ParaView: A Sustainable Success Story

Berk Geveci, Patrick O’Leary, Utkarsh Ayachit, Will Schroeder
Kitware Inc.

This whitepaper describes the stewardship history of one of DOE’s premier visualization tools, ParaView [1]. We aim to describe one successful sustainability and community model and draw conclusions that are hopefully beneficial to other projects and the leadership in making future decisions. ParaView is a visualization tool that was originally developed for the post hoc analysis of large datasets produced by large-scale scientific simulations. Over time, it has also been extended to handle a larger set of use cases. It was mainly funded to support the DOE mission, but over time it has received funding from various sources as well as many contributions from a broader community.

Brief History

ParaView heavily leverages the Visualization Toolkit (VTK) [7] for much of its functionality. Therefore, the best place to start is a brief history of VTK from its inception to ParaView’s inception. VTK was conceived in 1993 as an open source, cross-platform library for general data analysis and visualization. It was written to accompany the Visualization Toolkit book [6]. As the leading C++ library for visualization at the time, it attracted a community of users and developers. Furthermore, several organizations, including GE, adopted it and contributed resources to its development and maintenance. The developers of VTK recognized early on the importance of strong software practices in developing and maintaining a complex platform such as VTK. As such, VTK is one of the first large-scale C++ projects to adopt open version control (initially CVS), a cross-platform build system, regression and continuous integration testing, and automated documentation generation. It is one of the earliest projects to adopt cross-platform build with CMake, automated testing, and reporting with CTest and CDash. In fact, its testing implementation predates any of these tools as described in [3]. After 28 years, VTK continues to thrive and grow.

In 1999, researchers at the Los Alamos National Laboratory started developing distributed memory visualization capabilities. They had the foresight not to reinvent the wheel and decided to build upon an existing open source platform. VTK was chosen as the best of the breed and it was decided to follow a two-pronged approach: 1) expand VTK to support distributed memory parallelism; 2) develop an end user application built on VTK. The second effort led to the development of ParaView.
Figure 1 shows a basic timeline of ParaView. The project's first five years were focused on building distributed parallel visualization capabilities and a basic user interface. The main driver for the project was the vision of the Los Alamos researchers who were later joined by those from Sandia National Laboratories. This focus made it easier for the development team to push on core functionality and achieve a foothold among DOE users who needed large data visualization capabilities.

After the initial development phase, the development team expanded the ParaView user community by improving usability by designing a new user interface and deriving new applications such as Computational Model Builder (focused on pre-processing to generate simulation inputs), Tomviz (focused on TEM data visualization) and ParaView on the Web. These improvements opened access to larger user and developer communities. At the same time, the ParaView team continued to innovate by introducing new capabilities such as Catalyst [2] for in situ analysis, integration with VTK-m [4] and DIY [5] for more advanced distributed and shared-memory parallel algorithms, and integration with advanced rendering libraries such as OSPRay and OptiX.

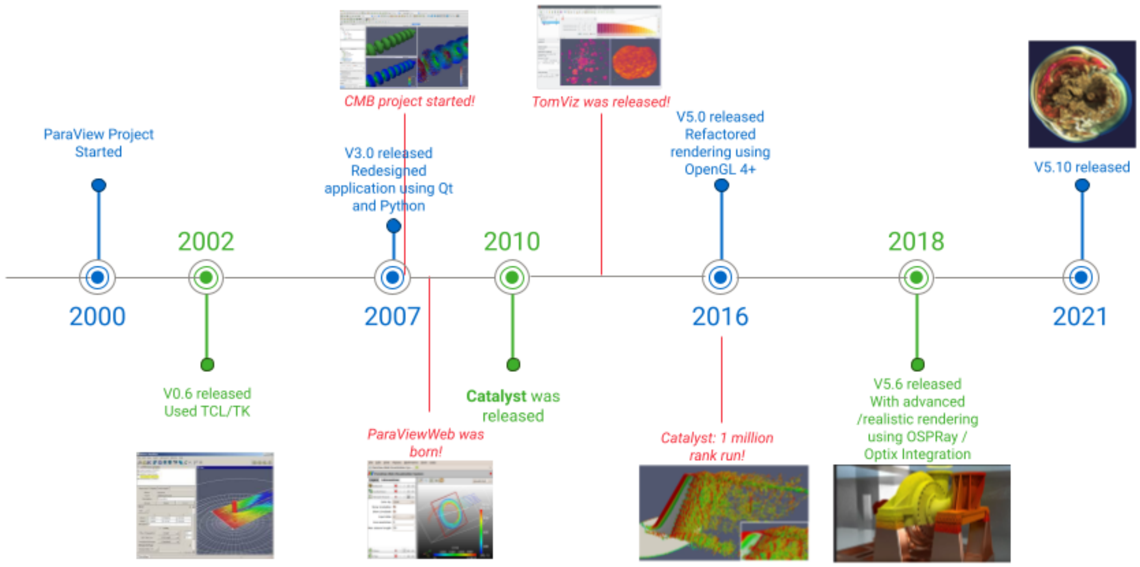

Figure 1. ParaView Timeline

Factors of Success

In this section, we try to identify factors behind ParaView's success and longevity.

Provide value early. The faith of a project such as ParaView is tenuous at best in the beginning. The project sponsors often expect an early demonstration of feasibility. ParaView was able to deliver because 1) it already had a strong foundation in VTK, a mature visualization library, 2) the development team was highly focused and motivated, and 3) there was sufficient funding for a strong push. We built a fully functioning prototype in less than two years and started building a user community.

Broaden Community. When the project started in the early 2000s, the customer community for an HPC visualization tool was small. If the ParaView team had focused only on this community, it would have had difficulty gaining enough momentum to be a sustainable platform. Therefore, emphasis was placed on broadening its reach by redesigning the user interface, extending its functionality to a more general-purpose tool, and building new applications using the same foundation.

Open development. ParaView was an open project from the beginning. In addition to being open source, the core development team was open to and encouraged community contributions. This had a significant impact on the project by growing the project's development resources and enabling partnerships that supported the project's diverse funding resources.

Continuous innovation and growth. HPC, scientific and engineering simulation and visualization have continuously and rapidly evolved. Some great examples of disruptive innovation in these domains are the rise of distributed computing, general-purpose GPU computing, and the impact of AI and deep learning on computing. ParaView was born out of the rise of distributed computing. However, it would not have been a sustainable project if it did not adapt as the other disruptive technologies emerged. For example, in response to the growing discrepancy between our ability to compute and move data, the ParaView team developed Catalyst, a general-purpose in situ library. In response to the rise of general-purpose GPUs, the ParaView team, with many partners, founded the VTK-m project.

Software practices. As the previous sections and its history hopefully demonstrate, ParaView is a rapidly changing software. It received contributions from a broad set of developers with diverse backgrounds. It would not have been possible to maintain a stable product without strong software practices, let alone grow the quality and stability over time. We were fortunate to start with a strong foundation, benefit from the efforts of synergistic projects, and have project sponsors that appreciated the need for investing in the software infrastructure.

Diverse funding stream. ParaView's stewardship enabled access to a diverse funding stream that made the above possible. ParaView's core development team is part of Kitware Inc., a company that was founded based on the Visualization Toolkit. Our position has provided us access to a broad range of funding sources. Core funding for ParaView was provided by Los Alamos and Sandia National Laboratories under the ASC program. We also participated in core ASCR programs such as SciDAC and ECP. Furthermore, we worked with other government agencies and the industry to expand ParaView further. Kitware, as a small business, has access to the SBIR/STTR program, which funded several innovative projects without which ParaView would not have succeeded. These include the Catalyst in situ library, ParaView on the Web, and many core changes.

Conclusions

In this section, we attempt to generalize some of the lessons we learned from ParaView's sustainability model. We hope that these will help guide other projects and leadership decisions.

For many DOE projects, the first few years are critical. We have seen many projects disappear before they could cross this threshold. DOE being a mission-driven department, it expects from software projects early results that demonstrate impact on the mission. Furthermore, without a user (and also hopefully developer) community, the project can struggle to clarify its mission. The following are very helpful in successfully reaching early milestones and building a community.

- A narrow mission focused on the needs of a community, and with the intention of growing an active community. For ParaView, this was data analysis and visualization at scale (distributed parallelism). It was a niche area with a tangible need and community aligned with DOE's mission.
- A driven and focused developer team. It is very challenging to get an HPC software project off the ground. A team that is motivated to succeed and that can dedicate a big portion of its energy towards success is critical.
- A stable funding source. This venture has to be adequately and stably funded. Funding timelines with gaps are significant barriers to building and maintaining focused teams. Under-funded projects rarely reach a stable user community because they cannot respond to its needs.

Once a project crosses this initial threshold (often referred to as the "valley of death"), the initial funding source is likely to either taper off or to concentrate on developing the software to satisfy very program-specific needs. At this point, if the project has strong roots and can be shared by a larger community (open source, for example), it may be feasible to start diversifying it to access a broader user base and a broader set of funding sources. However, this requires the appropriate investment. At this point in ParaView's history, we were lucky enough to have access to DOE and DoD SBIR/STTR funding to make it possible. This kind of funding is not always available to the development team. The original project sponsors may want to provide this additional investment to ensure the long-term viability of the project - even though the immediate benefit may not be obvious.

The HPC software ecosystem is continually evolving to respond to changes in hardware and to new innovations in software. Software projects need to respond to these changes if they want to remain relevant. Therefore sustainability is often not about staying steady but about moving forward to maintain relevance. The ParaView project was able to sustain itself by adapting through innovating new capabilities such as Catalyst for in situ computing and VTK-m for data analysis and visualization on GPUs. Among early career computer scientists, there is a tendency to develop new frameworks rather than innovating through existing ones. This is sometimes harmful to established projects by diverting resources and community. This behavior is also driven by certain funding mechanisms that encourage innovating through new platforms. The ParaView project was able to navigate these issues by encouraging its adoption as well as by integrating emerging technologies (such as VTK-m).

The funding mechanisms to successfully go through the stages mentioned above are not always available or attainable. Much of the existing mechanisms are focused on the first stage - getting the project off the ground - but not diversification or continuous innovation. It is then left up to the project teams to navigate various funding mechanisms to achieve sustainability. We encourage program management to take these aspects into account when developing software stewardship programs in addition to more obvious aspects of sustainability such as community support and software quality processes. Furthermore, a healthy software ecosystem can be best achieved by a coordinated approach across multiple funding mechanisms such as programs like SciDAC and ECP, research programs and the SBIR/STTR program.

References

[1] Ahrens, James; Geveci, Berk; Law, Charles, *ParaView: An End-User Tool for Large Data Visualization*, Visualization Handbook, Elsevier, 2005, ISBN-13: 978-0123875822
[2] Ayachit U.; Bauer A.; Geveci B.; O'Leary P.; Moreland, K.; Fabian, N.; Mauldin, J. , *ParaView Catalyst: Enabling In Situ Data Analysis and Visualization*, in Proceedings of the Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization, 2015
[3] Hibbard, Bill; Lorensen, Bill; Miller; Jim, *Visualization toolkit extreme testing*. SIGGRAPH Comput. Graph. 35, 3 (August 2001), 8–11. DOI:https://doi.org/10.1145/601782.601785
[4] Moreland, K.; Sewell, C.; Usher, W.; Lo, L.-T.; Meredith, J.; Pugmire, D.; Kress, J.; Schroots, H.; Ma, K.-L.; Childs, H.; Larsen, M.; Chen, C.-M.; Maynard, R.; Geveci, B. *VTK-m: Accelerating the Visualization Toolkit for Massively Threaded Architectures*. IEEE Computer Graphics and Applications, 2016, *36*(3), 48–58. https://doi.org/10.1109/MCG.2016.48
[5] Morozov, Dmitriy; Peterka, Tom. *Block-parallel data analysis with DIY2*. IEEE 6th Symposium on Large Data Analysis and Visualization (LDAV), 2016 29-36. 10.1109/LDAV.2016.7874307.
[6] Schroeder, Will; Martin, Ken; Lorensen, Bill (2006), *The Visualization Toolkit* (4th ed.), Kitware, ISBN 978-1-930934-19-1
[7] The Visualization Toolkit, https://vtk.org