Transactional Parallel Metadata Services for Integrated Application Workflows

Jay Lofstead[1], Jai Dayal[2]
[1]CSRI, Sandia National Laboratories
[2]CERCS, Georgia Institute of Technology

ABSTRACT

Scientific simulations have a different relationship with all of the data generated than many data analysis systems that support applications like the Large Hadron Collider and the SLOAN Sky Survey. In many cases, simulations need to generate large number of intermediate data sets that ultimately are thrown away once some analysis routines are applied to the data. This generates some summarized, derived result that inspires some scientific insight. Traditionally, these routines use the storage array to persist the intermediate results between each step of the data analysis process. The volume and frequency of this data can be overwhelming compared with the available IO bandwidth on the machine. To handle this volume and frequency, current research efforts are determining how to move the storage of intermediate data from the storage array into the memory of the compute area. Then, the analysis routines are incorporated to create Integrated Application Workflows (IAWs). Data staging techniques require some mechanism to replace the semantics offered by the file system to control data movement and access. As part of an HPC-focused transaction services project, a first pass at a transactional metadata service for in compute area data storage is being developed.

1. INTRODUCTION

Many applications have shifted their data management focus from an ACID compliant transactional technique to use less strong semantics. For these applications, incomplete and partially out of date results are good enough. For example, queries against a dynamically growing data set need not be 100% acurate nor complete to be useful. Instead, response speed is a significant contributor to the perceived quality of the tool. This is bolstered by an assumption that a subsequent query will provide more up-to-date results. However, this sort of partial completeness model does not suit all application domains.

HPC applications come in many flavors. A popular group of HPC applications use some data source, typically a physical instrument of some sort, to collect a massive data set. Once this data is collected a series of analysis routines are performed on this static data set. In these cases, the data set is fixed and permanently stored. The SLOAN sky survey is a classic example of this kind of application. Another large category are scientific simulations that use models to represent some phenomena in an effort to better understand how something works or to learn how to introduce new efficiencies.

In this second category, the simulation implements a series of formulas that are applied across a simulation domain to model some physical phenomena. Typically, these applications run cyclically slowly progressing the physical processes being modeled in an effort to represent some real-world process. At periodic increments, the dataset is dumped to give a snapshot of the current state of the simulation offering an opportunity to evaluate the current state of the simulation and the progression of the model. The collection of these snapshosts can be used with some analysis routines to make new scientific discoveries. Unlike the previous category, this intermediate data is generally discarded once the analysis is complete. Rather than maintaining the data set for future experiments, these intermediate data sets are simply the necessary components of the discovery process. The actual desired outcome is some new understanding that may influence some engineering application, mathematical model, or some basic understanding of a physical process. Given current technology, this intermediate data is stored in a centralized storage array in between each step of the subsequent analysis routines. Given the faster growth of compute capacity than IO bandwidth, alternative approaches to enable this workflow are being explored.

Integrated Application Workflows (IAWs) offer the ability to host data storage within the compute area for subsequent data analysis routines. These efforts are commonly grouped under the name of 'data staging' projects. Several current efforts [1–3] are exploring how and where to host the analysis functionality as the data moves from the computation to storage. In all of these cases, the quality of semantics offered by the storage system is lacking. For example, there is no way to make the equivalent of a 'write lock' to prevent access by other processes. There is also no way to block the visibility of a data set prior to it being validated as both complete and correct. For IAWs to achieve any large adoption rate, these shortcomings must be addressed.

The doubly distributed transaction services [4] offer these sorts of guarantees to the data movement operations for these data staging systems, among other scenarios. This prevents premature access to a data set prior to data being complete and correct and keeps analysis routines from incorrectly attempting to process a partial data set leading to either a failure or incorrect result due to the missing data. These transactional services are also being applied to system reconfiguration operations to help manage the availability of components by only allowing data flows to shift once the new resources have been successfully deployed. While both of these scenarios are useful by themselves, the additional service required to make these services fully useful for an IAW is a metadata service that is both transaction and parallel client aware. Traditional database engines are generally transaction aware, but work under the assumption that the client is a single process working against a collection of potentially distributed server resources. Even systems like Zookeeper that implement the Paxos algorithm work well for distributed clients, but not for parallel clients acting as a single agent.

Parallel clients introduce a level of coordination such that only when all clients have finished providing their input on the operation can the total operation across the distributed servers be completed. For transactions, this means that there needs to be consensus among not just the servers, but also the clients for a commit or a rollback is required. Existing metadata systems do not support this sort of functionality. Parallel file systems typically work by having a single process create a file and then all of the parallel clients open the file. There are also assumptions of conflict avoidance/prevention. While it is possible to use the create/open approach to adapt a parallel file system metadata system, the other semantics do not work well for this online use. Further, the transaction support would have to be integrated into the existing system. Given the extent of changes, creating a simplified, streamlined metadata service makes more sense than trying to make extensive changes to existing systems.

More broadly, the transaction system is intended to work across not just local compute resources, but also incorporate potentially widely distributed components. This requires a variety of simultaneous different networking technologies all working in concert on a single transaction. This paper explores the requirements and initial implementation efforts to build such a metadata system to support IAWs and the current implementation status of a prototype system.

The remainder of this paper is organized as follows. Section 2 presents a short overview of the related work. We introduce the metadata requirements in Section 3. A working example of how this system would be used in an HPC environment is presented in Section 4. We next present our current implementation status in Section 5. Section 6 presents our conclusions.

2. RELATED WORK

Metadata systems are nothing new. In particular file systems have used a metadata system to track the contents of the storage since early days. Transactions have been applied to distributed file systems in the past [5, 6]. However, these distributed file systems are focused on the single client model or are limited to just file system transactions. The system we are building focuses specifically on distributed clients using an online storage area as part of a larger transaction. The combination of multiple clients and broader operations as part of the transactions makes this system quite different.

In recent years, metadata system research has shifted the focus to making metadata scalable for parallel file systems. In these cases, the number of files in a single directory, simultaneous file creations, and similar operations have been the focus. Giga+ [7], Spyglass [8], the adaptive & scalable system [9], Ceph [10], and Ursa Minor [11] are prime examples of this sort of work. In contrast, this project is focused on online use for intermediate, temporary data. We also have a strong requirement for having controls over the visibility and accessibility of the data from various local and remote locations in an effort to fully support the needs of IAWs.

Recent efforts to popularize the Paxos [12] algorithm with Zookeeper [13] have superficially addressed the needs of this project. While it does offer distributed metadata management, it is still single client focused.

The closest work is really as much motivation as it is related work. In this case, PreDatA [1] and Glean [3] are examining how and were to place computation on the path from the simulation to storage. While both of these efforts are steps towards IAWs, both lack the semantics of transactions to manage the data sets as they are being processed and the metadata management for finding data that is in some state and location along the path. DataSpaces [2] is quite similar, but it has focused on offering a querying capability from the published data sets. It lacks the ability to control visibility and enforcing atomic actions that would make an IAW a reliable and predictable tool for scientists.

The biggest effort to address the needs to this system is likely SciDB [14]. The major difference is in philosophy based on the motivating applications. For SciDB, the astronomy-related applications have large data sets that seem to be independent pieces all stored together based on some temporal or spatial relation. This eliminates the need to control a collection of insertions from a set of clients as a single, atomic action. Instead, the simulation driven example has massive data sets generated on a periodic basis. Each of these data sets is independent from each other, but are distributed over a large number of clients requiring a different interaction model for storing data. In this case, the whole set from all clients should be treated as a single entry with the transaction managing the insertion of the set of pieces as an atomic action and controlling visibility until the transaction has been committed.

Overall, there does not exist a current effort that delivers the kind of parallel client with support with additional functionality, like transactions, metadata service. There are many pieces that contribute ideas, such as the examples described above, but the new effort will have to address this different scenario in order to bolster the reliability of IAWs.

3. TRANSACTIONAL PARALLEL METADATA REQUIREMENTS

IAWs have a relatively simple set of requirements that can be summarized as

> Making *complete* data available *quickly* to the next stage of processing across the *distributed* environment hosting different processing components.

Implementing this in a parallel and distributed system is unfortunately far more complex. There are several complications that must be addressed to enable these seemingly simple requirement.

First, a data set must be *complete*. For a parallel and potentially distributed system, *complete* requires that the metadata system have either direct or indirect coordination with all clients to agree that all data is 'posted' and available somewhere. Scientific workflow systems using a centralized disk storage system for intermediate data perennially wrestle with how to determine if a dataset is complete and ready for processing. Tricks like creating a dummy 'finished' file or waiting for the file size to stabilize for a period of time are commonly used to try to prevent premature data set processing. If the metadata service provides a way to prevent discovery of an 'in process' data set until it is complete, IAWs become simpler than their offline counterparts. Through the use of a transactional mechanism supporting ACID semantics, a data set is only made visible in the metadata system once the transaction that creates the data set is committed.

Second, making data visible *quickly* is a two-part problem. First, there must be a viable way to determine when the data set is ready for consumption. This has been addressed above. The second part of the problem is doing this with minimal delay. With transactions preventing the appearance of a dataset in the metadata service until it has been successfully completed, a client can rely on the data set being available as soon as it appears in the catalog. Limited compute area storage demands that the temporary intermediate data be processed and discarded as quickly as possible to prevent delays in the simulation. Additional functionality such as triggers, callbacks, and explicit notification messaging will minimize the discovery delay and likely to be incorporated in some form as the service evolves.

The third requirement addresses the *distributed* nature of data processing for scientific workflows. Data must be accessible across not just the single HPC resource, but also across a WAN environment for cases where additional compute platforms are used to distribute the compute load to potentially more appropriate resources as described below. While this attribute is not required for a proof-of-concept implementation, it is key for any production scale deployment. For scenarios where one HPC compute resource is architected for simulation tasks by making a certain ratio of compute cores to memory, visualization resources tend to have a much larger amount of memory per core to afford more efficient rendering. These differing hardware profiles are nearly universally deployed as separate resources. In many cases, these are deployed in the same data center. In others, they are hosted at another location somewhere across the Internet. For these latter cases, with sufficient gateways and bandwidth, the IAWs should be able to find and request data from locations across the WAN to continue processing using the most appropriate hardware.

A final, implied requirement is that if there are multiple metadata service instances responsible for portions of the overall workflow, there must be some way to directly or indirectly query other sources to find the requested data. This feature is advanced and not likely to be in the initial implementation. Instead, the clients will have to have direct knowledge of where other metadata services are and how to contact them manually.

Overall, the parallel and distributed nature of data creators and consumers requires a metadata service that can control visibility of data sets, be integrated with the consensus of creators before revealing a data set, and available across the WAN no matter where the data is stored.

Ancillary to these requirements is the ability to rethink the interaction mode. The file system interface, while comfortable, is not necessarily the best option. Instead, an object based structure organized by run, variable, and iteration is much more natural for the processing. This structure hides any details of how and where the data is stored simplifying discovery and access. Without an underlying file system, introducing this sort of interface is as easy as any other. It also makes it more natural to introduce a query-style interface for retrieving the data. The whole file metaphor disappears in favor of thinking about the data directly. The eSiMon [15] system from ORNL offers a view of how this might work from a user's perspective.

4. APPLYING TO AN HPC WORKFLOW

At a detailed level, the IAWs targeted by this work contain a core simulation that generates the raw data that is then analyzed by various routines before generating some output that is written to storage.

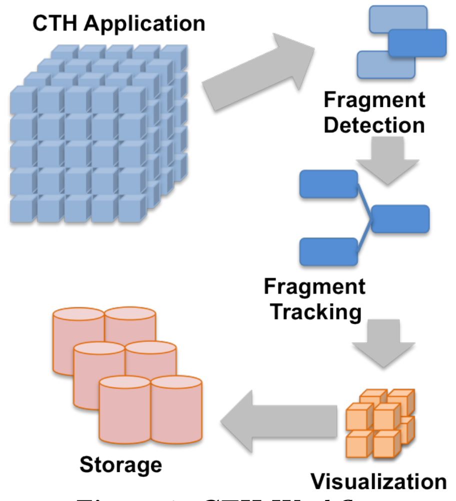

Figure 1: CTH Workflow

At a concrete level, the CTH high energy physics simulation in use at Sandia models explosions of various sorts. The simulation itself generates raw data representing the state of the materials and various attributes, such as the temperature and velocity, as the simulation progresses. This raw data is traditionally written to a centralized disk storage system for later processing. The next step of the processing transforms this raw data into a list of fragments useful both for tracking and rendering using a visualization system like ParaView. This processed data is sometimes written back to disk prior to the next phase that generates a visualization of the current data set.

Key to understand with this system is that the data moved in the example is not stored in the metadata system. Instead, it is stored using an object store on some location with the metadata about each object that comprise the entire variable stored in the metadata service described in this paper.

By moving the various analysis and visualization components online, the rate of data output can be increased exceeding the IO bandwidth of the storage system. This enables more detailed visualization movies and affords tracking fragments automatically for more detailed knowledge of how the material is performing in the simulation. In order for these steps to take place, each output action must be complete before the next one can begin. Using a data staging technique to store the data in the compute area requires a mechanism like the doubly distributed transactions to control the visibility of the data as it moves through the workflow. The metadata service is critical for tracking the data as it is generated so that each additional component along the IAW path can discover what data is available for processing and when. The actual raw data is not generally useful once these derived elements are generated allowing it to be safely discarded. These derived elements are considerably smaller in total than the raw data required to create them saving time spent writing to the storage array resulting in more detailed science output from simulation runs.

5. CURRENT IMPLEMENTATION STATUS

The current system is well underway. The transaction protocol is in its second iteration with a model that is expected to scale easily towards exascale. It is in use for both system reconfiguration as well as data movement tasks. The metadata service has been designed taking into account the lessons from the development of the transaction protocol itself as well as data staging efforts. The implementation of the metadata system itself has expected completion of a rudimentary system meeting the general requirements described above is scheduled to be complete in the next 3 months based on other project milestones.

6. CONCLUSIONS AND FUTURE WORK

IAWs are a likely requirement for scientific simulations in the move towards exascale. The differing requirements of parallel clients and handling groups of operations across applications similar to how traditional database transactions operate makes existing metadata services inadequate to support data discovery and tracking operations.

The current implementation effort is focused on developing for a single HPC platform. Once this scenario is fully tested, the next step will expand the system to work across a wide area network so that a scientific workflow can be distributed across not just multiple platforms within a single data center, but across the Internet. With this iteration, the platform will offer a foundation for a potential exascale file systems. With the requirements outlined in this paper, the additional work is well understood.

7. ACKNOWLEDGEMENTS

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

8. REFERENCES

[1] F. Zheng, H. Abbasi, C. Docan, J. Lofstead, S. Klasky, Q. Liu, M. Parashar, N. Podhorszki, K. Schwan, and M. Wolf, "PreDatA - preparatory data analytics on Peta-Scale machines," in *In Proceedings of 24th IEEE International Parallel and Distributed Processing Symposium, April, Atlanta, Georgia*, 2010.

[2] C. Docan, M. Parashar, and S. Klasky, "DataSpaces: An interaction and coordination framework for coupled simulation workflows," *HPDC '10: Proceedings of the 18th international symposium on High performance distributed computing*, 2010.

[3] V. Vishwanath, M. Hereld, and M. Papka, "Toward simulation-time data analysis and i/o acceleration on leadership-class systems," in *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, oct. 2011, pp. 9 –14.

[4] J. Lofstead, J. Dayal, K. Schwan, and R. Oldfield, "D2t: Doubly distributed transactions for high performance and distributed computing," in *IEEE Cluster Conference*, Beijing, China, September 2012.

[5] B. Braban and P. Schlenk, "A well structured parallel file system for PM," *ACM Operating Systems Review*, vol. 23, no. 2, pp. 25–38, Apr. 1989.

[6] M. R. Brown, K. N. Kolling, and E. A. Taft, "The alpine file system," *ACM Trans. Comput. Syst.*, vol. 3, no. 4, pp. 261–293, Nov. 1985.

[7] S. Patil, G. A. Gibson, S. Lang, and M. Polte, "Giga+: scalable directories for shared file systems," in *PDSW*, G. A. Gibson, Ed. ACM Press, 2007, pp. 26–29.

[8] A. W. Leung, M. Shao, T. Bisson, S. Pasupathy, and E. L. Miller, "Spyglass: Fast, scalable metadata search for large-scale storage systems," in *FAST*, M. I. Seltzer and R. Wheeler, Eds. USENIX, 2009, pp. 153–166.

[9] J. Xing, J. Xiong, N. Sun, and J. Ma, "Adaptive and scalable metadata management to support a trillion files," in *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, ser. SC '09. New York, NY, USA: ACM, 2009, pp. 26:1–26:11.

[10] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn, "Ceph: A scalable, high-performance distributed file system," in *Proceedings of the 2006Symposium on Operating Systems Design and Implementation.* University of California, Santa Cruz, 2006, pp. 307–320.

[11] M. Abd-El-Malek, W. V. C. II, C. Cranor, G. R. Ganger, J. Hendricks, A. J. Klosterman, M. P. Mesnier, M. Prasad, B. Salmon, R. R. Sambasivan, S. Sinnamohideen, J. D. Strunk, E. Thereska, M. Wachs, and J. J. Wylie, "Ursa minor: Versatile cluster-based storage," in *FAST.* USENIX, 2005.

[12] L. Lamport and K. Marzullo, "The part-time parliament," *ACM Transactions on Computer Systems*, vol. 16, pp. 133–169, 1998.

[13] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed, "Zookeeper: Wait-free coordination for internet-scale systems," in *In USENIX Annual Technical Conference*, 2010.

[14] P. Cudre-Mauroux, H. Kimura, K.-T. Lim, J. Rogers, R. Simakov, E. Soroush, P. Velikhov, D. L. Wang, M. Balazinska, J. Becla, D. DeWitt, B. Heath, D. Maier, S. Madden, J. Patel, M. Stonebraker, and S. Zdonik, "A demonstration of scidb: a science-oriented dbms," *Proc. VLDB Endow.*, vol. 2, no. 2, pp. 1534–1537, Aug. 2009.

[15] R. Barreto, S. Klasky, N. Podhorszki, P. Mouallem, and M. A. Vouk, "Collaboration portal for petascale simulations," in *CTS*, W. K. McQuay and W. W. Smari, Eds. IEEE, 2009, pp. 384–393.