Multi-tenancy Cloud Access and Preservation:

Virginia Tech Digital Libraries Platform

James Tuttle, Yinlin Chen, Tingting Jiang, Lee Hunter, Andrea Waldren, Soumik Ghosh, and William A. Ingram
University Libraries, Virginia Tech
Blacksburg, VA, 24061, USA
{james.tuttle, ylchen, virjtt03, whunter, awaldren, soumikgh, waingram}@vt.edu

ABSTRACT

Virginia Tech Libraries has developed a cloud-native, microservervices-based digital libraries platform to consolidate diverse access and preservation infrastructure into a set of flexible, independent microservices in Amazon Web Services. We have been an implementer and contributor to various community digital library and repository projects including DSpace[1], Fedora[2], and Samvera[3]. However, the complexity and cost of maintaining disparate application stacks have reduced our capacity to build new infrastructure.

Virginia Tech has a long history of participation in and contribution to community-driven Open Source projects and has, in that time, developed more than a dozen independent applications architected on these stacks. The cost of independently addressing vulnerabilities, which often requires work to mitigate incompatibilities; reworking each application to comply with developing branding guidelines; and feature development and improvement has burgeoned, threatening to overwhelm our capacity. Like many of our peers[5], our maintenance obligations have made continued growth unsustainable and have pushed older applications to near abandonware.

We have designed and developed the Digital Libraries Platform to address these concerns thus reducing our maintenance obligations and costs associated with feature development across digital libraries. This approach represents a departure from the monolithic architectures of our legacy systems and, as such, shares more infrastructure among individual digital library implementations. The shared infrastructure facilitates rapid inclusion of new and improved features into each digital library instance. New features can be developed independent of any digital library instance and integrated into that instance by inclusion of that feature in the React/Amplify template.

JCDL '20, August 1–5, 2020, Virtual Event, China

ACM ISBN 978-1-4503-7585-6/20/06. https://doi.org/10.1145/3383583.3398624

Changes to the template super class, such as those necessitated by evolving branding guidelines, may be immediately inherited by the template instances that subscribe to it.

The platform implements Terraform[6] deployment templates, Lambda serverless functions, and other cloud assets to form a microservices architecture on which multiple template-based sites are built. Individual sites are configured in AWS DynamoDB, Amazon's NoSQL database service, and via modification of shared template.

Additional services provide digital preservation support including auditing, file fixity validation, replication to external cloud storage providers, file format characterization, and deposit to third-party preservation services.

This presentation also discusses the cost of operating these services in AWS and strategies for mitigating those costs. These strategies include containerization to allow deployment of high-cost, asynchronous services to local infrastructure to take full advantage of existing infrastructure and advantageous utility pricing while allowing for local redeployment. In the past, developers worked in local, independent environments. New features and fixes were submitted to a central development environment testing and validation, which significantly slowed development. Migrating development, review, integration, and deployment processes to AWS decreased the time and resource bottlenecks for those processes.

Our AWS cost accounting demonstrates an 87% savings over our traditional, on-premises Fedora/Samvera approach For a team of four software developers, the total cost using a traditional server-based (a t2-medium EC2 instance) development approach is about $133 per month versus our serverless-based development approach using AWS Amplify at an average of $17 per month.

As the Digital Libraries Platform project expands, we anticipate publishing a set of API documents allowing us and others to reimplement specific microservices independent of the architecture.

Existing Services and Service Compositions

The Digital Libraries Platform consists of independent services[4] which may be called together to support functional areas. Each

service may be called independently or reimplemented, as needed. The microservices include:

Audit Service: Provides a detailed record of events in the lifecycle of digital assets. These events are recorded in PREMIS Data Dictionary for Preservation Metadata[7] format and serialized using JSON in DynamoDB.

Batch Metadata Services: Accepts delimited-field text files containing descriptive and administrative metadata to be associated with digital assets. This service generates JSON entries in DynamoDB.

Derivative Services: Set of services that accept preservation formats and generate access derivatives. For example, Tagged Image File Format (TIFF) images may be submitted to generate JPEG derivatives.

Distributed Preservation Deposit Services: When called against an object, the service manages packaging, deposit, and deposit confirmation receipt of the object and metadata into third-party preservation services such as MetaArchive[8] and Academic Preservation Trust[9].

Format Characterization Service: Provides an API to generate and distill information about the file format and technical characteristics. This service can be used to abstract the underlying tools, such as File Information Toolset (FITS)[10].

Fixity Service: Validates files stored in S3 and generates audit events to demonstrate accuracy of preservation objects.

Identifier Minting Service: Implements Nice Opaque Identifiers[11] (NOIDs) to provide persistent, unique identifiers. These are stored in DynamoDB and are the Foreign Key to support identifier resolution, association with metadata, and other functions.

Image Tile Service: Generates static tiles and manifests complying with the International Image Operability Framework[12] specification to power compatible viewers.

Indexing Service: Using per-site configurations, creates search and browse indexes of metadata. This also uses crosswalks to generate an index for all constituent digital libraries to support a federated search and browse service across all sites in the platform.

Resolution Service: Provides a Universal Resource Identifier resolution service implementing the Archival Resource Key[13] (ARK) specification to support persistent, citable, location-independent Universal Resource Name.

Serialization Service: Abstracts the storage of files and metadata into the storage and database systems, creation of audit events, and confirmation of accurate data via the Fixity Service.

This submission aims to present our serverless architecture design and implementations, elaborate the technical solution on integrating multiple AWS services with other techniques, and describe our streamlined and scalable approach to build a multi-tenancy digital libraries solution in the cloud. Our platform eliminates the need to manage underlying servers and delegate all the heavy lifting to AWS. This approach has allowed us to focus on implementing our business logic into microservices in AWS. We want to share our experience and humbly hope this work can open a new direction for institutions, libraries, and scholars on building the next generation of digital library platform.

CCS CONCEPTS

• Information systems → Digital libraries and archives

KEYWORDS

digital preservation; digital libraries; microservice; cloud computing

REFERENCES

[1] DSpace. 2020. DSpace - A Turnkey Institutional Repository. Retrieved April 6, 2020 from https://duraspace.org/dspace/

[2] Fedora. 2020. Retrieved April 6, 2020 from Fedora - The Flexible, Modular Open-Source Repository. Retrieved from https://duraspace.org/dspace/

[3] Samvera. 2020. Samvera - a welcoming, vibrant community. Retrieved April 6, 2020 from https://samvera.org/

[4] Virginia Tech Digital Libraries Github. 2020. Retrieved June 15, 2020 from https://github.com/vt-digital-libraries-platform

[5] Michael Giarlo and Justin Coyne. 2019. Sustaining a Large-Scale Repository Architecture: Behind the Scenes of the Stanford Digital Repository. Retrieved April 6, 2020 from https://www.conftool.net/or2019/index.php?page=browseSessions&form_session=362

[6] Terraform. 2020. Terraform by HashiCorp. Retrieved April 6, 2020 from https://www.terraform.io/

[7] PREMIS. 2018. PREMIS Data Dictionary for Preservation Metadata. Retrieved April 6, 2020 from https://www.loc.gov/standards/premis/

[8] MetaArchive. 2020. MetaArchive: A Digital Preservation Network. Retrieved April 6, 2020 from https://metaarchive.org/

[9] Academic Preservation Trust. 2020. Retrieved April 6, 2020 from http://aptrust.org/

[10] File Information Toolset (FITS). 2020. Retrieved April 6, 2020 from https://projects.iq.harvard.edu/fits/

[11] Nice Opaque Identifiers. 2006. Nice Opaque Identifiers. Retrieved April 6, 2020 from https://metacpan.org/pod/distribution/Noid/noid

[12] International Image Interoperability Framework. 2020. Home IIIF. Retrieved April 6, 2020 from https://iiif.io/

[13] John Kunze. 2013. The ARK Identifier Scheme. Retrieved April 6, 2020 from https://tools.ietf.org/html/draft-kunze-ark-18