Using Data Partitions and Stateless Servers to Scale Up Fedora Repositories

Gregory Jansen
College of Information Studies
University of Maryland
College Park, USA
jansen@umd.edu

Aaron Coburn
Inrupt.com
(formerly Amherst College)
Worldwide

Adam Soroka
the Smithsonian Institution's
Office of Research Computing
Washington DC, USA

Richard Marciano
College of Information Studies
University of Maryland
College Park, USA
marciano@umd.edu

***Abstract*—We describe the development and testing of the next-generation Trellis Linked Data Platform with Memento versioning support. In addition to highlighting several features that set this system apart from others, we elaborate on the extensive testing and compatibility work that was done in order to align this system with the Fedora 5.0 specification. We draw attention to the performance and scaling features provided by the Trellis Linked Data Platform in general and by the Cassandra database back end. We review the profound impact that such a system can have on demanding, next generation use cases, such as crowd sourcing, machine learning, and direct file access by desktop applications.**

Keywords—Linked Data Platform, Memento, Fedora Repository, Apache Cassandra, Digital Repositories, Distributed Database

I. THE DRAS-TIC FEDORA PROJECT

The Digital Repository at Scale that Invites Computation (DRAS-TIC) Fedora research project, funded by a two-year National Digital Platform grant[1] (now extended into a third year) from the Institute for Museum and Library Services (IMLS), is producing open-source software, tested cluster configurations, documentation, and best-practice guides that enable institutions to reliably manage Fedora-compatible linked data repositories with petabyte-scale collections. DRAS-TIC is a digital repositories research initiative at the Digital Curation Innovation Center at the University of Maryland (UMD) College of Information Studies. We have been working extensively with our organizational partners on this project, including UMD Libraries, Georgetown University Libraries, the Smithsonian Institution, Amherst College, and the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champain.

II. PERFORMANCE AT SCALE

We began this project to engage with problems of data scaling. In other words, we identified a need for repository systems that can keep growing at a predictable and incremental cost while maintaining optimal performance. The first web-scale companies encountered these challenges fifteen or twenty years ago as their user bases grew, sometimes in unexpected leaps and bounds. Their data is shaped differently than repository data, to be sure, but their chief requirements for storage capacity management, high availability, and performance are now challenging digital repositories. In response, web-scale companies developed distributed storage and database systems. Instead of creating ever larger database servers, they partition or divide up their data, such that it can be managed across a whole fleet of smaller servers[2]. These servers together compose a storage or database cluster that provides redundant copies of data and is therefore resilient in the face of individual server failures. These clusters also offer a performance benefit. Since several servers within a cluster hold copies of a given piece of data, there is no single bottleneck server that limits access to that data.

Distributed technology offers sophisticated storage tuning choices that we in the repository world can use to meet increasing storage demands, especially for larger files alongside greatly expanded metadata, transcripts, and surrogate files. Many institutions find themselves engineering a second or third repository, in order to meet demand for a new collection stream. We think that a single distributed repository platform can grow to meet any organization's demands, while remaining more cost efficient, manageable, and reliable.

III. TRELLIS LINKED DATA PLATFORM

The Trellis Linked Data Platform (LDP)[3] was created by our software development partner, Aaron Coburn, with the support of Amherst College. Trellis LDP has been adopted by the Solid project for social linked data and by Inrupt.com, founded by Sir Tim Berners-Lee. Trellis LDP implements the same family of web standards that are used by the Fedora Repository 4 and 5 software series, namely the Linked Data Platform 1.0 and Memento 1.0. Meanwhile the Fedora Commons Repository, the flagship repository software for the galleries, archives, libraries, and museums (GLAM) community, was paired with a new Fedora 5.0 API specification in order to encourage the development of more

implementations of Fedora Commons to meet diverse community challenges. The DRAS-TIC Fedora project targeted the Fedora 5.0 specification from the outset and we soon partnered with Aaron Coburn and the Trellis LDP project, along with Adam Soroka with the Smithsonian Institution's Office of Technology. This partnership let us focus more time on a distributed storage implementation and extensive performance testing to compare various Fedora implementations.

Trellis is an ideal application to pair with distributed storage, as it is implemented as a stateless server. This means that no data is held in the web application's front-end. Instead data is streamed as quickly as possible to a chosen back-end storage system. The Trellis platform also decided not to support any locking for transactions, so every change to a resource is written as soon as possible. Dropping the transactions feature has a very important consequence, which is that no coordination is required between multiple Trellis front-end servers. An organization can "spin up" as many front-end Trellis servers as are required to meet performance demands. In today's cloud and virtual environments, this elasticity feature is built-in and leveraged by service management tools that may add and remove servers on-demand.

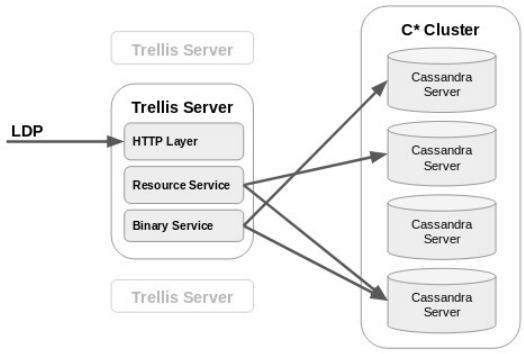

Fig. 1. *Trellis system diagram showing scaling.*

Finally, Trellis is well-designed software that is open to extension in the form of new Java code. One can create different back-end storage systems for RDF or binary resources or create additional front-end behaviors, such as the optional support for the WebDAV remote storage protocol. Where Trellis LDP may differ from the Fedora 5.0 Specification[4] this extensibility gives us the option to add code and alter its behavior to better match the standard. Lastly, Trellis LDP is high-performance Java server software that is written to fully leverage the modern Java virtual machine features of asynchronous operation and data streaming in a customizable MicroProfile container. This is borne out by the metrics we demonstrate in performance testing.

IV. Cassandra Storage Back-End

There are a wide range of distributed storage and database systems, both open source and proprietary. Apache Cassandra was originally developed by Facebook engineers, when it was one of the first companies to encounter the growth challenges of web-scale business. It has become the most popular no-SQL database system in the world and is used by many well-known web companies. Apache Cassandra[5] is unique in that it is mature, open source, and well-documented software that is complimented by commercial services and consulting support. The DataStax company provides a commercial version of the software, along with Cassandra hosting and support services. This company, along with several others, ensures that the organizational IT department can choose a mixture of hosting and support that fits with their own resources and needs.

Cassandra uses a data partitioning technique that stores your data on a subset of servers according to a partition key. The partition key is composed of columns that you choose from your data. These columns are also needed to access that data. This system is best visualized as a ring of data partitions. In Figure 2, each node is storing the data for two partition keys, so every piece of data is replicated to two of the eight storage servers.

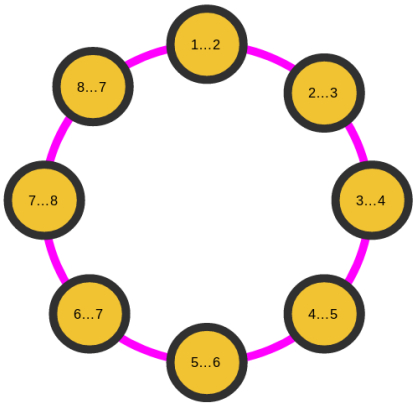

Fig. 2 *Simplified Cassandra Data Partition Key Ring*

The Trellis Cassandra storage module and the related application code were spearheaded by our software development partner, Adam Soroka, at the Smithsonian Institution. He created Cassandra storage services for Trellis RDF resources, binary resources, and mementos. In the Trellis Cassandra back end we use the path part of the URL as our partition key, since every time we want to access a resource we have that URL path. The result is that Trellis LDP data is spread across storage servers relatively evenly and that each of the data is replicated to additional storage servers. The replication factor is a configuration item that can be tuned to one's needs. When the Trellis Cassandra back end receives a request for data, it talks to all Cassandra servers that are supposed to have copies of that data. It can then quickly use the first available response.

Cassandra is the key technology for creating manageable and predicable costs for storage. By adding storage servers to a Cassandra cluster you increase storage capacity. This is a straightforward process that is usually followed by a server command to re-balance existing data into partitions that include the new servers. Cassandra storage servers are cheap, commodity devices with ordinary disks or cloud-based machines, in contrast to high-cost, single-vendor enterprise storage systems. We think that memory institutions need to easily calculate new storage costs when accepting massive new collections. With Cassandra they can add storage servers at an incremental and predicable cost, without having to plan and then migrate to expensive new enterprise storage systems.

V. Performance Testing

The project partners knew from the outset that distributed technologies would offer performance benefits over traditional single-server designs. However, we did not know exactly how much more load such systems could handle. The DRAS-TIC Fedora project proposed to build an extensive LDP testbed system to measure web client performance at a massive scale. This testbed has become the drastic-testbed.umd.edu portal[6]. There you can find results from all performance tests, including the graphs of maximum load and performance over the course of individual test scenarios. Please read our prior article, *DRAS-TIC Linked Data: Evenly Distributing the Past*[7] for a deep exploration of this performance testing and the underlying software engineering concerns.

We found that by adding just three front-end nodes and three back-end storage nodes, we were able to improve performance dramatically. Whereas Fedora with a PostgreSQL back end starts to show overwhelming failure responses at 40 requests per second (Figure 3), our Trellis Cassandra system handles more than 480 requests per second (Figure 4). Note that the red vertical lines below indicate failed requests and that the graphs are at different scales.

Fig. 3. *Fedora PostgreSQL degrades at 40 requests per second.*

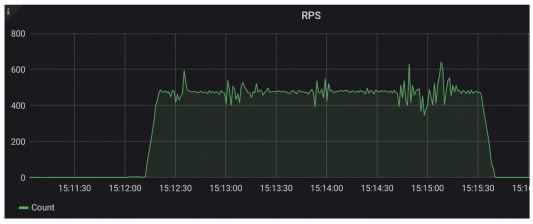

Fig. 4. *Trellis Cassandra 4-4 cluster handles 480 requests per second without failures.*

In fact the Trellis Cassandra system did not begin to show signs of trouble until we stressed it with 560 requests per second. (Figure 5)

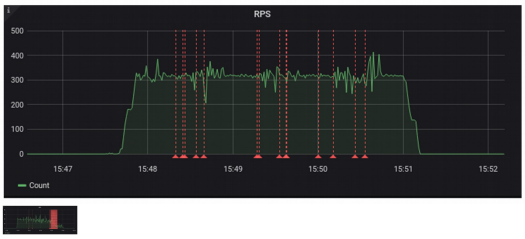

Fig. 5. *Trellis Cassandra 4/4 cluster degrades at 560 requests per second. (Figure 3 repeated for scale)*

We measure a dramatic performance improvement resulting from the distributed software patterns of Trellis Cassandra. The expanded green area in these graphs represent more work the repository can perform over a given time. Whereas repository developers often concern themselves with overloading repository systems with too much ingest load, we see a future where repository write operations are reliable and workload issues are rare and easily addressed when they do occur through additional front or back end servers.

We continue to run performance tests to try out new configurations and tunings of Trellis Cassandra. We also continue to add workload scenarios that are based on use cases from our institutional partners. Inrupt.com recently adopted our performance testing suite and contributed code back to it, in order to measure Trellis performance against social linked data use cases. In our work for the next year we will try more unconventional scenarios. These include the replication of Cassandra data to a second, geographically distant datacenter and the ingest of file collections by drag-and-drop into repository-hosted folders via WebDAV protocol. We also want to test computational scenarios that leverage distributed computation in Cassandra and efficient random-access to bitstreams within files. The ability to process these next-generation workloads can open the door for new approaches to the curation of and access to repository resources.

VI. Fedora 5 Specification

The Fedora 5 specification[8] is supported by a Fedora test suite, which measures whether a candidate system complies with various sections of the specification. Our aim is to have Trellis comply with the Fedora tests and specification as much as is feasible, recognizing that some departures are acceptable and leaving a clear path for those who wish to customize Trellis behavior in this regard. Trellis, through the DRAS-TIC Fedora project, is the second implementation to pursue compliance with the Fedora specification. The Fedora developers have accepted several of our proposed changes to the test suite in order to bring its tests closer to the exact

language of the specification. We are also collecting a library of extensions to Trellis that make it behave more in line with the specification. Finally, we are pursuing a few changes to the Fedora specification, which we hope will simplify the compliance demands for future implementations and clarify repository behavior for developers. At the time of this writing Trellis passes 102 compliance tests. It fails 21 tests in a way that indicates different implementation decisions within the Trellis software. These were either features not adopted in Trellis due to security concerns or different behaviors resulting from interpretations of the Memento, LDP, or WebAC specifications. Some 10 remaining test failures are minor behavior differences that are easy to address through extension code. The process of testing for compliance has been a healthy one for Fedora and Trellis, bringing more focus to the specification and the test code. Another result is that the areas where Trellis differs from the Fedora specification are now documented and readily available through the project portal[9].

VII. Systems Integration & Adoption

One of our main concerns in the remaining year is to ready the various Trellis platform applications for adoption by those in the Fedora community that find them to be a good fit for requirements. One thing required for adoption is clear testing and documentation of systems integration patterns, including all of the messaging and digital curation workflows that surround any repository. For instance, we will need to demonstrate and document that complex workflows are straightforward to implement without relying on locking transactions, if you have fast, reliable write operations. We will need to document the basic operational procedures needed to maintain a Trellis Cassandra cluster. We are working with our partners at the University of Maryland Libraries to test their existing ingest and workflow tools against a candidate Trellis repository. An initial test of their ingest tools, with transaction code removed, was completely successful. We also are committed to pursue their other workflow needs, both for triggering indexing and other curation operations, in the absence of transaction support.

VIII. Conclusion

In our research and development, we have demonstrated the benefits of Trellis Cassandra for performance and storage capacity management. However, we also need to draw out the implications of higher scale repositories for our collections. Higher performance and scale pave the way for massive collections, but they also allow us to explore next generation uses cases. These include web-scale crowd sourcing, machine learning, and high resolution on-demand tile rendering for humanities research, to name just a few examples. In the realm of scientific data and audiovisual materials, where collections are subjected to ongoing computational treatments or reuse, the new possibilities expand to include the application of ordinary desktop tools to repository resources, via WebDAV remote folders, allowing direct, read-only access by all conventional software tools. Many of these are slated for investigation alongside our partners at the Smithsonian Institution, which represents the advanced use cases and digital curation needs of several museums, scientific organizations, and memory institutions. In parallel, the UMD Digital Curation Innovation Center (DCIC) has initiated a new repository research and development project supported by the Capital Region of the National Parks Service. Together with the National Parks Service we will pursue the DRAS-TIC research agenda through development of a prototype repository for the National Archives for Black Women's History[10], a part of the Mary McCloud Bethune National Historic Site. In this work we have the unique opportunity to apply all that we have learned through previous DRAS-TIC research efforts to the requirements of large and demanding collections. As a research and development effort, it affords us the opportunity to reach for next-generation repository capabilities in the ingest, management, preservation, and enhanced access to digital archives. We look forward to sharing the outcomes from these efforts in the coming year.

IX. Acknowledgment

We wish to acknowledge major funding from the NSF "Brown Dog" project (NSF Cooperative Agreement ACI-1261582). We also obtained an IMLS grant [LG-71-17-0159-17], which helped us implement Fedora on Trellis. More details at: http://dcic.umd.edu/about-us/infrastructure/. We also wish to especially acknowledge the deeply cooperative software development partnership that existed between our institutions and was a key element of our success. (Amherst College, Inrupt.com, the Smithsonian Institution, and The University of Maryland)

X. References

[1] Institute of Museum and Library Services. National Digital Platform Research Grant LG-71-17-0159-17. Available online: https://www.imls.gov/grants/awarded/lg-71-17-0159-17 (accessed on 7 October 2019).
[2] Tanenbaum, A.S.; van Steen, M. Distributed Systems: Principles and Paradigms. 2006. Available online: http://barbie.uta.edu/~jli/Resources/MapReduce&Hadoop/Distributed%20Systems%20Principles%20and%20Paradigms.pdf (accessed on 7 October 2019).
[3] The Trellis Linked Data Platform Project Website. Available online: https://www.trellisldp.org/ (accessed on 7 October 2019).
[4] The Fedora 5.0 API Specification. Available online: https://fedora.info/2018/11/22/spec/ (accessed on 7 October 2019).
[5] The Apache Cassandra Software Project Website. Available online: https://cassandra.apache.org/ (accessed on 7 October 2019).

[6] The DRAS-TIC Project Website. Available online: http://drastic-testbed.umd.edu (accessed on 7 October 2019).
[7] Jansen, G.; Coburn, A.; Soroka, A.; Thomas, W.; Marciano, R. DRAS-TIC Linked Data: Evenly Distributing the Past. Publications 2019, 7, 50. Available online: https://www.mdpi.com/2304-6775/7/3/50 (accessed on 7 October 2019)
[8] The Fedora 5.0 API Specification. Available online: https://fedora.info/2018/11/22/spec/ (accessed on 7 October 2019).
[9] Fedora Test Suite results for Trellis PostgreSQL (annotated). Available online: http://drastic-testbed.umd.edu/static/Fedora-reports/trellis-ext-db/report/testsuite-execution-report.html (accessed on 7 October 2019)
[10] Available online: https://www.nps.gov/mamc/learn/historyculture/mamc_nabwh.htm (accessed on 7 October 2019)