Proposal for Archiving LFA Documents in IBM Content Manager OnDemand (CMOD)

1. Executive Summary

This proposal outlines a strategic approach for archiving LFA documents using IBM Content Manager OnDemand (CMOD). The solution is designed to efficiently manage large-volume, high-compression text-based files while supporting long-term retention, rapid retrieval, and seamless integration with existing infrastructure.

2. Document Characteristics and Constraints

- File Size: LFA documents frequently exceed 2GB, surpassing CMOD's single-document storage limit.

- Format: Files are bar (|) delimited CSVs with a semi-structured XML-style header containing metadata.

- Compression Expectation: Anticipated storage footprint is approximately 30% of original file size due to high compression efficiency.

3. CMOD Configuration Overview

3.1 Application Groups

- Design: Ten (10) parallel application groups with identical configurations to facilitate high-throughput ingestion.

- Naming Convention: LFA0 through LFA9

- Index Fields: In addition to existing custom indices, a technical field PARTNO (integer) will be introduced to support document chunking.

- Retention Policy: 10 years (3,650 days) with load-expiry type; legal hold is not required.

3.2 Storage Configuration

Two storage options are available:

1. HCP (S3-compatible) – Current LA Hub configuration.

2. IBM Storage Protect (TSM/VTS) – Managed by the infrastructure team.

Selection should be based on:
- Storage performance
- Cost efficiency per GB
- Operation model

4. Application and Folder Structure

- Folder: All application groups will be organized under a common folder: LFA

- Applications: Each group will be assigned its own CMOD application configured for:
  - Generic indexing
  - Document type: Line (Text)
  - Compression format: OD77

5. Data Ingestion and Interface

- Delivery Method: Irregular monthly bursts via MFT or FTP over SSH

- Naming Convention: To be defined

- Encoding: Specified in file headers

- Drop Zone:
  - Input: /ars/spool/input/lfa
  - Output: /ars/spool/output/lfa
  - Includes receipt management for delivery confirmation

6. Indexing Process

6.1 External Indexing Tool

- Tool: Proposed implementation using a custom Perl script

- Functionality:
  - Parse semi-XML headers
  - Split source files into ≤2GB segments
  - Assign and track PARTNO identifiers
  - Output .ind index files in round-robin format across 10 application groups

7. Loading and Retrieval

7.1 Loading

- Method: Parallel loading via multiple instances of arsload, each handling its corresponding .ind file and application group.

7.2 Retrieval

Process:

- Request initiated via OCR or other pre-defined requesting method
- Bash script performs arsdoc get to extract relevant chunks
- Chunks are reassembled into final (original) file
- The document is compressed (zipped) and deposited to /ars/spool/output/lfa for collection

8. Migration Strategy

8.1 Current LFA Storage Analysis

Assumption: Input files are not split across multiple tapes. Each tape contains one or more whole files with minor zOS overhead.

8.2 Migration Execution

- Indexing: The same indexing tool will be used to parse data directly from tapes.

- Data Delivery: Binary tapes delivered to /ars/spool/input/lfa using existing LAZAR migration tooling.

- Cut-over Strategy: To avoid incremental rework, we shall transition production to the LA Hub before initiating migration.

8.3 Estimated Timeline

Migration expected to complete within 3–6 months based on historical performance benchmarks (~2.4 TB/day).