The Kitware Source contains articles related to the development of Kitware projects in addition to software updates, news, and other content relevant to the open source community. In this issue, Luis Ibáñez, David Cole, Bill Hoffman, Brad King, and Xiaoxiao Liu discuss modularization as a way to group interrelated code for easier code maintenance.

Gabe Hart, Marc Neithammer, Danielle Pace, Jack Van Horn, and Stephen Aylward give an overview of their soon-to-be-published paper on traumatic brain injury assessment using geometric metamorphosis.

Brad Davis and Bill Hoffmann explore issue RADAR tracking for streamlining workflow and managing projects.

George Zagaris, Charles Law, and Berk Geveci examine visualization and analysis of hierarchical structured dataset enhancements using an AMR approach.

In Kitware News, Kitware team members share their experiences at recent conferences, and training courses, and highlight new offerings from the company.

The Source is part of a suite of products and services offered to assist developers in getting the most out of our open-source tools. Project websites include links to free resources such as: mailing lists, documentation, FAQs, and Wikis.

Kitware supports its open source projects with textbooks, consulting services, support contracts, and training courses. For more information, please visit www.kitware.com.

ANNOUNCING KIWI VIEWER
Coming soon to the Apple app store is KiwiViewer, an exciting new project that brings scientific visualization to mobile, multi-touch devices. KiwiViewer is built on the vtkES library, a budding open-source effort to harness the power and flexibility of VTK on mobile and multi-touch devices.

KiwiViewer is designed to be simple and lightweight with the following features:

- Interactive model exploration using multi-touch displays
- Support for surface and line geometry
- Ability to load data from the web, email, or DropBox
- Available on the iPad

Watch for KiwiViewer in the app store this summer!
CMAKE 2.8.5

The latest release of CMake is notable as it was made possible in part by a number of individuals around the world who contributed code. Overall, 33 people submitted changes that are now part of CMake 2.8.5. The new git workflow and the fact that git patches preserve the author in the commit objects, even when applied and pushed by someone else, means that we can now measure and report this number.

There are several other noteworthy additions included in this release. The ProcessorCount module provides a CMake language-based way to retrieve the number of processors usable for parallel building or testing. The CheckPrototypeDefinition module allows users to check if an exact function signature definition is available, rather than a simple function-exists-by-name-only check. The file (UPLOAD…) subcommand allows users to send files to web servers from CMake scripts. The ctest_upload() command allows users to send attachment files (i.e. an installer for your app) to the CDash server where others can download them. This feature also requires upgrading your CDash server to the svn trunk version for now, as this feature is not yet in an official CDash release.

For the full list of changes and updates made since the 2.8.4 release, please see the file ChangeLog.manual in the CMake source distribution and the bug tracker’s change log page for the full list of issues resolved.

SLICER 4.0 GAMMA RC2

Kitware is pleased to announce the release candidate 2 of Slicer 4.0 Gamma. Many new features have been added since beta and RC1. Release candidate 1 included additional modules such as “volume rendering” and “sample data”. Extension modules were also integrated with CDash for easy configuration, submission, and retrieval.

Release candidate 2 includes a number of small bug fixes; changes to the presets in the volume rendering module; a first pass in optimizing the 2D views and other various speed improvements; the addition of the Harden Transform in the data module; inclusions of a Line/Tube/Glyph display tab in the Tractography Display Module; the ability to select annotations; and improvements to extension support. Additionally, VTK observers can be debugged using the Event Broker module.

Visualization & Analysis of AMR Datasets

Multiphysics simulations of complex phenomena exhibiting large length and time scales, such as astrophysics, have been greatly facilitated by the AMR approach. By discretizing the computational domain with multiple, overlapping, uniform grids of varying resolution, high-fidelity accuracy can be attained in regions of interest where high-resolution, uniform grids are employed. Lower-resolution is used where the solution is not changing to reduce memory and storage requirements; however, the nested grid structure poses new challenges for the visualization and analysis of such datasets. Specifically, visualization algorithms need to process a dynamic hierarchy of grids that change at each time-step and properly handle the overlapping regions and inter-level grid interfaces. At the same time, the size generated from typical simulations can be prohibitively large, necessitating the development of new visualization approaches that enable scientists and engineers to analyze their data. The enabling technology for visualization and analysis at such large scales is query-driven visualization[3] and is the primary and ongoing goal of this work.

The basic idea of query-driven visualization is to selectively load data within a user-supplied region of interest (ROI). It is rooted in the observation that most of the insights are derived from only a small subset of the dataset [4,5]. For example, the user may request to visualize the data at a specific level of resolution, or just a slice of the data. To this end, we have developed an extensible and uniform software framework for AMR datasets that supports loading of data on-demand, as well as handling of particles. Further, we have implemented a slice filter that operates directly on AMR data and produces 2-D AMR slices that enables the inspection of volumetric data in varying resolutions. Lastly, we have developed an AMR dual-grid extraction filter that normalizes the representation of cell-centered AMR data to an unstructured grid and allows the use of existing contouring algorithms (i.e., Marching Cubes (MC)) for iso-surface extraction without further modification. The remainder of this article discusses the aforementioned developments in more detail.

AMR DATA STRUCTURES

The algorithms and operations for query-driven visualization are best described by first presenting the underlying VTK AMR data-structures. Although there are several variations of AMR, our implementation focuses on Structured AMR datasets where the solution is stored at the cell centers. In particular, the algorithms and operations described herein focus on the Berger-Collela AMR scheme [1,2], which imposes the following structure:

1. All grids are Cartesian.
2. Grids at the same level do not overlap.
3. The refinement ratios, \( R_L \), between adjacent levels are integer (typically 2 or 4) and uniform within the same level.
4. Grid cells are never partially refined; i.e., each cell is refined to four quads in 2D or eight hexahedra in 3D.

A sample 2-D AMR dataset with two levels and a refinement ratio, \( R_L=4 \), is shown in Figure 1. Note that the
low-resolution cells that are covered by high-resolution grids, shaded by green and red wireframe colors, are hidden.

In VTK, the collection of AMR grids is stored in a *vtkHierarchicalBoxDataSet* data-structure. Each grid, \(G(L_i,k)\), is represented by a *vtkUniformGrid* data structure where the unique key pair \((L_i,k)\) denotes the corresponding level \(L_i\) and the grid index within the level \(k\) with respect to the underlying hierarchical structure. An array historically known as *IBLANK*, stored as a cell attribute in *vtkUniformGrid*, denotes whether a cell is hidden or not. The blanking array is subsequently used by the mapper to hide lower resolution cells accordingly when visualizing the dataset.

However, the size of the data generated for typical simulations prohibits loading the entire dataset in memory. Even in a parallel, distributed memory environment, loading the entire dataset in memory is not a practical solution for most interactive visualization tasks on AMR datasets. Most of the time, scientists and engineers interact with their datasets by either performing queries that concentrate on a sub-region (e.g., a slice of the data at a given offset), or perform operations that can operate in a streaming fashion (e.g., computing derivatives or other derived attributes). To enable the execution of such queries without loading the entire dataset in memory, metadata information is employed. The metadata stores a minimal set of geometric information for each grid in the AMR hierarchy. Specifically, the AMR metadata, \(B(L_i,k)\), corresponding to the grid \(G(L_i,k)\), is represented using a *vtkAMRBox* object and it consists of the following information:

1. \(N=(Nx, Ny, Nz)\) -- the cell dimensions of the grid (since the data is cell-centered)
2. The grid spacing at level \(L_i\), \(h_i=(hx, hy, hz)\)
3. The grid level \(L_i\) and grid index \(k\)
4. The global dataset origin, \(X=(X_0, Y_0, Z_0)\), i.e., the minimum origin from all grids in level \(L_0\) and
5. The LoCorner and HiCorner, which describe the low and high corners of the rectangular region covered by the corresponding grid in a virtual integer lattice with the same spacing \(h_i\) that covers the entire domain.

Given the metadata information stored in the AMR box of each grid, the refinement ratio at each level can be easily computed using relationship (1) from Table 1. Further, the cartesian bounds the corresponding grid covers and the number of points and cells is also available (see relationships 2-4 in Table 1). Notably, geometric queries such as determining which cell contains a given point, or if a grid intersects a user-supplied slice plane, can be answered using just the metadata. With this approach, data that satisfies a given query can be loaded into memory for analysis and visualization more efficiently through inspection of the metadata.

### QUERY-DRIVEN AMR VISUALIZATION FRAMEWORK

The core functionality for query-driven visualization is provided by employing a lazy-loading design pattern, which defers loading the data until it is requested. This imposes slightly different requirements in the design and implementation of readers and filters for query-driven visualization. First, AMR readers must implement functionality for (a) loading the metadata given an input file and (b) for accepting data requests from downstream filters and/or interactively by the user. Second, filters must implement functionality for sending data requests upstream to the reader according to user-supplied input. To make this concept more concrete, consider slicing an AMR dataset at a low resolution and inspecting the volume as a simple example. Typically, once the region of interest is identified, a higher resolution slice would be required for further analysis. Allowing users to increase the resolution on a slice triggers an upstream request to the reader to load the blocks to the prescribed level of resolution that intersect the given planar slice. Efforts are ongoing in enhancing the VTK pipeline to better support this type of demand-driven operation within ParaView. In this section, we present preliminary work towards that goal, specifically targeting AMR datasets.

#### Query-Driven AMR Readers

Typical AMR codes, such as Enzo [6,7] and FLASH [8], are hybrid codes that combine both the fluid quantities (e.g., density, velocity, etc) defined on the AMR grid hierarchy with particle quantities (e.g., mass, position, etc) in a single set of coupled governing equations. Hence, the capability to visualize both the particles and AMR grid datatypes is crucial in the analysis of such datasets. Two separate readers...
are implemented in our framework to address the separate queries required for each datatype (AMR grids and particles) respectively. A schematic class diagram of the readers implemented is depicted in Figure 2.

The abstract base classes implement common functionality to all AMR and particle readers respectively. Specifically, they define a common API, implement common UI components, and implement assignment of blocks to different processes when running in a distributed environment. In this approach, the user interface of all concrete AMR and particle readers is consistent and a set of common operations is guaranteed. In the present implementation, the framework has native support for Enzo and Flash datasets. Native support for other AMR datasets, such as RAMSES and Orion, is also projected in the future.

Typical operations the AMR reader provides for fluid data analysis include the ability to specify the maximum level of resolution and select which attributes to load. This functionality is extremely useful when loading large-scale datasets since most of the time loading the entire dataset in memory is not feasible. By loading just a couple of low resolution levels, users gain insight as to where potential regions of interest are and extract those regions for further analysis. Figure 3 (a) depicts the reader panel accessible from ParaView that illustrates the user controls currently enabled; namely, the ability to select a maximum level of resolution and which attributes to load. Figure 3 (b) shows the resulting surface mesh when prescribing a maximum resolution of seven.

Similar functionality is also enabled for the particle readers. Figure 4 (a) depicts the particle reader panel controls and (b) shows the resulting particle dataset. As with the AMR reader, the ability to select which attributes to load is enabled. Further, using a frequency parameter, particles can be subsampled; for example: in this case, the reader will load every other five particles, an operation that is commonly employed when analyzing particle simulations. In addition, particles can be filtered based on location or by particle type (e.g., in the case of Enzo datasets, Dark Matter, Star, or Tracer particles can easily hidden/shown).

The AMR Slice Operator

One of the most fundamental yet effective approaches for analyzing 3D AMR datasets is extraction of axis-aligned AMR slices through the dataset. Benefits to this approach include the simple calculation of which blocks to load, and the ability to exploit the multi-resolution, hierarchical data-structure within the slice. The AMR Slice operator is implemented in `vtkAMRSliceFilter`, a concrete implementation of the `vtkHierarchicalBoxDataSetAlgorithm`. It accepts as input a 3D AMR dataset, and produces a 2D AMR dataset at a user-supplied offset.

Given an offset (dx) and an axis-aligned plane \((P = \{XY, XZ, YZ\})\), construction of the 2D AMR dataset is achieved in the following steps:

1. Construct the cutplane, \(P\), positioned \(dx\) from the global dataset origin provided by the AMR metadata (recall section 2).
2. Given the bounds of each block from the metadata, determine which blocks intersect with the cutplane, \(P\).
3. For each block, create a 2D `vtkUniformGrid` instance along the plane \(P\).
4. Copy the solution from the 3D blocks to the corresponding 2D cells (direct injection).

Figure 5(b) shows the output generated by the AMR slice filter for the parameters set in Figure 5(a) as a simple example demonstrating the present functionality. It should be noted that in the present implementation, by increasing the resolution (level), upstream requests for higher resolution blocks are sent to the reader.
Slicing AMR datasets is a very powerful technique, especially when analyzing very large datasets. Scientists typically initially load the data at a low-resolution, create a slice, and sweep within the volume until they find regions of interest.

Once a region of interest is identified, scientists can increase the resolution to gain further insight into the physical phenomena and subsequently use other filters to perform other quantitative analysis, e.g., plots etc. within the slice. For example, Figure 7(a) shows a sample Flash dataset with slices positioned within a region of interest, acquired by sweeping the volume of the AMR dataset. Next, Figure 7(b) shows the slices after requesting higher resolution data to be loaded.

**DUAL-BASED ISO-SURFACE EXTRACTION OF AMR DATASETS**

Iso-surface extraction is a fundamental technique employed for qualitative analysis of scalar fields. However, without proper handling of AMR data, the use of commonly employed techniques, such as the Marching Cubes (MC) [9], is complicated by discontinuities at the inter-level interfaces.

To address this issue, we have implemented a `vtkAMRDualExtractionFilter`, which constructs a dual-mesh (i.e., the mesh constructed by connecting the cell-centers) over the computational domain that can be subsequently used for extracting the iso-surfaces by the MC approach.
For a schematic depiction of the primary steps of the dual-based, iso-surface extraction approach implemented in this work, see Figure 8. At a high-level, the main steps of the algorithm are the following:

1. Extract one layer of ghost-cells and transfer (direct injection) the solution from the adjacent cells of lower or equal resolution.
2. Process each block separately and generate the dual mesh, M. Note: this approach introduces gaps where there is a level difference (See Figure 8(b)).
3. Stitch the gaps by moving the nodes of the dual mesh (M) to the centroid of the adjacent lower resolution cells, i.e., Figure 8(c).
4. Pass the modified dual mesh to the MC for contouring, i.e., see Figure 8(d).

Although the current implementation is not yet at a production state, our approach has been successfully employed with complex AMR datasets, such as the Implosion Enzo AMR sample dataset shown in Figure 9(a) and the 3D Red-shift AMR dataset depicted in Figure 9(b).

**CONCLUDING REMARKS**

Current progress towards the development of a query-driven visualization framework targeting large-scale AMR datasets is presented with preliminary results. Ongoing efforts focus on further exploiting the VTK pipeline to better suit query-driven operations; implementation of new operators and more readers, such as RAMSES and Orion; integration with domain specific toolkits, such as yt [10]; and investigating approaches for query-driven volume rendering.

**ACKNOWLEDGEMENTS**

The work presented in this article is funded by DOE under Phase II SBIR Contract No. DE-FG02-09ER85461. The authors would like to thank Matt Turk from the Department of Astronomy at Columbia University and Paul Sutter from the University of Illinois at Urbana-Champaign for their insightful discussions and for providing astrophysical simulation datasets. Further, the authors would like to thank Utkarsh Ayachit, Robert Maynard, Sebastien Jourdain, and Dave DeMarle for helpful technical discussions.

**REFERENCES**


**George Zagaris** is an R&D Engineer in the Scientific Visualization team at Kitware. He joined Kitware in January 2011 and focuses primarily on the visualization and analysis of large-scale AMR data.

**Charles Law** is the VP of Strategic Growth at Kitware. He one of the five co-founders of Kitware. Charles’ research interests include parallel visualization of large data, CAD visualisation, and robotic path-planning algorithms.

**Berk Geveci** is the Director of Scientific Computing at Kitware’s Clifton Park office. He is one of the leading developers of the ParaView visualization application and the Visualization Toolkit (VTK).
ISSUE RADAR USING GIT/MASTER/NEXT/MANTIS

Kitware has a track record for applying agile software development processes to our projects. Since each project is unique in its needs and workflow, the same processes are not used for every project. We use mainstream development models like SCRUM [1], Kanban [2], and even waterfall [3], but we typically adapt the process that best suits the needs of a particular project. In this article, we will describe some of the tools we have used for a Kanban/SCRUM style project. The tools are open-source and available for all to use. A combination of git-master/next branchy workflow and the MANTIS [4] issue tracker have been used successfully to limit work in progress, and give developers and customers a realistic view of where the project stands at any point in time with little overhead by the developers. We have called this project management system RADAR.

This article describes the process used, and provides some instruction on how to set-up MANTIS and git to implement this system for your project.

MOTIVATION

Transparency is key in software project management. It makes people more comfortable to know what is going on, and it makes it easier to manage expectations of everyone involved. Comfortable customers with realistic expectations are a joy to work with and often become repeat customers. Conversely, opaque project management styles frequently lead to muddled expectations and frustration for all involved.

One major way to increase transparency is to maintain an issue tracker that is visible to the customer. An issue tracker is a communication tool: it makes it easy to agree on lists of things to do; it is a platform for prioritizing tasks and adding notes; it provides a workflow infrastructure; and it is a forum for forming new ideas. This allows everyone involved to be informed and engaged.

An important benefit to maintaining a shared issue tracker is that it helps answer the question “should we start new work or completely finish some nearly-finished work?” This is important because it is easy and tempting to start work, and it is much more difficult to completely finish work. Completely finished work has been and is tested, documented, validated by the customer, merged into a releasable code base, and deployed. In the lingo of git, this means merging your topic branch into the master or release branch. This is a critical piece of agile processes like Kanban, but also an important question for any project style.

A growing backlog of nearly-finished work is problematic in a number of ways. First, it represents a significant amount of future work that must get done. It may appear obvious, but if you wait till the end of the project to start finishing work then you can find yourself in big trouble. Second, the cost of completely finishing work typically increases as it waits in a development branch; the code base will evolve and it will be harder for developers and customers to recall the details of a change. Third, nearly-finished issues often become dependent on each other in complicated ways, making merging branches and even starting new work more difficult and time consuming. Finally, it is not easy to explain why demoed work is not completely finished—"but you showed it to me..." they might say—especially if there is no list of unfinished work. After a first demo, there is still a lot of work required before code is ready for release. Customers will have a better appreciation of focusing on unfinished work if they see it in the form of a growing list with a growing cost.

Bottom line: projects need a transparent way for organizing unfinished work and prioritizing it over new features.

ISSUE RADAR

One obvious answer for enumerating unfinished work is to look at the status of the issues in the issue tracker. The key to using an issue tracker for a workflow is to define the states that work can be in. For the project described here, we use a typical system V workflow, as seen in Figure 1.

![Figure 1. A depiction of a V workflow from wikipedia.com](image1)

We have a master and a next branch, which is why there is a “to be merged” state; issues in this state have passed QA and their topic branches should be merged into the master branch. Roughly speaking, issues in the range of active development and "to be merged" represent unfinished work and resolved issues represent finished issues; this is a good first cut at quantifying unfinished work.

Things become more complicated when you introduce issue dependencies. If issue XXX depends on issue YYY, then it is important to know that and organize work so that issue YYY is merged to master before XXX is needed. How do you keep track of this? It is possible for some small groups to keep these dependencies in their heads; for larger groups, this ad hoc methodology falls apart. It is also possible to specify relationships between issues in most issue trackers; this is not enough, however, because it is not automatic—people only do it when they remember and sometimes they forget or enter the wrong information.

Luckily, the git master/next workflow gives us the opportunity to keep track of outstanding work. Even better, with a couple of Python scripts, the process is automatic. We use git and MANTIS to create a diagram called issue RADAR – a diagram of unfinished issues and inter-issue relationships. Figure 2 contains an example.

![Figure 2. Issue RADAR diagram generated by git and MANTIS](image2)
Here are the details of how the issue RADAR is produced:

**Find next-only commits:** Create a list of all commits in next but not master. These commits represent all unfinished work; call these 'next-only commits'.

**Translate commits to issues:** For each next-only commit, find all issue numbers in the commit log. These numbers represent all unfinished issues; call these 'unfinished issues'.

**Compute issue dependencies:** For each unfinished issue, pretend to merge all of its next-only commits into master. This process will tell you what other commits, for different issues, must be merged before merging the current issue. Call these dependencies 'issue merge dependencies'.

**Graph results:** Create a directed graph where unfinished issues are nodes and issue merge dependencies are arrows. We look-up issue information such as lifecycle stage, issue summary, and priority from MANTIS in order to decorate the nodes of the graph, making it easier to comprehend. We also apply transitive reduction [5] to remove most of the relationship arrows (this means that if A depends on both B and C, and B depends on C, then simplify the graph by removing the A->C dependency; it is implied by A->B->C).

**Interpret graph:** The obvious heuristics for interpreting the graph and creating an action plan are to minimize the number of nodes (unfinished issues) and the number of edges (issue dependencies). Priority, developer availability, etc. will dictate the details.

**Communicate:** Send your customer (or developer) the graph and say: "I know you don't think we should spend time on XXX, but you really want YYY soon and the way things have worked out (point to graph), XXX needs to be done first."

The issue RADAR is not perfect. It requires you to add an issue number to every commit, which many projects do not do. On the other hand, we find that development teams that consistently organize branches and commits around issue numbers find the process to be lightweight and effective. Occasionally, we make mistakes with these messages and type the wrong number; we translate these issue numbers in the script that generates the daily RADAR. There are also some bad, old commits that we filter out of the process altogether. Also, it does not reveal "weak dependencies" where an issue XXX depends on a trivial part of issue YYY, for example, some basic refactoring early in a topic branch that could be merged to master without merging all of YYY.

**TECHNICAL SETUP**

We built issue RADAR using git, MANTIS, graphviz, and a few Python scripts to glue everything together. It would be easy to convert the process to another issue tracker that provides a programmatic (e.g., Python/SOAP) interface.

Git is a modern, distributed source code management tool that is similar in some ways to Subversion and CVS. One fundamental aspect of git is that it makes it easy to organize commits in branches and then merge that work into development or stable branches. The next-master workflow organizes work into separate topic branches that are merged into the next (development/testing) branch for demos and integration testing. After the work is completely finished, it is merged into the master (stable) branch.

MANTIS is a web-based, free, open-source issue tracker. It provides customization of workflow, issue information, as well as a secure login for customers and programmatic access via a SOAP interface. Using the programmatic interface, we decorate issues on issue RADAR with issue names, the workflow stage of issues, and issue assignments.


## CONCLUSION

Issue RADAR has improved our ability to plan work, communicate with development teams, and communicate with customers. It is a good way to measure and be transparent about unfinished work, and it helps to motivate that work while keeping customers happily involved in the process.

## REFERENCES


**Bill Hoffman** is currently Vice President and CTO for Kitware, Inc. He is a founder of Kitware, a lead architect of the CMake cross-platform build system and is involved in the development of the Kitware Quality Software Process and CDash, the software testing server.

**Brad Davis** is a technical leader at Kitware where he contributes to commercial and academic R&D efforts for the medical team.

## How Modularization Enables Organized Software Growth

The Insight Toolkit (ITK) was modularized recently as part of the work done for the upcoming release of ITKv4. Here we describe the process we followed in order to identify and execute modularization of the toolkit. We hope that this account may give insight to other software development teams on how to prepare for and undertake a modularization effort, particularly by highlighting the things that work well and the mistakes that could be avoided.

**WHAT IS MODULARIZATION?**

Modularization is the process of partitioning the code base of a software package into a set of smaller, well-defined packages, the modules. Some of the resulting modules will have dependencies on others. The minimization of dependencies between modules is a measure of success in a modularization effort. Homogeneity of size, making all modules contain about the same amounts of code, is a desirable feature, but not a requirement. In practice, it turns out that it is common to find that modules have a power-law-like distribution of size; that is, a few very large modules, and a multiplicity of medium and small size...
ones. Given that ITK is an object-oriented, C++ library, the quantum unit of the modularization is the “class” typically implemented as a pair of source code files (.h and .cxx, or .txx). For the purpose of the follow-up discussion here, we will refer to modularization as the effort of partitioning the files of a software package into modules.

The motivation for modularizing the Insight Toolkit came from observing the continued growth of the code base, both in terms of lines of code and in terms of scope and type of algorithms and classes. For the past ten years, the toolkit has been growing at an average of about 180,000 lines of code (loc) per year (that is ~15,000 loc/month, or ~500 loc/day).

The difficulty of code maintenance increases more than linearly with the number of lines of code, given that the interactions between different parts of the code increase exponentially. The very action of adding lines of code to the code base introduces the challenges of ensuring that the new code is consistent with the existing one, and that it doesn’t bring in any modifications that break existing functionalities. Over time, it becomes prohibitively expensive to support the quality control process required to continuously add code to a single pool package. The dreadful consequence is that quality is sacrificed as for every new line of code, new defects are introduced. Modularization mitigates this problem by restoring the independence between subsections of the code, and essentially converting a large software package into many medium size ones.

The next figures illustrate this point. Figure 3 shows the size distribution of ITK modules. The three largest modules are actually third-party libraries (VXL, GDCM, HDF5), while the ITKCommon module is the largest one. Figure 4 shows the size distribution of ITK modules excluding the third party libraries. The largest five are Core-Common, Nonunit-Review, Numerics-Statistics, Registration-Common, and Core-Transforms.

HOW TO MODULARIZE?
The modularization process aims to group together pieces of code (files) that have interrelated functionalities in the same module, while simultaneously minimizing the level of crosstalk between separate modules. In an ideal world, if we had information about all the dependencies and interactions between source code files, we could perform a graph-cut partition of the toolkit. Unfortunately, the automatic identification of dependencies is not a trivial problem. In our initial take, we attempted to mine the information that Doxygen gathers when it is building the class collaboration diagrams. We then explored the option of mining the database that SourceNavigator builds when it analyzes a software package, and also the use of the database created by ctags. After several weeks of effort on these fronts, we realized that such a problem was worth a Ph.D. dissertation, and that we needed a more pragmatic approach in order to match the schedule of our project. Therefore, we resorted to the manual classification of files into modules, supported by continued testing to identify whether any dependencies have been missed.

HOW TO MAINTAIN A MODULARIZED PACKAGE
Here, we are against the second law of thermodynamics. It is one thing to organize a room, and another to keep that...
defects and remove them from the system.

advantage of finer granularity for making it easier to detect

ied by design decisions and a software process that takes

software packaging exercise, but it also has to be accompa-

These observations show that modularization is not only a

principle of avoiding duplication of code that leads to create

"Law of Demeter" or the "Principle of least Knowledge." The modern

version of it in object-oriented programming is known as the

EXTREME MODULARIZATION

If we apply the modularization rationale to the limit, we reach the corollary that matches one of the oldest rules of a job well done: “Do one thing; do it right!” The modern version of it in object-oriented programming is known as the “Law of Demeter” or the “Principle of least Knowledge.” This leads to the fact that classes should have a minimum of dependencies between them. It is a trade-off with the principle of avoiding duplication of code that leads to create classes that provide services to other ones.

These observations show that modularization is not only a software packaging exercise, but it also has to be accompanied by design decisions and a software process that takes advantage of finer granularity for making it easier to detect defects and remove them from the system.

Mistake #1: When a new dependency is discovered (or introduced) between a file in module A and a file in module B, the quickest solution is to make the entire module A depend on the entire module B. Paradoxically, because the CMake setup we created makes it so simple to add a dependency between modules, this quick, self-defeating shortcut is an easy one to take. However, this must be avoided at all times. Every inter-module dependency that you add defeats the purpose of the modularization, and brings you back to the situation where maintenance will be too expensive and the quality of the code will degrade.

Mistake #2: As the modules become independent, they also tend to be maintained by a different subset of the development team and start diverging on style, design and practices. This is essentially a forking in the work that must be prevented by introducing automated tools for verification and by requiring code reviews by other members of the team. It is a truism of human nature that we continuously attempt to put our personal touch on what we create; unfortunately, that’s undesirable in large-scale software projects. Professional software developers must write code that others can take over. They must work every day to make themselves replaceable. It is that mindset that will guide the developer to: (a) generously comment code, (b) adhere to consistent style, (c) fanatically write tests for new code, (d) leave traces connecting to the discussions that led to the current code design. Good software developers write code that can survive once they are gone.

Many traumatic brain injury (TBI) cases involve blunt force damage resulting from the brain hitting the skull due to rapid acceleration/deceleration, e.g., during car accidents. The damage exists as lesions with hemorrhagic and nonhemorrhagic components. In order to plan treatment for TBI, patients typically receive an MRI scan on initial presentation in the clinic, and another one four-to-eight months later. The clinical goal is to distinguish areas in these scans that represent permanent damage from those that represent transient effects (see Figure 1). Determining the regions where the hemorrhage has receded can be particularly useful in predicting long-term outcome. However, the geometry of the lesion, the deformation of the brain due to swelling, the presence of the lesion, and the infiltration of the hemorrhage into the brain can cause drastic changes between these scans and confound the determination of correspondence and hemorrhage recession.
Using data from TBI patients from the University of California at Los Angeles, Kitware and the University of North Carolina at Chapel Hill have been developing registration algorithms that can help distinguish permanent damage from transient effects in longitudinal MRI scans.

Our methods can distinguish global tissue deformations, local changes in the geometry of a lesion, and local changes in the composition of the tissue and the lesion. We refer to this collection of changes as “geometric metamorphosis.” These changes are also evident in tumor cases where post-treatment assessment requires determination of changes in tumor geometry, tumor infiltration, scarring, and overall brain morphology. Similarly, in stroke cases, there is a clinical need to predict chronic changes in blood perfusion from acute scans.

Working as a team, we have produced a paper that details our algorithms and provides extensive experimental results. That paper has been accepted for publication at the Medical Image Computing and Computer-Assisted Intervention (MICCAI) conference to be held in Toronto, Canada, September 18-22, 2011. [Neithammer2011]

RELATED WORKS
Most image transformation models will fail in the presence of lesions that infiltrate or recede from healthy tissue and that deform the healthy tissue, as is common with tumors and other lesions. Direct application of a classical, deformable registration method will likely produce unrealistic deformation estimates. To overcome this limitation, deformable registration methods with weak and strong models of appearance change have also been proposed [3, 6]; proposed changes include models of tumor growth, for example. However, these methods are application-specific and degrade when their tumor growth model does not match the changes in the specific patient being registered.

Others [4, 7, 1] attempt to pre-mask areas in and around each lesion that cannot be matched. However, areas within these regions do not then contribute to the computation of the deformable registration. Therefore, registration in and around the lesion, where it may be clinically most important, may be poor. Our method explicitly includes a geometric model of the pathology so that its deformations can be explicitly captured in conjunction with the deformations of the underlying image.

METHODS
The basis of our work is that a registration method should be able to distinguish image appearance changes arising from the composition of (a) background deformations of the image with (b) foreground deformations of an embedded geometric object, e.g., a tumor. We model these transformations through a large displacement diffeomorphic metric mapping (LDDMM) [5]. LDDMM, however, is an inexact matching registration algorithm that only allows for the deformation of the source image and not for a change of its appearance.

In order to explicitly model appearance changes, we augment the LDDMM registration model with an extra control to model foreground deformation of a geometric model, which induces image appearance changes through an image composition model.

Specifically, we define the geometric metamorphosis problem as the minimization of:

\[
E = (1 - w) \int_0^1 ||v(t)||^2 \, dt + w \int_1^2 ||v(t)||^2 \, dt + \frac{1}{2} \sigma_1 \text{Sim}(I^r(t), I^c(t), T_0, I^r(t), T_2) \]

where \(I_0\) is the source image that is being warped to the target image \(I_c\), and \(I(1)\) represents the warped source image. \(T_i\) is the geometric model of the lesion in \(I_0\); \(T_0\) is the geometric model of the lesion in \(I_c\); and \(I'(1)\) is the warped \(T_r\); \(v\) is the background (lesion) deformation field. \(v_c\) is the background (lesion) deformation field. Note that the geometric model \(T_c\) and its image \(I_c\) are subject to both deformations, whereas the source image is only subject to the background deformation. \(L\) is a differential operator that controls the spatial regularity of these deformation fields. \(σ\) controls the weighting of the image match term. \(w \in (0, 1)\) controls the trade-off between background and foreground deformations. \(\text{Sim}\) denotes a similarity measure of choice; we use the \(L_2\) distance measure. Two similarity terms are used to assure matching of (a) the regions that correspond in both images and (b) the geometric models. \(I'(•, •, •)\) denotes the image composition model. Additional detail is given in the full version of our paper [2].

REFERENCES
**Gabe Hart** joined Kitware in June, 2010 as an R&D engineer in the medical imaging group. Prior to joining, he studied medical image analysis at UNC, Chapel Hill. He splits his time between the ITKVideo and SimpleITK groups in the ITK v4.0 team and the NAMIC algorithms research team.

**Marc Neithammer** is an Assistant Professor at the University of North Carolina in the Department of Computer Science and the Biomedical Research Imaging Center. He received his Ph.D. in Electrical and Computer Engineering from the Georgia Institute of Technology and post-doctoral training at Harvard Medical School/Brigham and Women’s Hospital.

**Danielle Pace** is a Research and Development Engineer in Kitware’s medical imaging team. She joined Kitware’s North Carolina office in July 2010, and works on both research in deformable image registration and software engineering for 3D Slicer and Kitware’s commercial consulting projects.

**Jack Van Horn** is an Assistant Professor of Neurology at UCLA. He completed his doctoral work at the University of London, he has over 20 years of experience in the field of human neuroimaging, neurology, cognitive neuroscience, and large-scale data analysis and visualization.

**Stephen Aylward** is Kitware’s Director of Medical Imaging Research. He is also an adjunct professor at UNC, an Associate Editor of IEEE transactions on Medical Imaging, and PI for a wide range of NIH-funded research efforts.

**Editions of the ParaView Guide Coming Fall 2011**

Kitware is preparing to release the latest version of the ParaView Guide in Fall 2011. Since the previous release, the ParaView Guide has been updated to cover ParaView 3.10. The guide has also been expanded to include discussion about the theory behind Paraview and contains new topics. The print version of the book will feature additional chapters on specific use-cases of ParaView.

Unlike previous editions, the ParaView Guide for 3.10 is also available for free online in a public wiki. The guide is also included for free with the latest version of ParaView.

**Where Does Your Money Go?**

In April, Google announced their Visualize Your Taxes challenge, which uses data visualization to show people where their tax money goes. Out of 40 entries, six were selected as finalists, including one by Kitware’s Jeff Baumes.

Jeff’s visualization centered on using a calendar to let people see how their money is distributed throughout the year. His interactive calendar can be found at [http://whatdoyouworkfor.appspot.com/index.html](http://whatdoyouworkfor.appspot.com/index.html).

**VTK/MIDAS Course at Vega Technologies**

In April, Kitware’s Julien Jomier taught a course on VTK and MIDAS at Vega Technologies in Toulouse, France. Vega is a small subsidiary consulting and engineering company working in the air and space field. Owned by VEGA Space Ltd, Vega Technologies aims to provide software for satellite operation and visualization, in addition to work on GPU programming for fast visualization of terrains.

The two-day VTK course was attended by four research and development engineers who came with some prior knowledge of the toolkit. Topics covered included an overview of VTK architecture, visualization pipelines, information visualization methods, how to write custom filters, and parallel processing and rendering techniques.

The third day was the first MIDAS course given out of Kitware’s European office. MIDAS was met with great enthusiasm, as Vega Technologies deals with terabytes of maps and 3D terrains that require integration with their applications.

If you are interested in one of our courses, please visit our website to see our full range of offerings.

**Kitware Talks Open Source at the University of Houston**

In April, Luis Ibanez was invited by Badri Roysam, chair of Electrical and Computer Engineering at the University of Houston, to give a series of talks about open-source software practices, ITK, and open-source image processing, and present a hands-on course on the use of ITK.

The Open Source Software Practices talk was attended by about 50 people, and covered why and how open source development works. The course on ITK was attended by close to 100 people and was standing room only. The hands-on ITK session ran for two hours and included the students’ discovery of a bug in one of the examples in ITK.

Dr. Roysam is poised to make the University of Houston an open source hub in Texas and is planning a number of upcoming open source activities and courses.
KITWARE AND VICARIOUS VISIONS GIVE FIRST TECH VALLEY TALK
On May 17th, Kitware held its first “Tech Valley Talk” with co-hosts Vicarious Visions, a video and computer game development firm best known for their work on Guitar Hero.

The talk was led by Bill Hoffman, VP and CTO at Kitware, and Nicholas Ruepp, a Producer at Vicarious Visions, and focused on Agile Programming Practices. They discussed how agile practices are implemented at each company. Nicholas highlighted how Vicarious Visions uses agile practices to maintain a consistent level of effort throughout production, rather than excessive swings often experienced with a waterfall processes. Bill spoke about how Kitware implements agile programming by starting from a key text and adapting the methodologies to fit the needs of the company.

The event was attended by about 70 people from Kitware, Vicarious Visions, local technology companies, and local universities. Kitware hopes to host Tech Valley Talks quarterly and is actively seeking potential collaborators.

KITWARE HOSTS LUNCH AND LEARN SEMINAR
In early May, Kitware hosted a lunch and learn seminar called “Innovate, Collaborate and Reduce Costs with Open-Source Solutions” at the North Carolina Biotechnology Center in Research Triangle Park. The seminar was attended by people from a range of businesses in the Research Triangle region including RENCI, Synexis, Geomagic Inc., and Duke University.

Will Schroeder presented on open-source solutions from a high-level point of view, and provided four specific case studies to further clarify how open source can address today’s business challenges. Brad Davis spoke and presented case studies on MIDAS and Slicer, and Lisa Avila discussed case studies on CMake and ParaView.

After presentations, the floor was opened for questions and discussion. There was considerable enthusiasm around Kitware’s software process, the CMake/CTest/CDash suite, and work being done in the medical field.

KITWARE ATTENDS THE 6TH INTERNATIONAL OPENFOAM WORKSHOP
The 6th OpenFOAM Workshop was held June 13-16th at Penn State University. Kitware sponsored this event and Dave DeMarle attended and presented a tutorial on Advanced ParaView usage for CFD researchers. The tutorial was a demonstration of some of the recent capabilities available in ParaView 3.10. These include those made possible by Takuya Oshima’s overhaul of the vtkOpenFOAMReader, such as native support for parallel cases, as well as some of ParaView’s quantitative analysis capabilities such as temporal data analysis, plotting forces acting along arbitrary features, material and flow analyses, and some advanced rendering techniques provided by ParaView plugins.

The rest of the conference was equally constructive. Impressive papers included a demonstration and scaling study of analysis on Jaguar, the 3rd fastest supercomputer in the world; a simulation of odorant transport in the canine olfactory organs; the creation of a more flexible computational model and its application to the problem of modeling anaerobic digesters; and an industrial design optimization using open source tools. This last paper presented an effort in which a suite of open source tools spanning the entire preprocessing, simulation, and post processing pipeline were brought to bear on different design projects and in which genetic algorithms were used to tune the design parameters to maximize utility.

Finally, as a newcomer to this particular open source community, Dave was struck by how similar the larger problems are, such as the need to increase the availability of high quality learning materials; to facilitate increased contributions from a wider development community; to speed development, testing, and integration generally; and a desire to reconcile competing development camps.

OPENFOAM® is a registered trade mark of OpenCFD Limited, the producer of the OpenFOAM software.

KITWARE SELECTED AS TOP SMALL COMPANY WORKPLACE BY WINNING WORKPLACES
Kitware has been selected by Inc. magazine as one of Winning Workplaces’ 2011 Top Small Company Workplaces competition. The annual contest recognizes the nation’s 50 best small and mid-sized company work environments in the magazine’s June issue.

Since 2003, Winning Workplaces, an Evanston, Ill.-based nonprofit organization, has recognized firms that attribute much of their business success to their commitment to exemplary people practices and outstanding workplace cultures. This year, Winning Workplaces received nearly 350 completed applications for the contest. A national panel of experts in leadership and small to mid-sized business judged the finalists based on specific metrics and qualitative assessments of the finalists’ success in creating workplace that engage employees and deliver bottom-line results.

In addition to creating cutting-edge toolkits and developing advanced research solutions in the fields of scientific visualization, informatics, data management, quality software process, medical imaging, and computer vision, Kitware strives to provide a collaborative and supportive work environment. By offering great benefits within an open source business model, the unique culture has earned a spot in the 2011 Top Small Company Workplaces.

“We are honored to be recognized as one of this year’s Top Small Company Workplaces,” said Will Schroeder, CEO of Kitware. “Making this list affirms that we have an empowered, collaborative culture, and that our team is solving some of the world’s most challenging technology problems while having lots of fun along the way.”
UNC STUDENTS COMPETE USING PARAVIEW
ParaView was recently used in Russell Taylor’s Comp 715: Visualization in the Sciences class at the University of North Carolina at Chapel Hill. The course is aimed at both computer science students whose future work will include design and implementation of visualizations, and natural science students who will use them.

As part of the course, students are asked to develop a visualization tuned to a particular problem by using a toolkit or custom code. Working in teams of two, students used the ParaView visualization application to generate images.

Movement and development of rotating stall cells based on three different turbulence models. Data courtesy of the 2011 IEEE Visualization Contest. Images courtesy of Laura Miller, UNC Applied Mathematics

ITK FOR IMAGE REGISTRATION IN MICROSCOPY
Luis Ibanez visited the San Diego Supercomputer Center to participate in a hack-a-thon with Kitware collaborators from the International Neuroinformatics Coordination Facility (INCF) Atlasing Task Force. The INCF is an international organization that aims to foster the sharing of resources among neuro-scientists. They are particularly concerned with data-sharing and the development of computational resources.

With labs around the world gathering experimental data on mouse brain studies, there is a need to register those datasets to a common reference system, typically in the form of a brain atlas. INCF has prepared a set of image datasets to serve as such an atlas.

During the hack-a-thon, the group advanced towards the goal of incorporating ITK-based image registration capabilities with web services developed by the San Diego Supercomputing Center. This combined resource is intended to be made publicly available to neuroscience research labs to make it easy for them to take the datasets that they have acquired and register it to the INCF Atlases.

For this project, ITK-based registration methods will be interfaced with a computational infrastructure that includes data servers based on iRODS and computer clusters. Unlike other projects, this collaboration is not asking for methods that are new, original, or mathematically flashy. Instead, the requirements are that it work well, be maintainable, and be open; therefore, this project is a natural fit for the functionalities and characteristics of ITK.

KITWARE EDGE DEBUTS
In June, Kitware released the first issue of a new monthly, digital newsletter, the Kitware Edge. The newsletter highlights Kitware news, upcoming events, project updates, and inside stories to keep you on the cutting edge of what is happening at Kitware. Visit www.kitware.com/products/edge today and sign up for the Kitware Edge to get the inside scoop delivered to your inbox each month.

KITWARE TO HOLD ONLINE COURSES
Kitware has announced four upcoming introductory courses for our toolkits. Current offerings cover VTK, CMake, ITK, and ParaView and begin in early September. There are no prerequisites, though a basic knowledge of C++ is necessary to fully benefit from the included programming examples.

Introduction to VTK
Wednesday, September 7th, 1:30-3:00 PM
This course covers the various data types supported by VTK, the pipeline architecture used for processing data, and the rendering framework used to display results. Attendees gain hands on experience through several programming exam-
Equally varied were the participants who included artists, proteins, to visualizing cells, organisms, and populations. Included everything from visualizing the genome and all forms of biological visualization. The aims to bring together a broad group with interests in any single modality or a narrow field of study, VIZBI was unique and interesting workshop. Rather than concentrating on a single modality or a narrow field of study, VIZBI was an excellent mechanism for meeting new contacts and to hold one-on-one discussions. Overall, the forum was an excellent mechanism for meeting new contacts and exploring new opportunities.

VIZBI 2011

From March 16-18th, Wes Turner attended VIZBI 2011, a workshop exploring the visualization of biological data. The Broad Institute in Cambridge, Massachusetts served as an excellent and stylish venue for what turned out to be a unique and interesting workshop. Rather than concentrating on a single modality or a narrow field of study, VIZBI aims to bring together a broad group with interests in any and all forms of biological visualization.

Presentations spanned from the micro to the macro and included everything from visualizing the genome and proteins, to visualizing cells, organisms, and populations. Equally varied were the participants who included artists, animators, and documentarians along with the more usual scientists and engineers. Notable among the presentations was a documentary video on the lifecycle of the malaria parasite given by Drew Barry, and a presentation on using the web to explore cultural data given by Google employees Martin Wattenberg and Fernanda Viégas. Kitware participated in the tutorial sessions and presented data visualization on the macro and informational levels using ParaView and VTK.

KITWARE TALKS COLLABORATION AT NESCENT

The National Evolutionary Synthesis Center (NESCent) in Durham, N.C., hosted a workshop on “Cyberinfrastructure for Collaborative Science” from May 18-20th. The mission of NESCent is to sponsor interdisciplinary and synthetic research to address fundamental questions in evolutionary biology. This workshop brought together a wide range of expertise to explore the mechanisms by which collaborative research can be enabled and improved using modern, collaborative, and open source tools. Kitware’s Stephen Aylward and Wes Turner attended as experts in open source collaborations and scientific visualization. The workshop used the “unconference” paradigm and attendee consensus drove towards the exploration of how technology assistance and career incentives could be used to drive improved open data and shared research archives.

UPCOMING CONFERENCES AND EVENTS

SIGGRAPH 2011
August 7-11, in Vancouver, BC, Canada. This is the 38th international conference and exhibition on computer graphics and interactive techniques. Aashish Chaudhary will be in attendance. http://www.siggraph.org/s2011/

Mil-OSS: Military Open Source Software
August 30–September 1, in Atlanta, Georgia. Mil-OSS believes in adopting open technology philosophies for military applications. Chuck Atkins will in attendance. http://mil-oss.org/wg3-overview

MICCai 2011
September 18-22, in Toronto, ON, Canada. Andinet Enquobahrie will be leading the Systems and Architectures for Computer Assisted Interventions workshop. Kitware will also be participating in “Your Career in Industry and Academia,” an informal interactive session for students to discuss career opportunities with leading organizations. Andinet Enquobahrie and Stephen Aylward will be in attendance. http://www.miccai2011.org/

BioVis 2011
October 23-24th, in Providence, Rhode Island. This symposium discusses visualization of biological data. Wes Turner will be in attendance. http://www.biovis.net

VisWeek 2011
October 23-28, in Providence, Rhode Island. This year marks the 22nd VisWeek conference for advances in academic, government, and industry. Visit http://www.visweek.org/

RSNA 2011
NEW EMPLOYEES

Katie Sharkey
Katie joined Kitware as a member of the Communications team in May. She received a bachelor’s degree from the State University of New York at Potsdam, and a master’s in Public Communications from The College of Saint Rose.

Lai-Yee Burnham
Lai-Yee joined Kitware as a member of the of the Accounting team in May. She received her associate’s degrees from Hudson Valley Community College in Individual Studies and Business. Lai-Yee is currently working on her bachelor’s degree from the City University of New York.

Ilknur Kabul
Ilknur joined Kitware as a member of the medical research staff at the Carrboro office in June. She is a current Ph.D. student at the University of North Carolina at Chapel Hill. Her dissertation focuses on patient-specific atlas generation using model-guided texture synthesis. Ilknur also holds a master’s degree from Bilkent University, and a bachelor’s degree from the Middle East Technical University, both in Ankara Turkey.

Joseph Snyder
Joseph joined Kitware as a member of the medical research staff at the Clifton Park office. He received his bachelor’s degree in Biomedical Engineering from Rensselaer Polytechnic Institute in May 2011. Joseph has experience working for Kitware as an intern in June and December 2011.

KITWARE INTERNSHIPS

Kitware Internships provide current college students with the opportunity to gain hands-on experience working with leaders in their fields on cutting-edge problems. Our business model is based on open source software, which makes for an exciting, rewarding work environment.

Kitware is pleased to welcome 15 interns this summer. The 11 interns at the Clifton Park office span the medical, computer vision, and human resources teams and include Wendel Silva, Jorge Poco, Emily Rayfield, Lauren Bange, Ilseo Kim, Kishore Reddy, Chia-Chih Chen, Kang Li, Tor Hagemann, Zachary Clapper, and Ahmed Tashrif Kamal.

At the Carboro office, Huai-Ping Lee, Istvan Csapo, Alberto Vacarella, and Nathan Taylor will spend the summer assisting the medical research team.

We are actively recruiting interns for the fall semester and for long term internships. If you are interested in applying, please send your resume to internships@kitware.com.

EMPLOYMENT OPPORTUNITIES

Kitware is seeking talented, motivated and creative individuals to become part of our team. As one of the fastest-growing companies in the country, we have an imme- diate need for software developers, especially those with experience in computer vision, scientific computing, and biomedical imaging.

At Kitware, you will work on cutting-edge research problems alongside experts in the fields of visualization, medical imaging, computer vision, 3D data publishing and technical software development. Our open source business model means that your impact goes far beyond Kitware as you become part of the worldwide communities surrounding our projects.

Kitware employees are passionate and dedicated to innovative open-source solutions. They enjoy a collaborative work environment that empowers them to pursue new opportunities and challenge the status quo with new ideas. In addition to providing an excellent workplace, we offer comprehensive benefits including: flexible hours; six weeks paid time off; a computer hardware budget; 401(k); health, vision, dental and life insurance; short- and long-term disability, visa processing; a generous compensation plan; profit sharing; and free drinks and snacks.

Interested applicants are encouraged to send their cover letter and resume to jobs@kitware.com.

In addition to providing readers with updates on Kitware product development and news pertinent to the open source community, the Kitware Source delivers basic information on recent releases, upcoming changes and detailed technical articles related to Kitware’s open-source projects, including:

- The Visualization Toolkit (www.vtk.org)
- The Insight Segmentation and Registration Toolkit (www.itk.org)
- ParaView (www.paraview.org)
- The Image Guided Surgery Toolkit (www.igstk.org)
- CMake (www.cmake.org)
- CDash (www.cdash.org)
- MIDAS (www.kitware.com/midas)
- BatchMake (www.batchmake.org)

Kitware would like to encourage our active developer community to contribute to the Source. Contributions may include a technical article describing an enhancement you’ve made to a Kitware open-source project or successes/lessons learned via developing a product built upon one or more of Kitware’s open-source projects. Authors of any accepted article will receive a free, five volume set of Kitware books.

Kitware’s Software Developer’s Quarterly is published by Kitware, Inc., Clifton Park, New York.


Graphic Design: Steve Jordan
Editor: Katie Osterdahl, Katie Sharkey

To contribute to Kitware’s open-source dialogue in future editions, or for more information on contributing to specific projects, please contact the editor at editor@kitware.com.

This work is licensed under a Creative Commons Attribution-NoDerivs 3.0 Unported License.

Kitware, ParaView, CMake and VolView are registered trademarks of Kitware, Inc. All other trademarks are property of their respective owners.