Journal of Physics: Conference Series **125** (2008) 012100 doi:10.1088/1742-6596/125/1/012100

Software Infrastructure for exploratory visualization and data analysis: Past, present, and future

Cláudio T. Silva[1,2] and Juliana Freire[2]

[1]Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT 84112, USA

[2]School of Computing, University of Utah, Salt Lake City, UT 84112, USA

E-mail: csilva@sci.utah.edu, juliana@cs.utah.edu

Abstract. Future advances in science depend on our ability to comprehend the vast amounts of data being produced and acquired, and scientific visualization is a key enabling technology in this endeavor. We posit that visualization should be better integrated with the data exploration process instead of being done after the fact – when all the science is done – simply to generate presentations of the findings. An important barrier to a wider adoption of visualization is complexity: the design of effective visualizations is a complex, multistage process that requires deep understanding of existing techniques, and how they relate to human cognition. We envision visualization software tools evolving into "scientific discovery" environments that support the creative tasks in the discovery pipeline, from data acquisition and simulation to hypothesis testing and evaluation, and that enable the publication of results that can be reproduced and verified.

1. Introduction

As scientists need to deal with large volumes of heterogeneous data, such as medical MRI and CT scans, water salinity measurements, engine temperature measurements, and results of computational experiments, there is an increasing need for techniques and usable tools to help them analyze and make sense out of these data. Although the term "visualization" is somewhat recent, generally accepted to having been coined for the 1987 NSF report on scientific visualization, visualization has been used as a means of "data understanding by visual representation or other visual means" for hundreds of years. What separates the old from the new is the availability of advanced computing capabilities, including modern computer graphics techniques, which form the backbone of modern visualization research.

While computer graphics techniques are an integral and important part of visualization, it is important to contrast the two fields. In the more traditional computer graphics, used to make movies or games, the goal is to produce visually engaging (beautiful) imagery that "appears plausible." In visualization, *accuracy* is more important than aesthetics. As such, users must be aware of errors and uncertainty present in the visualizations so that they are not misled by the resulting images [1]. Furthermore, the process of extracting insight from data goes beyond the generation of imagery and needs to encompass the complete scientific discovery pipeline [2–4].

Several visualization techniques have been developed that greatly improve our ability to comprehend large amounts of complex data. Figure 1 shows two images of a CT tooth dataset that clearly exemplify the the capabilities of advanced visualization techniques.. On the left, we

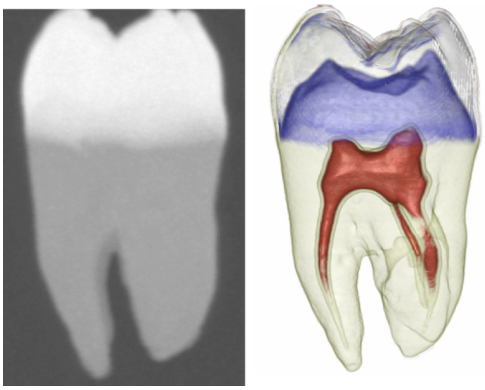


Figure 1. Maximum intensity projection (MIP) volume rendering (left) versus full direct volume rendering (right). Image courtesy of the Scientific Computing and Imaging (SCI) Institute.

show a maximum intensity projection (MIP) image, of the type we could compute in the early 1980s. On the right, we display a full volume rendering that uses multidimensional transfer functions [5]. While in the MIP visualization, we are limited to seeing overall density changes, using state-of-the art volume rendering, we are able to precisely extract the different tooth structures, and study them in detail.

In spite of the benefits of visualization, currently visualization is not well integrated with the scientific process. In this paper, we discuss some of the barriers to a wider adoption of visualization techniques in science and new research directions that aim to close this gap. One issue is the lack of communication between the visualization community and the potential users in different scientific areas. Scientists are often unaware of the state of the art and often use older techniques that may suffer from deficiencies (e.g., in performance or visualization quality). We need to make visualization techniques more widely available. We also need tools that integrate these techniques and that guide users through the process of creating insightful visualizations. To paraphrase Don Norman: we should design tools that make us smarter [6].

Presentation versus exploratory visualization. Most contemporary scientists often see the best visualizations as finished products, for instance, in the form of short videos that present detailed explanations of some complex phenomena. To this end, web sites such as `www.scivee.tv`, have been created as a forum for scientists to present their research by contributing videos. One of the earliest, and still best, examples of a visual presentation using advanced visualization techniques is the National Center for Supercomputing Application (NCSA) classic "Study of a Numerically Modeled Severe Storm." This short video presents the physics of a severe storm in a clear and concise manner. It is divided into several segments, each of which builds up intuition for the underlying characteristics of the simulation. After introducing many of the important aspects, the movie ends by present a complex visual representation of the overall phenomena by using many visualization techniques at the same time. This movie provides a very good example of the use of visualization for summarizing and presenting scientific results. It was highly influential, and it is routinely used as a case study. For instance, Tufte [7] writes an excellent critic of this classic and proposes a number of improvements to the presentation.

While this use of visualization techniques for the presentation of complex technical results is extremely important, in our view the area can play an even more important role in the sciences if it is integrated into the scientific progress. Many scientists take the view that visualization is what is done "after the fact"—when all the science is done—to generate presentations of the findings. However, we believe the real benefits of visualization become more pronounced when used as a part of the data exploration process.

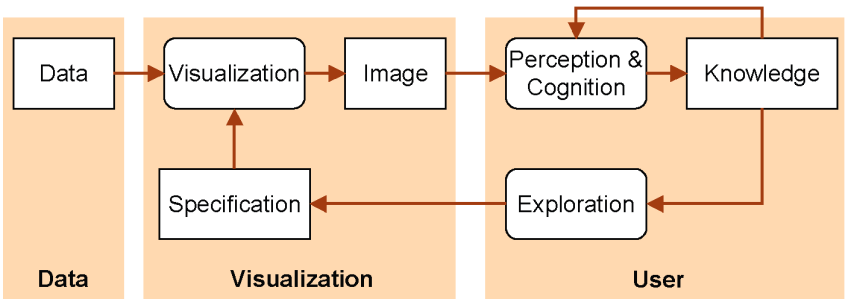


Figure 2. A model for exploratory visualization. Adapted from J. van Wijk [2].

We take the view that future advances in science depend on the ability to comprehend the vast amounts of data being produced and acquired. Visualization is a key enabling technology in this endeavor: it helps people explore and explain data through software systems that provide a static or interactive visual representation. A basic premise of visualization is that visual information can be processed at a much higher rate than raw numbers and texr – as the cliché goes: "A picture is worth a thousand words."

Despite the promise that visualization can serve as an effective enabler of advances in other disciplines, the application of visualization technology is non-trivial. The design of effective visualizations is a complex process that requires deep understanding of existing techniques, and how they relate to human cognition. Although there have been enormous advances in the area, the use of advanced visualization techniques is still limited.

Challenges in exploring data through visualization. Data exploration through visualization requires scientists to go through several steps. To successfully analyze and validate various hypotheses, one must pose several queries, correlate disparate data, and create insightful visualizations of both the simulated processes and observed phenomena.

As illustrated in figure 2, a scientist needs to assemble and execute complex workflows that consist of data set selection, specification of series of operations that need to be applied to the data, and the creation of appropriate visual representations, before finally viewing and analyzing the results. Often, insight comes from comparing the results of multiple visualizations created during the exploration process, for example, by applying a given visualization process to multiple datasets generated in different simulations, by varying the values of certain visualization parameters, or by applying different variations of a given process (e.g., which use different visualization algorithms) to a dataset.

The challenge of this exploration process is that for a visualization to be insightful, it needs to be both effective and efficient. This requires a combination of art, technology, and science to reveal information that is otherwise obscured. Assessing the value of an insight or measuring how much insight is required in the analysis of the data is a difficult task because the scientist may not even know what aspects or features the data contains [8]. Therefore, visualization can be regarded as an open-ended problem with no single answer.

The process is further complicated when viewed in the larger context of reproducible science. In particular, scientists and engineers need to expend substantial effort managing data (e.g., scripts that encode computational tasks, raw data, data products, images, and notes) to ensure that the visualizations are reproducible by themselves as well as others in the scientific community.

Despite the enormous progress in scientific visualization research, existing tools do not support the analytical reasoning process that scientists use in their work. There is little or no support for linking data, visualizations, and knowledge. For instance, as insight is generated

over time, the findings are not linked to supporting evidence, and scientific publications to a large extend stand on their own with little hard evidence of the scientific facts.

2. Incomplete history of scientific visualization tools

To see where we stand today in terms of visualization and data analysis tools for scientific discovery, it is useful to take a look at some of the tools that have been developed by the visualization community over the past two decades.

The Application Visualization System (AVS) [9] was one of the earliest and most influential visualization environments developed in the 1980s. It was based on a dataflow model and aimed to provide an easy-to-use, powerful system for supporting the filter/map/render pipeline. The IBM Data Explorer (DX) [10] and the IRIS Explorer are two other systems from the same period. Each system had its own innovations; e.g., IBM DX had a very flexible data model. Testimony to their effectiveness, these dataflow-based visualizations systems are still widely used today, over 20 years since they were originally developed; and they can be seen as precursors of current scientific workflow systems.

A dataflow is a directed graph where nodes represent computations and edges represent streams of data: each node or module corresponds to a procedure that is applied on the input data and generates some output as a result. The flow of data in the graph determines the order in which the processing nodes are executed. In the dataflow model, a change in one variable forces the recalculation of other connected variables (similar to inter-dependent cells in a spreadsheet). Furthermore, computation is stateless: the output of a module is functionally determined by the module and its inputs. In visualization, it is common to refer to a dataflow network as a "visualization pipeline." The actual computation is performed by the modules by operating on the input data. It is sometimes useful to think of the modules as performing different parts of the computation. Common types include modules that generate data (sources), consume data (sinks), and operate on data by applying algorithms to the data (processes). In their most general form, each module has a set of inputs, a set of outputs, a set of parameters, and, in certain systems, some form of GUI that can be used for interacting with the modules. Tthese inputs and outputs are often referred to as "ports." It is useful to enforce "types" in these systems, allowing connections only between ports of compatible types.

A "visualization pipeline" provides a means to create visual representations of raw data. It is fairly common to break up (at least, conceptually) the visualization pipeline into three main phases that each data item needs to pass through to be visualized: *filter*, *map*, and *render*. In practice, there might be steps that need to happen before (e.g., reading the data from disk and uncompressing it) or after (e.g., image delivery). Also, depending on the exact visualization algorithm being applied, one can think of "skipping" a step. This "3-stage pipeline" is a bit too simplistic but, despite its potential faults, serves as the basis of a general framework for setting up visualization pipelines.

The goal of the filtering phase is to prepare data for visualization. It takes the raw simulated or sensored data and transforms it into another form that is more informative or less voluminous. It might also generate auxiliary information that is not readily available but is necessary for later processing steps. For instance, it might do data resampling, interpolation, or gradient calculation. The mapping phase will turn the filtered data into geometric primitives that can be rendered. For instance, it can generate OpenGL display lists from triangulated models or create polygonalized versions of glyphs. The rendering phase takes the outputs of the map phase and creates images, taking advantage of graphics hardware when possible.

The early 1990s brought us Kitware's Visualization Toolkit (VTK) [11], which is an open source, object-oriented toolkit based on the dataflow programming model. For efficiency, the core components of the system are written in a compiled language (C++). For flexibility and extensibility, an interpreted language (e.g., Python, Tcl, or Java) can be used for higher-level

applications. This scripting capability and a large core set of algorithms have promoted VTK to its current status as one of the most popular visualization packages for researchers. VTK is widely used around the world, and many tools have been developed on top of it. Its open-source license has enabled the development of a number of influential end-user visualization tools. Two such tools developed under DOE funding are ParaView and VisIt. The ParaView project started by James Ahrens at Los Alamos National Laboratory [12] was aimed not only at extending VTK into a parallel framework but also at developing a turnkey end-user tool that did not require users to explicitly build dataflow graphs. VisIt [13] had a similar goal, but its design was dictated by the visualization needs of Lawrence Livermore National Laboratory. In particular, the development team led by Hank Childs decided to keep the look and feel of LLNL's MeshTV visualization tool.

In the 1990s, we also saw the start of the movement toward integrating scientific computing, parallelism, and visualization. SCIRun [14] and GRASPARC (GRAphical Support for PARallel Computing) [15] are two great examples of systems developed during this time. For the most part, other than the ability to generate complex visualizations, these systems lacked other features that were explicitly aimed at supporting the scientific process. One notable exception is that as part of their pioneering work on the GRASPARC system, Brodlie and colleagues proposed the use of a history mechanism that allowed scientists to steer an ongoing simulation by backtracking a few steps, changing parameters, and resuming execution.

3. Building tools to support scientific discovery

While visualizations in the form of plots, pictures, animations, and interactive 3-D applications are very important to generate insight, recently there has been a new research trend that is based on the premise that it is possible to do more to help humans be more effective at scientific discovery [3, 4, 16]. Pirolli and Card [17, 18] study the sensemaking process (i.e., the process of looking for meaning in a body of data with the intent of solving particular problems). They propose categories for the overall process and explore opportunities where software can help. The goal is to "enhance" the discovery process by providing software tools that support existing tasks, for example, better organization tools that improve the utilization of working memory. Social data analysis, a process by which multiple people collectively analyze data, has recently been proposed as a means to uncover new paths of discovery, and tools have been proposed that are aimed at facilitating collaboration and knowledge reuse.

Very recently, Schrinivasan and van Wijk [16] proposed Aruvi, a system for supporting the analytical reasoning process. Their system is based on three main components: a *data view*, where a user can interact with visual representations of the data (this is effectively what is available in traditional visualization systems, such as the ones that were discussed in the previous section and the main visualization supported in their system is the scatterplot), a *navigation view* which provides a visual representation of the exploration process itself, basically it records the history of exploration; and a *knowledge view*, where users can maintain analysis artifacts (e.g., hypothesis, conclusions) with direct links to the plots that support those conclusions. While the type of analysis that can be done with Aruvi is somewhat limited, the system clearly shows the potential for building tools that can be used for improving the support for discovery.

Social Web sites and web-based communities (e.g., Flickr, Facebook, Yahoo! Pipes), which facilitate collaboration and sharing between users, are becoming increasingly popular. An important benefit of these sites is that they enable users to leverage the wisdom of the crowds. For example, in Flickr, users, in a mass collaboration approach, tag large volumes of pictures. These tags, in turn, help them to more easily find pictures they are looking for. A notable social data analysis system is IBM's ManyEyes. This system is based on the blog paradigm and enables a large number of users to collaboratively analyze data. Users can upload datasets into the site and create visualizations from a dozen or so predefined types. Anyone can modify the

existing visualizations (always in a nondestructive manner) and comment on each other's work, just as in a normal blog. The discussions are persistent, and they provide a log that can be used for understanding any conclusions reached by the participants. It has been reported that ManyEyes gets over 500,000 visits a month, clearly highlighting the appeal of open discussion.

While it would be interesting to use the same technology for supporting scientific discovery, there are many challenges [19]. The heterogeneity of the data, its size, and its location greatly complicate the data analysis pipelines. Whereas existing "social data analysis" (SDA) systems require that data be uploaded to a central location for analysis, in many scientific applications that manipulate large volumes of data, this is not feasible. Furthermore, the visualization process is likely to require staged processing and a larger variety of underlying visualization techniques than what is currently supported by systems such as ManyEyes. Data analysis generates more data (e.g., graphs, visualizations) that add to the overflow of information scientists need to deal with. Thus, ad hoc approaches to data exploration, which are widely used in the scientific community, have serious limitations. In particular, scientists and engineers need to expend substantial effort managing data (e.g., scripts that encode computational tasks, raw data, data products, images, and notes) and to record provenance information so that basic questions can be answered, such as: Who created a data product and when? When was it modified and by whom? What was the process used to create the data product? Were two data products derived from the same raw data? Not only is the process time-consuming but also error-prone. The absence of provenance makes it hard (and sometimes impossible) to reproduce and share results, to solve problems collaboratively, to validate results with different input data, to understand the process used to solve a particular problem, and to reuse the knowledge involved in the data analysis process. In addition, it greatly limits the longevity of the data products–without precise and sufficient information about how the data product was generated, its value is greatly diminished. SDA systems aimed at the scientific domain need to provide a flexible framework that not only enables scientists to perform complex analyses over large data but also captures detailed provenance of the analysis process.

4. VisTrails: provenance and data exploration

Data exploration is an inherently creative process that requires users to locate relevant data, to visualize these data and discover relationships, to collaborate with peers while exploring different solutions, and to disseminate results. Given the size of data and complexity of analyses that are common in scientific exploration, new tools are needed and existing tools should be extended to better support creativity [3, 4].

The ability to systematically capture provenance is a key requirement for these tools. The provenance (also referred to as the audit trail, lineage, and pedigree) of a data product contains information about the process and data used to derive the data product. The importance of keeping provenance information for data products is well recognized in the scientific community [20–27]. Provenance provides important documentation that is key to preserving the data, to determining the data's quality and authorship, and to reproducing as well as validating the results. In addition, the availability of provenance supports reflective reasoning, allowing users to store temporary results, to make inferences from stored knowledge, and to follow chains of reasoning backward and forward [6].

VisTrails (`www.vistrails.org`) is an open-source system that was designed to support exploratory computational tasks such as visualization, data mining, and integration. VisTrails provides comprehensive provenance management infrastructure and can be easily combined with existing tools and libraries. A new concept we introduced with VisTrails is the notion of *provenance of workflow evolution* [28]. In contrast to previous workflow and visualization systems that maintain provenance only for derived data products, VisTrails treats the workflows (or pipelines) as first-class data items and keeps their provenance. In Section 4.1, we discuss

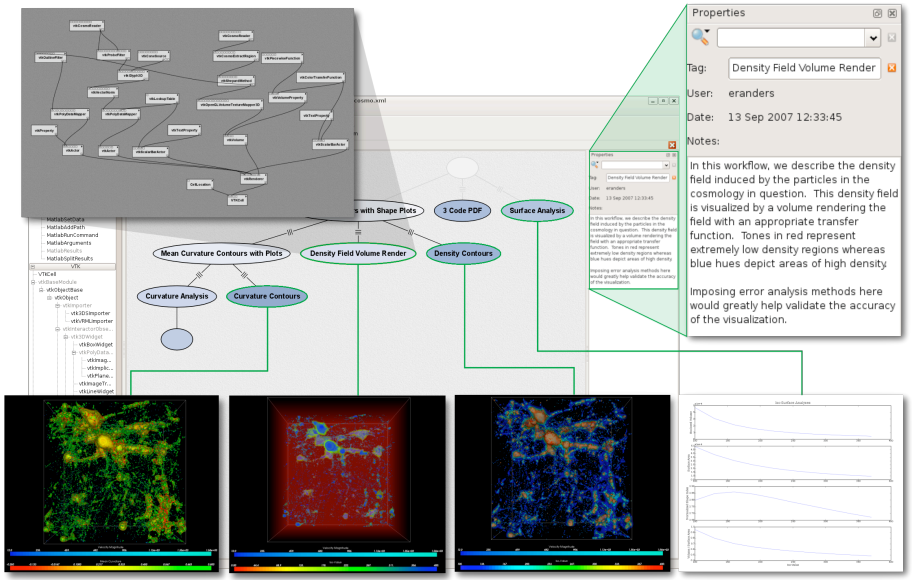


Figure 3. Example of an exploratory visualization for studying cellestial structures derived from cosmological simulations using VisTrails [30]. Complete provenance of the exploration process is displayed as a vistrail (history tree) with each node representing a workflow that generates a unique visualization. Detailed metadata is also stored including free-text notes made by the scientist, the date and time the workflow was created or modied, optional descriptive tags, and the user that created it.

how workflow evolution provenance enables a series of operations which simplify exploratory processes and foster reflective reasoning. For example, scientists can easily navigate through the space of workflows created for a given exploration task, visually compare workflows and their results, and explore large parameter spaces.

As part of its provenance infrastructure, VisTrails provides usable interfaces for exploring the provenance information and supporting knowledge reuse. In Section 4.2, we describe three components of the system that serve this purpose: a query-by-example interface, a mechanism for refining workflows by analogy, and a recommendation system that aids users in the design of workflows.

VisTrails is an extensible system. Like workflow systems, it allows pipelines to be created that combine multiple libraries. In addition, the VisTrails provenance infrastructure can be integrated with interactive tools, which cannot be easily wrapped in a workflow system. We give a brief overview of how new packages can be added to VisTrails in Section 4.3, where we also describe a VisTrails-based provenance plug-in developed for ParaView [29].

4.1. Using provenance to streamline data exploration

Compared both to visualization [14, 29] and scientific workflow systems [31–34], a key distinguishing feature of VisTrails is that, besides recording provenance for data products, it captures information about how workflows evolve over time. VisTrails uses a new change-based provenance mechanism [28, 35] that uniformly captures changes to parameter values as well as to workflow definitions. By maintaining *detailed provenance of the exploration process*, it not only ensures reproducibility but also allows scientists to easily navigate through the space of workflows created for a given exploration task.

Interacting with History. Figure 3 shows an example of an exploratory visualization using

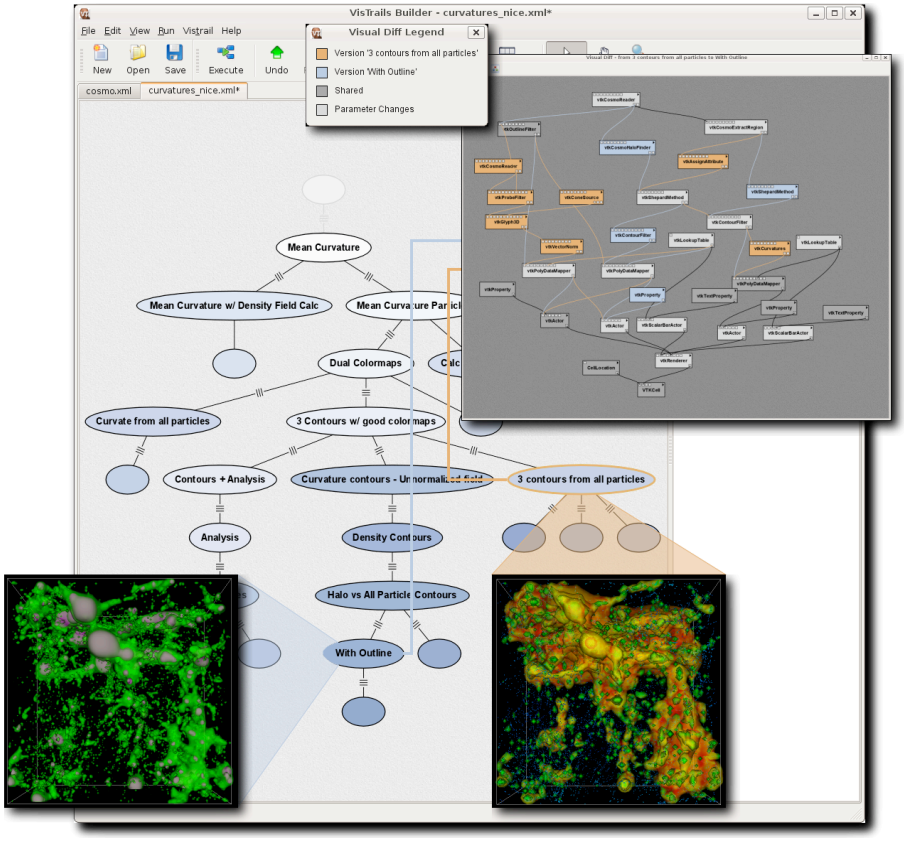


Figure 4. By representing provenance as a series of actions that modify a pipeline, visualizing the differences between two workflows becomes possible. The difference between workflows is described in a meaningful way, as an aggregation of the two workflows forming it. This representation of the difference is both informative and intuitive, reducing the time it takes to understand how two workflows are functionally different.

VisTrails. In the center, the vistrail (the visual trail) captures all modifications users apply to the visualizations. Each node in the vistrail tree corresponds to a workflow (or pipeline) and edges between two nodes correspond to changes applied to transform the parent pipeline into the child (e.g., through the addition of a module or a change to a parameter value). The tree-based representation allows a scientist to return to a previous version in an intuitive way, to undo bad changes, to compare different workflows, and to be reminded of the actions that led to a particular result.

Comparing Workflows and their Results. Another important benefit of the change-based provenance model is that it enables a series of operations that simplify the exploration process including the ability to compare workflows and a scalable mechanism for exploring large parameter spaces [28]. The discovery process requires many trial-and-error steps, and it is not uncommon for tens to hundreds of different workflows to be generated in the course of a study. In such a scenario it becomes important to understand what the differences between workflows are, especially if multiple people are collaboratively exploring the data. As illustrated in figure 4, VisTrails provides a visual difference that allows structural comparison of workflows. This figure depicts two similar workows that yield drastically different results. By analyzing the difference between the two workflows, the salient aspects of each visualization can be more conveniently explored. The visual difference shows in blue modules unique to the workflow on the left; in

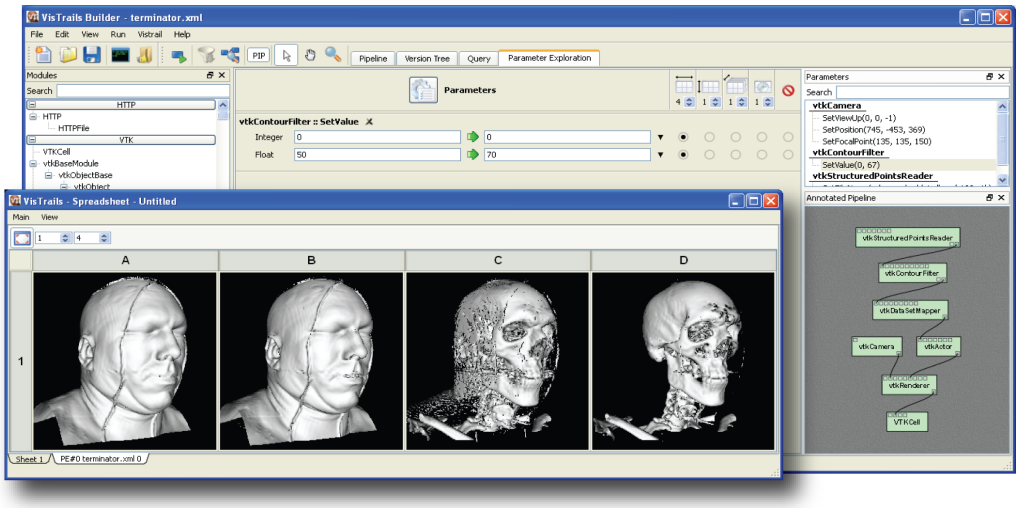


Figure 5. Parameter exploration is performed in VisTrails using a simple interface. The results are computed efficiently by avoiding redundant computation and displayed in the spreadsheet for interactive comparative visualization.

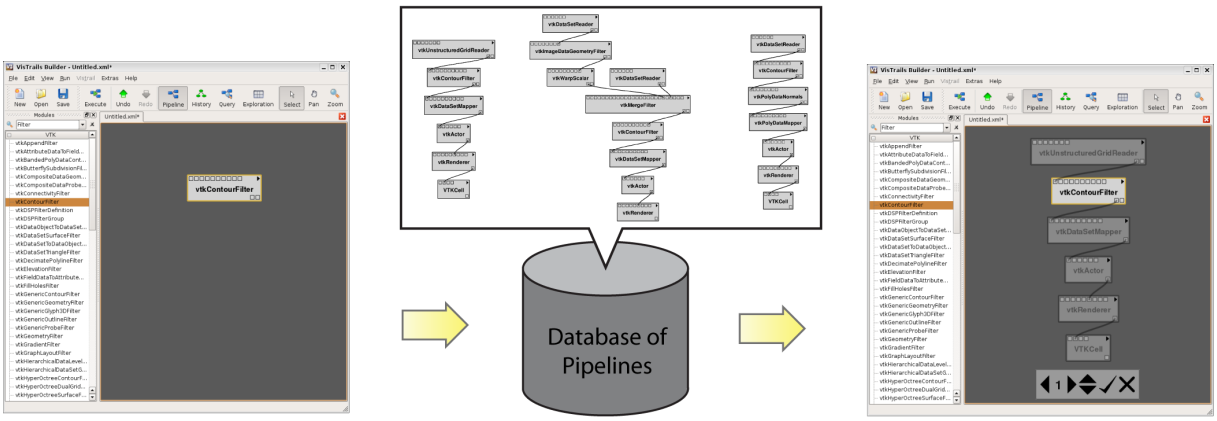
orange modules unique to the workflow on the right; and in gray, the modules shared by the two workflows.

Exploring Parameter Spaces. The parameter exploration mechanism allows the user to specify a set of parameters to be explored, as well as how to explore them and display the results. As a simple one dimensional example, we show an exploration of the isosurface for the visible human head value as four steps between 50 and 70 and display them horizontally in the VisTrails Spreadsheet. Figure 5 shows the interface as well as the results of this exploration. The VisTrails Spreadsheet provides the means to compare visualizations in different dimensions (row, column, sheet, time). The cells in the spreadsheet can be linked to synchronize the interactions between visualizations. We note that VisTrails leverages the dataflow specifications to identify and avoid redundant operations. By using the same cache for different cells in the spreadsheet, VisTrails allows the efficient exploration of a large number of related visualizations [36].

4.2. Exploring and reusing provenance

Although the benefits of using workflow and workflow-based visualization systems are well known, the fact that workflows are hard to create and maintain has been a major barrier to wider adoption of the technology in the scientific domain [1, 37, 38]. VisTrails provides infrastructure that allows users to reuse provenance information to simplify the creation and refinement of workflows. This includes the ability to query provenance; to refine workflows by analogy, using previous workflow refinements as examples; and a recommendation system that guides users through the process of workflow design by using a database of previously created workflows.

Auto-Completion for Workflows. Scientific workflows and visualization pipelines are specified as graphs, where nodes represent processes (or modules) and edges capture the flow of data between the processes. Assembling these workflows requires programming expertise as well as detailed knowledge of multiple tools and libraries. Often, it also entails tightly coupled collaborative efforts. Even for systems that have sophisticated visual programming interfaces, such as DX, AVS and SCIRun, the path from the raw data to insightful visualizations is quite laborious and error-prone. Significant effort is required to build new workflows and to manually



(a) (b) (c)

Figure 6. Workflow auto-completion: (a) A user starts by adding a module to the pipeline. (b) The most likely completions are generated using indexed paths computed from a database of pipelines. (c) A suggested completion is presented to the user as semi-transparent modules and connections. The user can browse through suggestions using the interface and choose to accept or reject the completion.

modify existing workflows to cater to new requirements (e.g., to use different parameters or different visualization techniques). The difficulty of these tasks makes the use of workflows impossible for domain scientists with little or no programming expertise.

By learning common paths used in existing workflows, VisTrails predicts a set of likely module sequences that can be presented to the user as suggestions during the design process. The system acts as an auto-completion mechanism for workflows, suggesting potential modules and connections in a manner similar to a Web browser suggesting URLs. The workflow completions are presented graphically in a way that allows the user to easily explore and accept suggestions or disregard them and continue working as they were. Figure 6 gives an overview of how the auto-completion mechanism works. For more details, see [39].

Querying Workflows. The ability to query both the specification and provenance of workflows enables users to better understand exploratory processes and their results. For example, users can identify workflows that are suitable for and can be reused for a given task or can identify workflow instances that have been found to contain anomalies. Provenance information can also be associated with data products (e.g., images, graphs), allowing structured queries to be posed over these unstructured data.

Besides supporting simple, keyword-based queries as well as structured queries, VisTrails also allows queries to be specified by example, using an interface that mirrors the interface for building a workflow [40]. As figure 7 illustrates, a user can construct (or copy and paste) a pipeline fragment into the VisTrails query tab to identify in the history tree all nodes that contain that fragment. The user can then browse through the highlighted nodes. When a highlighted node is clicked on, the workflow is displayed and the modules that match the query are highlighted. The user can then click on the individual modules to view execution log records associated with modules.

Refining Workflows by Analogy. While the query interface allows users to identify workflows (and subworkflows) that are relevant for a particular task, the *analogy operation* provides a mechanism for reusing these pipelines to construct new data products in a semi-automated manner—without requiring users to directly manipulate or edit the dataflow specifications [40].

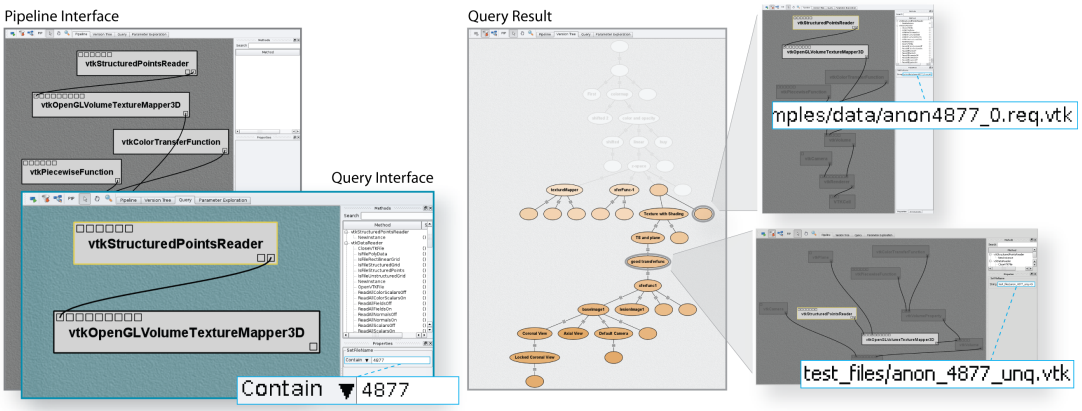


Figure 7. Querying by example. The interface for building a query over an ensemble of pipelines is essentially the same as the one for constructing and updating pipelines. In fact, they work together: portions of a pipeline can become query templates by directly pasting them onto the Query Canvas. In this figure, the user is looking for a volume-rendered image of a file whose name contains the string "4877." The system highlights the matches both at the visualization level (version tree, shown in the middle) and at the module level (shown in the right insets).

Figure 8 illustrates the use of analogies for creating visualizations. To apply an analogy, first the difference between a source pair of analogous visualizations is computed. This difference is then transferred to a third visualization. Analogies can be used as the basis for scalable updates: the user does not need to have knowledge of the exact details of the three visualization workflows to perform the operation.

4.3. Provenance for workflows and interactive applications

Integrating Tools and Libraries. VisTrails provides an infrastructure that allows users to integrate user-defined functions and libraries. Specifically, users can incorporate their own visualization and simulation codes into pipelines by defining custom modules (or wrappers) which are bundled in *packages*. A VisTrails package is a collection of Python classes, where each class corresponds to a module. These wrappers specify the input and output parameters for the modules as well as how the underlying function is invoked. Detailed instructions on how to add packages are given in [41]. The VisTrails distribution provides support for several libraries, including VTK, Web Services, and SciPy.

Provenance-Enabling Existing Applications. The initial focus of the VisTrails project was to develop a provenance infrastructure for computational tasks specified as workflows. But although the change-based provenance model was originally designed to capture changes applied to a workflow definition (Section 4.1), it is extensible and can be applied in different environments. In particular, it can capture change actions that are applied by a user through an interactive user interface (e.g., slicing a volume or editing a parameter in a scientific visualization system). The change-based representation of provenance is easily incorporated into existing applications that provide a mechanism for controlling the actions that are being performed by a user via a graphical interface. And by doing so, the interfaces and mechanisms provided by VisTrails for interacting and exploring provenance can be reused. An important advantage of this approach is that, instead of forcing users to learn to use a new environment, it allows users to capture and leverage provenance using the applications and environments that they are used

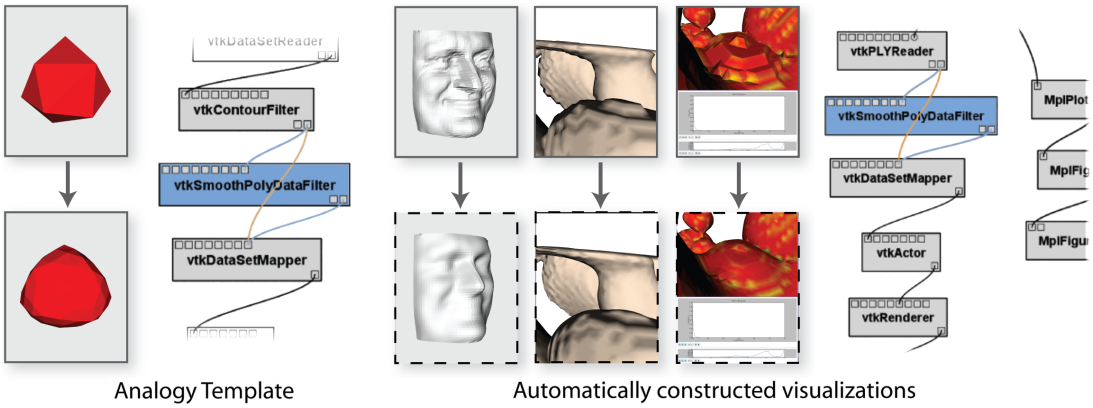


Figure 8. Visualization by analogy. The user chooses a pair of visualizations to serve as an analogy template. In this case, the pair represents a change where a file downloaded from the WWW is smoothed. Then, the user chooses a set of other visualizations that will be used to derive new visualizations, with the same change. These new visualizations are derived automatically. The pipeline on the left reflects the original changes, and the one on the right reflects the changes when translated to the last visualization on the right. The pipeline pieces to be removed are portrayed in orange, and the ones to be added, in blue. Note that the surrounding modules do not match exactly: the system figures out the most likely match.

to.

In general, applications that take advantage of the model-view-controller paradigm [42] have a mechanism for storing and reusing actions: the undo and redo operations. In a scientific visualization system, for instance, with *undo* a user is able to walk through the steps they took to create an image, albeit backwards. Although undo does not capture the complete exploration process nor does it persist across sessions, it provides valuable context for granularity of actions. The designers of the software have already determined the granularity of actions by designing the undo stack. The undo stack of an application may individually capture single mouse events or keyboard strokes if they are needed to recreate the state. Furthermore, interactions performed by the user may cause multiple actions to be performed, which the undo stack will store as one step. We can capture actions at the same granularity in which the undo stack does. In fact, in practice it is simpler to capture actions as they are being added to the undo stack instead of where they are handled by the controller. Obviously, this depends on the completeness and availability of the undo/redo mechanism in the application.

Figure 9 shows ParaView together with the VisTrails Provenance Explorer, transparently capturing the complete exploration process. The provenance capture mechanism was implemented by inserting monitoring code in ParaView's undo/redo mechanism, which captures changes to the underlying pipeline specification. Essentially, the action on top of the undo stack is added to the vistrail in the appropriate place, and undo is reinterpreted to mean "move up the version tree." Note that the change-based representation is both simple and compact—it uses substantially less space than the alternative of storing multiple instances or *versions* of the state.

5. Conclusions and future directions

In this paper, we discussed important issues that must be considered in the design of software environments for exploratory data analysis and visualization. The end goal is clear: build tools that aid scientists and streamline scientific discoveries. But to achieve this we need

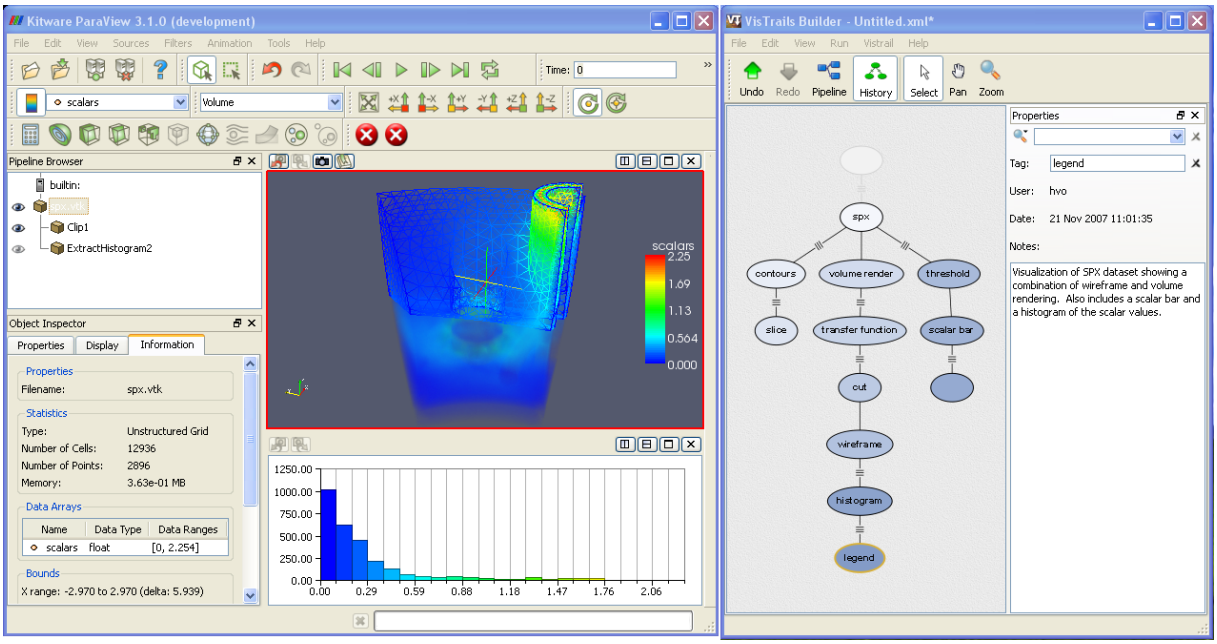


Figure 9. Screenshot of ParaView (left) with the provenance captured by VisTrails and displayed as a version tree in a separate window (right). This preliminary prototype taps into ParaView undo/redo mechanism to capture the exploration process.

creativity support tools that enable scientists to be more efficient and that provide effective means for collaboration. We view provenance management and scientific workflows as key enabling technologies that can serve as the basis for these tools.

There is still much work left for the visualization community to make this vision, of having visualization as an integral part of the scientific process, a reality. Below, we describe some of the directions we are currently pursuing. The ability to support and integrate multiple tools is of paramount importance in practice. Users often have a set of tools that they have been using and/or developing, and on which they have made substantial investments. For most, replacing those tools is not an option. We believe that any "scientific discovery environment" needs to be built in such a way as to interoperate with existing tools: it needs to be modular, and to have components that can be easily incorporated into or combined with custom tools. An early example of work toward enhancing the capabilities of an existing powerful tool is our provenance plugin for ParaView. We plan to work on similar capabilities for VisIt. On a more ambitious plan, we are also working with Dean Williams and his group on extensions to CDAT. Although CDAT has been developed with climate applications in mind, most of the concepts of the system could be reused for developing a more general Data Analysis Tool (DAT) by merging CDAT and VisTrails functionality.

Publications are an essential component of the scientific process. But although the way science is done has changed substantially in recent years, the same cannot be said of scientific publications. We have been exploring new tools and ways to extend existing tools (e.g., LaTeX, wiki, MS Word) to support the creation of articles that are more dynamic, reproducible, and verifiable. By allowing readers to replicate the experiments in a paper and to maintain detailed provenance of artifacts used across papers, the publications of the future have the potential to accelerate scientific discoveries and to improve the quality of science.

As provenance-enabled systems (including workflow systems) are deployed, large volumes of provenance information are being collected. Having these large collections creates opportunities

to explore provenance in novel ways. But it also presents several challenges. Just collecting provenance information does not bring major benefits. For the data to be truly useful, we need effective and efficient ways to analyze it. As provenance collections grow, new mechanisms are needed to deal with a potential information overload. The problem of mining and extracting knowledge from provenance data, however, has been largely unexplored. We are interested in "provenance analytics," which we define to be a new area aimed at investigating new techniques for mining, visualizing and reusing provenance of computational tasks.

Acknowledgments

We thank our students and collaborators for contributing to this work, and the anonymous reviewers for comments that helped us to improve our manuscript. Our research has been funded 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), and IBM Faculty Awards (2005, 2006, and 2007).

References

[1] Johnson C R, Moorhead R, Munzner T, Pfister H, Rheingans P and Yoo T S 2006 *NIH-NSF Visualization Research Challenges* (IEEE Computer Society) `http://tab.computer.org/vgtc/vrc/index.html`
[2] van Wijk J 2005 *Proceedings of IEEE Visualization*
[3] Shneiderman B 2007 *Commun. ACM* **50** 20–32
[4] Shneiderman B 2002 *Commun. ACM* **45** 31–34
[5] Kniss J, Kindlmann G and Hansen C 2002 *IEEE Transactions on Visualization and Computer Graphics* **8** 270–285
[6] Norman D A 1994 *Things That Make Us Smart: Defending Human Attributes in the Age of the Machine* (Addison Wesley)
[7] Tufte E R 1997 *Visual Explanations: Images and Quantities, Evidence and Narrative* (Graphics Press)
[8] Plaisant C 2004 *Proceedings of the working conference on Advanced visual interfaces* (ACM) pp 109–116
[9] Upson et al C 1989 *IEEE Computer Graphics and Applications* **9** 30–42
[10] IBM OpenDX `http://www.research.ibm.com/dx`
[11] Schroeder W, Martin K and Lorensen B 2003 *The Visualization Toolkit An Object-Oriented Approach To 3D Graphics* (Kitware)
[12] Law C C, Henderson A and Ahrens J 2001 *IEEE Symposium on Parallel and Large-data Visualization and Graphics 2001* pp 125–128
[13] Childs H, Brugger E S, Bonnell K S, Meredith J S, Miller M, Whitlock B J and Max N 2005 *IEEE Visualization 2005* pp 190–198
[14] Parker S G and Johnson C R 1995 *Proceedings of Supercomputing* p 52
[15] Brodlie K, Poon A, Wright H, Brankin L, Banecki G and Gay A 1993 *IEEE Visualization 1993* pp 102–109
[16] Shrinivasan Y B and van Wijk J J 2008 *ACM SIGCHI 2008* pp 1237–1246
[17] Pirolli P and Card S 1999 *Psychological Review* **106** 643–675
[18] Pirolli P and Card S 2005 *International Conference on Intelligence Analysis*
[19] Freire J and Silva C 2008 *CHI Social Data Analysis Workshop*
[20] Freire J, Koop D, Santos E and Silva C T 2008 *Computing in Science and Engineering* **10** 11–21
[21] Davidson S B and Freire J 2008 *Proceedings of ACM SIGMOD* pp 1345–1350
[22] Davidson S B, Boulakia S C, Eyal A, Ludäscher B, McPhillips T M, Bowers S, Anand M K and Freire J 2007 *IEEE Data Eng. Bull.* **30** 44–50
[23] Miles S, Groth P, Branco M and Moreau L 2006 The requirements of using provenance in e-science experiments Tech. rep. ECS, University of Southampton
[24] Bose R and Frew J 2005 *ACM Computing Surveys* **37** 1–28
[25] Simmhan Y L, Plale B and Gannon D 2005 *SIGMOD Record* **34** 31–36
[26] Moreau L, Ludäscher B, Altintas I, Barga R S, Bowers S, Callahan S, Chin Jr G, Clifford B, Cohen S, Cohen-Boulakia S, Davidson S, Deelman E, Digiampietri L, Foster I, Freire J, Frew J, Futrelle J, Gibson T, Gil Y, Goble C, Golbeck J, Groth P, Holland D A, Jiang S, Kim J, Koop D, Krenek A, McPhillips T, Mehta G, Miles S, Metzger D, Munroe S, Myers J, Plale B, Podhorszki N, Ratnakar V, Santos E, Scheidegger C, Schuchardt K, Seltzer M, Simmhan Y L, Silva C, Slaughter P, Stephan E, Stevens R, Turi D, Vo H, Wilde M, Zhao J and Zhao Y 2007 *Concurrency and Computation: Practice and Experience*

[27] Freire J, Miles S and Moreau L 2007 Second provenance challenge `http://twiki.ipaw.info/bin/view/Challenge/SecondProvenanceChallenge`
[28] J Freire, C T Silva, S P Callahan, Santos E, Scheidegger C E and Vo H T 2006 *International Provenance and Annotation Workshop (IPAW)* LNCS 4145 pp 10–18
[29] Kitware Paraview `http://www.paraview.org`
[30] Anderson E W, Ahrens J, Heitmann K, Habib S and Silva C 2008 *Computing in Science and Engineering* **10** 30–37
[31] The Kepler Project `http://kepler-project.org`
[32] The Taverna Project `http://taverna.sourceforge.net`
[33] The Triana Project `http://www.trianacode.org`
[34] Ailamaki A, Ioannidis Y E and Livny M 1998 *Proc. of the Conference on Scientific and Statistical Database Management* pp 190–199
[35] Callahan S, Freire J, Santos E, Scheidegger C, Silva C and Vo H 2006 *IEEE Workshop on Workflow and Data Flow for Scientific Applications (SciFlow)*
[36] Bavoil L, Callahan S, Crossno P, Freire J, Scheidegger C, Silva C and Vo H 2005 *Proceedings of IEEE Visualization* pp 135–142
[37] 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 *SIGMOD Rec.* **34** 24–30
[38] Schopf J, Coleman I, Procter R and Voss A 2006 Report of the user requirements and web based access for eresearch workshop Tech. Rep. UKeS-2006-07 UK National e-Science Centre
[39] Koop D, Scheidegger C, Callahan S, Freire J and Silva C 2008 *IEEE Transactions on Visualization and Computer Graphics* Accepted with minor revisions.
[40] Scheidegger C, Koop D, Vo H, Freire J and Silva C 2007 *IEEE Transactions on Visualization and Computer Graphics*
[41] VisTrails Packages `http://www.vistrails.org/index.php/UsersGuideVisTrailsPackages`
[42] Krasner G E and Pope S T 1988 *Journal of Object-Oriented Programming* **1**(3) 26–49