

Reference ID: 11227041897_Luszczek

Reference ID: 11227041897_Luszczek

Submission Date and Time: 12/16/2019 4:42:23 PM

This contribution was submitted to the National Science Foundation in response to a Request for Information, <https://www.nsf.gov/pubs/2020/nsf20015/nsf20015.jsp>. Consideration of this contribution in NSF's planning process and any NSF-provided public accessibility of this document does not constitute approval of the content by NSF or the US Government. The opinions and views expressed herein are those of the author(s) and do not necessarily reflect those of the NSF or the US Government. The content of this submission is protected by the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (<https://creativecommons.org/licenses/by-nc-nd/4.0/legalcode>).

Consent Statement: "I hereby agree to give the National Science Foundation (NSF) the right to use this information for the purposes stated above and to display it on a publicly available website, consistent with the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License (<https://creativecommons.org/licenses/by-nc-nd/4.0/legalcode>)."

Consent answer: I consent to NSF's use and display of the submitted information.

Author Names & Affiliations

Submitting author: Piotr Luszczek - University of Tennessee

Additional authors: Terry Moore, University of Tennessee

Contact Email Address (for NSF use only): (hidden)

Research domain(s), discipline(s)/sub-discipline(s)

Computer Science, System Architecture, Storage and Networking

Title of Response

Unpacked Data Management Systems

Abstract

Today the amount of research-relevant data being generated is going up exponentially, and the number of devices outside clouds and data centers that are generating it are also growing at similarly explosive rates. Under such conditions, the importance of having a shared and globally scalable data management platform through the use of which domain-specific scientific communities can collaborate cannot be

underestimated. Recent research on the "Deployment Scalability Tradeoff" argues that achieving such a performant and globally scalable service platform for distributed data management requires a common software infrastructure that is weak, simple, generic and resource limited. The underlying architecture of the Data Logistics Toolkit (DLT) stack, provides such an approach to exposing and safely sharing low level storage resources to local and wide area ICT environments. Building generic service infrastructure to implement domain defined data structures that do not fit the assumptions of the siloed stacks of current files systems and databases requires a primitive storage service that is accessible outside of the system to which the storage device is directly connected. The DLT architecture is based on such a fundamental service, which also serves as a spanning layer providing interoperability among all DLT-based systems.

Question 1 (maximum 400 words) – Data-Intensive Research Question(s) and Challenge(s). Describe current or emerging data-intensive/data-driven S&E research challenge(s), providing context in terms of recent research activities and standing questions in the field. NSF is particularly interested in cross-disciplinary challenges that will drive requirements for cross-disciplinary and disciplinary-agnostic data-related CI.

The fundamental research question is "How can we build usable, performant and scalable services for distributed data management on a common software infrastructure that is weak, simple, generic and resource limited?" Current commodity solutions (mainly file and database systems) are based on the data center model, which creates an ICT environment that is constrained geographically, administratively and in terms of network topology. Such systems implement an encapsulated system architecture that hides both the fundamental storage resources and their defining data structures and metadata behind the higher level abstractions of non-interoperable software stacks. Consequently, in order to take advantage of the development effort that has been invested in these systems, developers must implement data structures as an overlay on top of these high level services and their application-layer APIs. By contrast, Logistical Networking, as implemented in the Data Logistics Toolkit (DLT) stack, provides an approach to exposing and safely sharing low level storage resources to local and wide area ICT environments. The DLT is built on a system architecture that has been guided by the Deployment Scalability Tradeoff, which connects widespread adoption and future-proofing to a weak, simple, generic design. DLT components have been used to build two mature higher level services: 1) the Lstore enterprise storage system currently in use by Vanderbilt University computing center to maintain and provide access to datasets at petascale; and 2) the Intelligent Data Management System (IDMS), a Content Delivery infrastructure developed by researchers at Indiana University. Both LStore and IDMS build on the Internet Backplane Protocol, which is the lowest, "buffer-as-a-service" layer of the DLT and conforms to the Deployment Scalability Tradeoff principle. The higher layers of these services are more specialized and built to implement more conventional high level interfaces. This strategy conserves scarce software development resources and provides an expedient path to performance and user acceptance. However it also tends to make the advantages of interoperability and generality harder to demonstrate and exploit. The question that needs to be tested deployment and use is the following: Can a layered service stack constructed on this model enable the convenient implementation of new

data structures that, when embedded in local and wide area networks, provide the foundation for common utilities to support common services that can achieve high performance? An example of such a data structure would be a common service for storing, providing flexible access through domain-specific metadata, indices, links and algorithms.

Question 2 (maximum 600 words) – Data-Oriented CI Needed to Address the Research Question(s) and Challenge(s). Considering the end-to-end scientific data-to-discovery (workflow) challenges, describe any limitations or absence of existing data-related CI capabilities and services, and/or specific technical and capacity advancements needed in data-related and other CI (e.g., advanced computing, data services, software infrastructure, applications, networking, cybersecurity) that must be addressed to accomplish the research question(s) and challenge(s) identified in Question 1. If possible, please also consider the required end-to-end structural, functional and performance characteristics for such CI services and capabilities. For instance, how can they respond to high levels of data heterogeneity, data integration and interoperability? To what degree can/should they be cross-disciplinary and domain-agnostic? What is required to promote ease of data discovery, publishing and access and delivery?

Currently there is no public, shared and globally scalable data management platform through the use of which domain-specific scientific communities can collaborate. Commercial cloud services do not present a counter-example to this claim. Commercial clouds are inadequate as common infrastructure for collaborating scientific communities in a number of important regards: lack of interoperability, lack of control over policies and costs, security/privacy concerns, inadequate support for HPC, inability for communities to innovate in fundamental services, data export charges, lack of universal coverage, and so on. In general, building the required community data management platform on a siloed storage stack, including those offered by cloud providers, is problematic. The value of using a siloed stack is that by accessing its high level services the client gets access to constituent lower level services that have been developed, debugged, and optimized for them. This is a benefit as long as the client's requirements map cleanly and efficiently onto the assumptions and imperatives of the silo's high level abstractions. If they do not, then clients must either adapt their requirements or make do with an implementation that may make poor use of lower level system resources. For example, if one tries to use a POSIX-compliant file system to implement a domain-specific parallel storage system for large scale or complex linked data structures, one confronts a relatively stark choice: either create a proliferation of small linked files, which is cumbersome, inefficient and prone to corruption of its structure; or, pack up such objects into large files representing collections which may not be well aligned with the domain-specific dependences between constituent objects. Building generic service infrastructure to implement a variety of data structures that are not "packed" in this way requires a primitive storage service that is accessible outside of the system to which the storage device is directly connected. The DLT architecture is based on such a fundamental service, which also provides the basis for interoperability among all DLT-based systems. Some of the most important services used on top of primitive linkable storage objects are data structure integrity and maintenance functions typically provided by file systems or supervisory utilities. In the Unix and Windows file system, these functions included "file system checking" and "compaction." In order to provide similar services for a more general class of data structures, it must be possible to

characterize the integrity conditions that must be verified and actions to be taken upon verification or specific conditions. This requires a language that can express such integrity conditions, either declaratively or by specifying a checking algorithm. Packing everything inside a functionally specific and topologically restricted silo obviates this difficult research problem. Assuming the existence of a service capable of checking and enforcing integrity conditions, the benefits of creating unpacked data structures lies in the potential for implementing of application defined algorithms on them that do not match the structure of commodity file and database systems. Thus the ultimate benefit in the creation of such structures based on a generic storage and networking stack is the development of domain-specific services. Part of that development is the definition of data formats and representations that express the semantic structures that can them be reflected structurally. Of course, such new data structures will likely have initial issues achieving the high levels of performance application developers expect, since decades of optimization and software/hardware codevelopment have been put into their familiar commodity servers and software stacks. In some cases similar codevelopment may be necessary to fully achieve the potential of the generalized architecture. None of this can occur without gaining experience through practical deployment at scale.

Question 3 (maximum 300 words) – Other considerations. Please discuss any other relevant aspects, such as organization, processes, learning and workforce development, access and sustainability, that need to be addressed; or any other issues more generally that NSF should consider.

Modern networked data services take full advantage of the strong assumptions that can be made about the implementation of ICT silos in the industrial world. It is common for services to rely on continual low-latency datagram delivery, always-connected servers, stable and uninterrupted datagram routing paths and high bandwidth connectivity to take just a few examples. Services implemented at Cloud Computing centers are among those that place the greatest demands on the Internet backbone and "last mile" connectivity to edge networks. The generic, simple, and resource limited storage service underlying the DLT allows developers to make much weaker assumptions. By using it, they can decompose many services into synchronous and asynchronous components, so that different "Data Logistics" strategies can be applied to each part. Techniques used in Content Delivery Networks, including caching and prestaging, can be applied on a fine-grained and even per-client basis. In some cases, a careful analysis of the application combined with reconsideration of the truly necessary characteristics of the service delivered to the end-user can reduce the need for high quality synchronous connectivity and conventional enterprise storage services to the vanishing point. In a sense, strong network assumptions are a crutch that allows wasteful application design and ease of development. Today, some environments cannot support strong network and storage assumptions, even when local IT resources are available. Examples are communities isolated through geography, economic (poverty, discrimination) or political circumstances (famine, war), or social factors. Disasters create environments where infrastructure is disrupted even in the most advanced societies (e.g., Hurricane Katrina). The DLT offers the alternative of interoperably mixing heterogeneous synchronous and asynchronous data transport to create a flexible platform that supports a variety of distributed applications and that, by contrast, can be cheap, robust and easily deployed.

Response to NSF 20-015, *Dear Colleague Letter: Request for Information on Data-Focused Cyberinfrastructure Needed to Support Future Data-Intensive Science and Engineering Research*

Reference ID: 11227041897_Luszczek

-- End Submission --