Architecture Review for Advancing Hyku project.

Version 1.3

16 June 2020

Author	Role/Org	Version	Date	Remarks
Rathin Sundar	Solutions Architect, British Library.	V1.0	27-March-2020	Initial version
Ellen Ramsey	Acting Director of Scholarly Communications, University of Virginia Library.	V1.1	22-May-2020	Comments and suggestions from AH PI
Rachael Kotarski	Head of Research Infrastructure Services, British Library.	V1.2	11-June-2020	Resolved comments and suggestions from partners
Ellen Ramsey	Acting Director of Scholarly Communications, University of Virginia Library.	V1.3	16-June-2020	Final version

Contents

1 Introduction....4
2 Background....4
3 Implementation Guidelines....5
3.1 Merging....6
3.2 Multi-tenancy....6
3.4 Interoperability....7
3.5 Portability....7
3.6 Scalability....7
3.7 Stability....7
3.8 Security & Compliance....8
4 Conclusions....8
5 Further Reading....8

1 Introduction

The purpose of this document is to review the architecture and design of the Hyku implementation of British library, with reference to the capabilities of the platform and requirements of the Library. The current implementation of Hyku for the British Library's (BL) shared research repository project would form the baseline for the Samvera Hyku application towards the target state architecture for the Advancing Hyku project. The strategic goals to be achieved by this project are structural improvements for the Hyku framework and new features for the repository application. This document also outlines the scope of the requirements and the main principles and guidelines that drive the Advancing Hyku architecture and design. The main aspects considered for review of the product are the following:

- **Merging** – features from Hyku community into current BL's version and vice versa.
- **Multi-Tenancy** – provider-consumer model.
- **Interoperability** – with other systems e.g. DataCite, Crossref, OAI-PMH API communication.
- **Portability** – components should be deployed for Cloud as well as on-premise infrastructure
- **Stability** – High availability
- **Scalability** – performance should not degrade with increasing volume of data
- **Security** – Open access for users of data and role-based secure model for tenant admins

Prerequisites: It's assumed the reader of this document has good knowledge of the Samvera Hyku repository product and the related functionalities and technologies involved. Please refer to the links in the *Further Reading* section for more information on the product or technologies. This document should be read in perspective with the Grant proposal document[1] of Arcadia project, for a focus on the requirements of the British Library as a member of the Samvera Community.

2 Background

This section gives a background of the system architecture of the BL's Shared Research Repository[2] application that is based on the Samvera Hyku open source repository framework. The system components are designed and implemented in a multi-cloud infrastructure. The components stack involved in the Hyku application are: Sidekiq for handling asynchronous tasks, Redis used as task broker, ZooKeeper for distributed configuration, Memcache and Redis caching, and Solr search engine. The core Hyku application is deployed as Docker container managed by Kubernetes cluster on the Google Cloud platform. The main Hyku components are built from publicly available GitHub code repositories[3]. All data files (including metadata) from the applications are stored in Google Cloud NFS, ZFS file server with persistent storage. The operational database (PostgreSQL) is provisioned with Zonal redundancy backup that serves as a fall-back database for live service. The documents (repository contents) are actually stored securely and independently on AWS cloud using S3 storage.

[1] Available upon request from https://advancinghyku.io/contact/
[2] Currently available from https://iro.bl.uk/
[3] https://github.com/samvera/hyku

The public facing repository websites (client sites) are developed using React JS with Rest API and JSON data model. These components are hosted on a separate scalable infrastructure in the Google Cloud platform. The search indexes are stored using Solr in a dedicated instance as well as on a shared instance that also run on a Kubernetes cluster. This provides a fault tolerant and auto scalable capabilities for a faster search functionality. The application also uses proprietary bespoke components and tools for interfacing with external services. The system architecture aims to move away from the legacy monolithic architecture of the Hyku framework to a decoupled, modular components-based architecture, using a restful API approach.

Figure 1 provides a solution concept diagram to demonstrate the various components of Hyku repository implementation and its interactions.

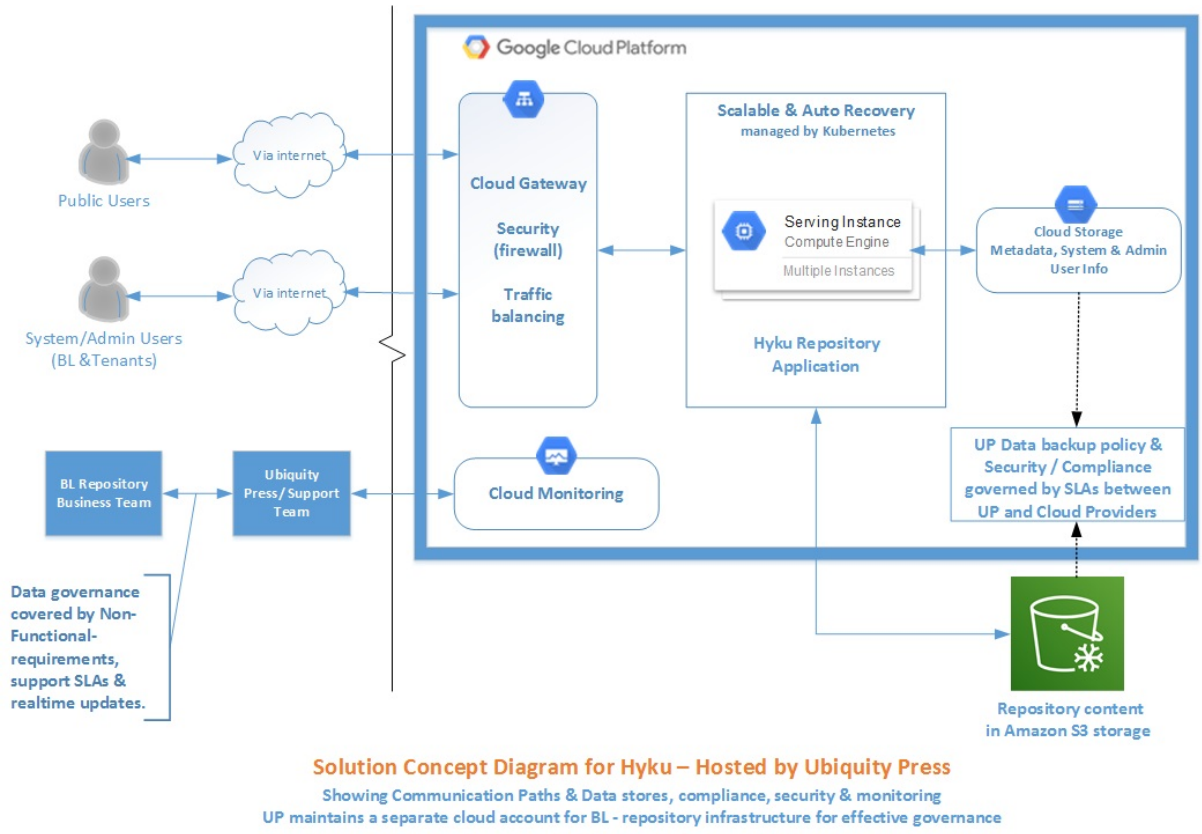

Figure *1. Conceptual view of the repository solution implemented for British Library.*

3 Implementation Guidelines

This section gives more context and describes the architectural principles and implementation techniques which could be useful for implementation in Advancing Hyku, as well as for the wider Samvera Hyku community. It's advisable to use an iterative approach for development and deployment of new functionalities by using multiple work packages and transition-state-architectures

with milestones, as schematically shown in Figure 2. The aim of this approach is that at any stage of the delivery, the code base should be useable for the community and should have upgrade-compatibility with future versions of releases during the project duration.

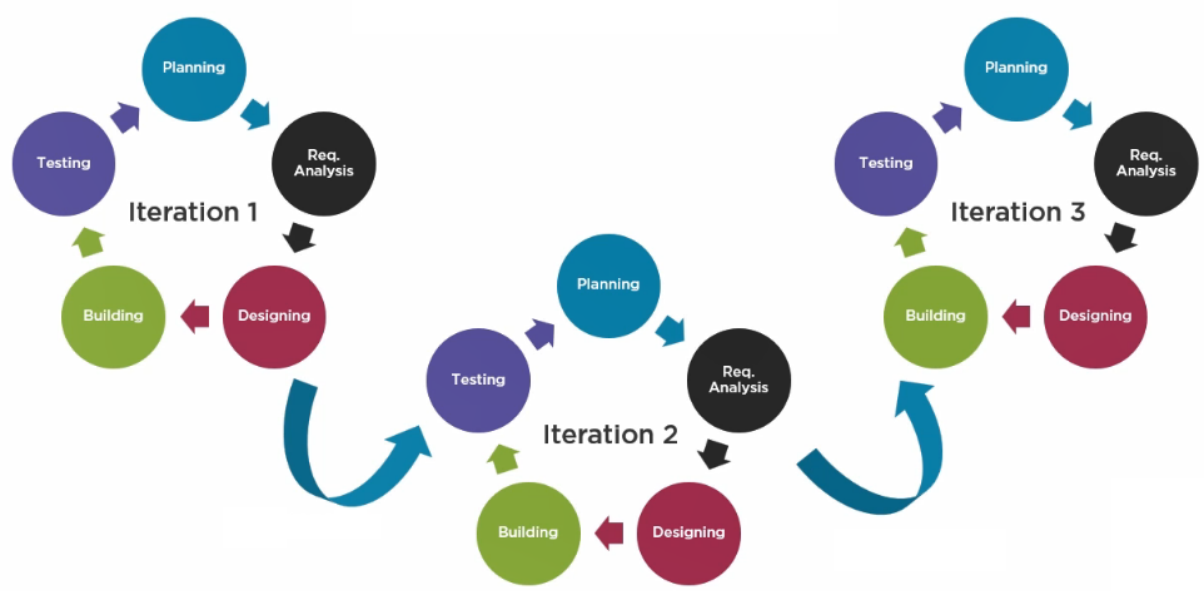

Figure 2. The iterative approach to development and deployment.

3.1 Merging

Ubiquity Press have progressed a lot of good work for the British Library's Shared Research Repository, delivering new functionalities into the Hyku product. But these changes are currently held in a specific code repository, though publicly available but still outside of the main git branch of Hyku. These developments have been implemented in Hyku Version 1, but the Hyku community has advanced to Version 2, with lots of new features. There will be significant gains for the Hyku community if the code repositories are merged and provided as the primary code base for new users. This merging process needs to be worked out in close collaboration with the Samvera community. It would also be useful to publish the documentation about the configurations (Docker and Kubernetes) and deployment/build scripts for the British Library's version of Hyku to the community, as part of this merge process. This would enable any new organisation to understand and on-board a new repository application service quickly and at a lesser cost.

3.2 Multi-tenancy

Multi-tenant architecture in this context means multiple instances of the application operating in a shared environment, thereby enabling a provider and customer model of repository service. For example, a research institution can host and manage the service in one physical or hosted environment and can serve multiple customers for storing their data and indexing it for searches (discovery and access). This would also allow the data to be shared across the tenants taking part in the same service. The advantages of these architectural features are:

- Customised client website design matching their brand, theme and logos
- Common data standard and business features for all the tenants
- Individualised data storage management for the tenants.

Having a core common data model for all of the tenants of the shared repository system would be easier to manage. But the design should consider having an extensible metadata standard, between the front-end components (e.g. search websites) and the back-end Hyku components. This would allow the tenants to customise the worktype and other templates handled in the repository solution. The British Library's version of Hyku has partly addressed this, by using separate client sites, JSON data model and an API layer. The metadata and the actual contents of the artefacts (documents) stored in the repository system should be independent of the Hyku application thereby facilitating easier extraction of data.

3.4 Interoperability

Interoperability features provide easier import and export functionalities of the data for the organisations using the repository service. Both the metadata and the actual content (documents) need to be persisted independent of the Hyku application semantics, i.e., the metadata and content should persist in the semantics even if they are migrated outside the Hyku application to another application environment. The design of the application should allow communication with other services such as ORCID, Crossref, DataCite, and proprietary applications that use OAI-PMH API standards for data communication.

3.5 Portability

Portability of Hyku application components for either on-premise or cloud installation is a key feature for increasing the user base and on-boarding new organisations to realise the work of this project. The Hyku application components should be packaged for both on-premise infrastructure and cloud infrastructure installations to support a range of installation requirements across the community.

3.6 Scalability

This feature refers to the system's ability to enlarge or diminish the capacity based on the number of users or requests experienced by the repository service. In the cloud hosted model of Hyku application, technologies like containerisation and Kubernetes will provide automatic scaling capabilities. The design of the Hyku components needs to provide both horizontal and vertical scaling based on the organisation's infrastructure needs. The main advantage of this feature is that the cost of providing the repository service will be optimised, as the cost follows the demand on the system.

Performance was a key factor that gets degraded in the earlier versions of Hyku product particularly when the volume of search data increases. Hyku's dependencies on legacy Fedora components need to be analysed. It should implement the latest stable versions of Fedora or alternatively consider using Valkyrie for data persistence and linking the search indexers. The British Library's implementation of Hyku has leveraged the Solr search indexes by provisioning them on a fault tolerant (Kubernetes) mechanism. Alternatively, using a Valkyrie component would increase the ability of extracting metadata to various relational databases for example SQL, MySQL, etc. at a later date.

3.7 Stability

The application design of the Hyku components should provide for high availability using fault tolerant mechanisms. This can be provisioned and configurable for the cloud-based installations more easily when compared with on-premise installations. All the legacy open source components need to be upgraded to the latest supported version of that component, for example the Solr indexing components in Fedora should continue to use the latest version available. Good documentation also

needs to be provided for users as how to recover systems in case of a disaster event. Documentation about setting up failsafe mechanisms for the repository service should be provided for users.

3.8 Security & Compliance

The websites provided along with the Hyku repository application suites need to be secured for the top 10 web application security risks. For example, injection, broken authentication, sensitive data exposure, broken access control, cross-site scripting, using components with known vulnerabilities, insufficient logging & monitoring. Sensitive information for data-at-rest needs to be encrypted and protected. Data-in-transit needs to be secured with adequate TLS security protocol. Best practices and installation requirements for security standards need to be published, for new users to understand the security implications of the Hyku repository application. The data components design needs to be compliant with privacy regulations e.g. GDPR, FIPPs.

4 Conclusions

The architecture and guidelines described here outline the British Library's assessment – based on its own requirements – of how best to guide the developing architecture of Samvera Hyku in a way that will best drive and deliver the outcomes of the Advancing Hyku project. This needs to be an iterative governance process, monitored during the implementation phase, and if any variance is detected corrective measures need to be applied.

Towards the closing of the Advancing Hyku project, the British Library will review progress that has been made.

5 Further Reading

Please refer to the information provided in the links of this section that will help the reader to understand the Hyku community and various things about the repository applications in general.

- Hyku is the official name of the repository product that is a main deliverable of the Hydra-in-a-Box project. https://wiki.duraspace.org/display/samvera/Samvera+Hyku+Interest+Group
- Visit https://hyrax.samvera.org/ for historic information about Samvera components.
- Advancing Hyku project partners are University of Virginia Library, Ubiquity Press and the British Library. The work funding is provided from Arcadia, a charitable fund of philanthropists Lisbet Rausing and Peter Baldwin. https://advancinghyku.io/
- The shared research repository (open access) service provided from the British Library on a multi-tenant model for a community of institutional repositories such as, with a shared layer and individual tenant's repository searches. https://iro.bl.uk/
- Open Access repository service provided by University of Virginia – Libra is a scholarly institutional repository. It provides safe and secure storage. The contents include journal articles, theses, and other completed scholarly works. https://www.library.virginia.edu/libra/
- Architectural guidelines and principles are based on the open group framework that provides the standards and references for methods and designs. https://www.opengroup.org/togaf
    - Google cloud platform. https://cloud.google.com/docs

    - Amazon cloud platform. https://docs.aws.amazon.com/s3/?id=docs_gateway
- Other tools mentioned:
    - SideKiq: https://github.com/mperham/sidekiq
    - Redis: https://redis.io/
    - ZooKeeper: https://zookeeper.apache.org/
    - Memchached: https://memcached.org/
    - Solr: https://lucene.apache.org/solr/
    - Kubernetes: https://kubernetes.io/