language for specifying tables of small displays based on the output of relational queries, especially Online Analytical Processing (OLAP) operations. These tables, however, do not provide 3D visualization.

6 Conclusions and Future Work

We have demonstrated how a database-style remote query engine optimized for scientific data can be integrated with a provenance-aware, visualization-capable workflow system to deliver new insights to scientists in a variety of domains. Specifically, the VisTrails system provides a collaboration vector for users with diverse backgrounds — ocean modelers, fisheries biologists, computer scientists, and the general public. However, workflow systems remain difficult to program due to physical data dependence — the workflow itself is responsible for every detail of data access and execution. By borrowing techniques from the database community, we have shown that it is possible to raise the level of abstraction by allowing the data management subsystem the freedom to optimize and evaluate workflows independently of the workflow user interface.

There are several directions we intend to pursue in future work. To further improve sharing and re-use and entrain non-experts to use and even create new workflows, we are currently exploring an interface that allows the creation of simplified pipeline abstractions that can be rendered as familiar web interfaces. To improve integration with remote query systems, we are investigating techniques that can intelligently exploit local and remote caches and reoptimize accordingly.

7 Acknowledgments

We would like to thank PaCOOS West Coast Habitat Server for providing the bathymetry dataset used on the CROOS project. Our research has been funded by the Department of Energy SciDAC (VACET and SDM centers), the National Science Foundation (grants IIS-0746500, CNS-0751152, IIS-0713637, OCE-0424602, IIS-0534628, CNS-0514485, IIS-0513692, CNS-0524096, CCF-0401498, OISE-0405402, CCF-0528201, CNS-0551724), the National Oceanic and Atmospheric Administration (NA07NOS4730203), and IBM Faculty Awards (2005, 2006, 2007, and 2008). E. Santos is partially supported by a CAPES/Fulbright fellowship.

References

[1] A. M. Baptista. Corie: the first decade of a coastal-margin collaborative observatory. In *Proc. of MTS/IEEE Conference and Exhibition for Ocean Engineering, Science and Technology*, Boston, MA, 2006. IEEE Press.

[2] R. Barga, J. Jackson, N. Araujo, D. Guo, N. Gautam, K. Grochow, and E. Lazowska. Trident: Scientific workflow workbench for oceanography. In *IEEE Congress on Services - Part I*, pages 465–466, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[3] L. Bavoil, S. Callahan, P. Crossno, J. Freire, C. Scheidegger, C. Silva, and H. Vo. Vistrails: Enabling interactive multiple-view visualizations. In *Proceedings of IEEE Visualization*, pages 135–142, 2005.

[4] G. Berti. *Generic software components for Scientific Computing*. PhD thesis, Faculty of mathematics, computer science, and natural science, BTU Cottbus, Germany, 2000.

[5] NSF Center for Coastal Margin Observation and Prediction (CMOP). http://www.stccmop.org.

[6] S. B. Davidson and J. Freire. Provenance and scientific workflows: challenges and opportunities. In *Proceedings of ACM SIGMOD*, pages 1345–1350, 2008.

[7] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. B. Berriman, J. Good, A. Laity, J. C. Jacob, and D. S. Katz. Pegasus: a Framework for Mapping Complex Scientific Workflows onto Distributed Systems. *Scientific Programming Journal*, 13(3):219–237, 2005.

[8] I. Foster, J. Voeckler, M. Wilde, and Y. Zhao. Chimera: A virtual data system for representing, querying and automating data derivation. In *Proceedings of IEEE SSDBM*, pages 37–46, 2002.

[9] J. Freire, C. T. Silva, S. P. Callahan, E. Santos, C. E. Scheidegger, and H. T. Vo. Managing rapidly pidly-evolving scientific workflows. In *International Provenance and Annotation Workshop (IPAW)*, LNCS 4145, pages 10–18, 2006.

[10] B. Gegosian. Ocean observing initiative, 2005. http://www.oceanleadership.org/ocean_observing.

[11] J. Gray, D. T. Liu, M. A. Nieto-Santisteban, A. S. Szalay, G. Heber, and D. DeWitt. Scientific data management in the coming decade. Technical report, Microsoft MSR-TR-2005-10, January 2005.

[12] K. Grochow, M. Stoermer, D. Kelley, and E. Lazowska. Cove: A visual environment for ocean observatory design. *Journal of Physics*, 125(012092), 2008.

[13] G. Heber and J. Gray. Supporting finite element analysis with a relational database backend; part 1: There is life beyond files. Technical report, Microsoft MSR-TR-2005-49, April 2005.

[14] B. Howe. *GridFields: Model-Driven Data Transformation in the Physical Sciences*. PhD thesis, Department of Computer Science, Portland State University, 2007.

[15] B. Howe and D. Maier. Algebraic manipulation of scientific datasets. *VLDB Journal*, 14(4):397–416, 2005.

[16] IBM Systems and Technology Group. IBM Deep Computing. Technical report, IBM, 2005.

[17] Kitware. The Visualization Toolkit (VTK) and Paraview. http://www.kitware.com.

[18] Lawrence Livermore National Laboratory. VisIt: Visualize It in Parallel Visualization Application. https://wci.llnl.gov/codes/visit [29 March 2008].

[19] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger-Frank, M. Jones, E. Lee, J. Tao, and Y. Zhao. Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice & Experience*, 18(10):1039–1065, 2006.

[20] T. Oinn, M. Greenwood, M. Addis, M. N. Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P. Lord, M. R. Pocock, M. Senger, R. Stevens, A. Wipat, and C. Wroe. Taverna: lessons in creating a workflow environment for the life sciences. *Concurrency and Computation: Practice & Experience*, 18(10):1067–1100, 2006.

[21] Paraview. http://www.paraview.org [29 March 2008].

[22] S. G. Parker and C. R. Johnson. SCIRun: a scientific programming environment for computational steering. In *Supercomputing*, page 52, 1995.

[23] B. Paul, S. Ahern, E. W. Bethel, E. Brugger, R. Cook, J. Daniel, K. Lewis, J. Owen, and D. Southard. Chromium Renderserver: Scalable and Open Remote Rendering Infrastructure. *IEEE Transactions on Visualization and Computer Graphics*, 14(3), May/June 2008. LBNL-63693.

[24] C. E. Scheidegger, H. T. Vo, D. Koop, J. Freire, and C. T. Silva. Querying and creating visualizations by analogy. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1560–1567, 2007.

[25] Silicon Graphics Inc. OpenGL vizserver. http://www.sgi.com/products/software/vizserver.

[26] C. Silva, J. Freire, and S. P. Callahan. Provenance for visualizations: Reproducibility and beyond. *IEEE Computing in Science & Engineering*, 6(4):12–19, 2007.

[27] C. Stolte, D. Tang, and P. Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65, 2002.