U.S. Government Publishing Office
Federal Digital System
System Design Document
Volume I: System Architecture

NextGen Edition

Prepared by: FDsys Program

Programs, Strategy, and Technology
U.S. Government Publishing Office

September 2016

Revision History

Revision	Date	Description
Version 1.0	March 3, 2008	Initial draft
Version 1.1	May 12, 2008	Incorporated comments from PMO and development team. Added logical data model and system-level component model
Version 2.0	September 5, 2008	Restructured SDD to multiple volumes of individual design documents. The current document is updated to focus on high-level system architecture and design. Added details on OAIS model implementation.
Version 2.01	February 12, 2009	Minor updates
Version 3.0	July 11, 2009	Updated with re-factored processing architecture, and metadata edit through an Xform based web app. Section 4.5.6.
Version 3.1	October 10, 2010	Final updates for Release 1.
Version 3.2	October 14, 2010	Final updates for Release 1, incorporated review comments.
Version 3.3	September 12, 2014	Initial update for NextGen.
Version 4	September 3, 2015	Update for NextGen.
Version 5	October 27, 2015	Changes Accepted.
Version 5.1	September 20, 2016	Updated Network Diagram.

Contents

1. Introduction

1.1 Purpose

This document describes the architecture and system design of the Federal Digital System (FDsys) for the U.S. Government Publishing Office. It is a living document that evolves throughout the design and implementation for each release. As necessary, each significant release will have an edition of the document. The first public release was R1C2, also known as Release 1 (January 15, 2009). Updates have been made to this document in version 5 to account for the Next Generation FDsys (NextGen). Note that information about the original architecture and design has been maintained for historical reference and the document has been updated as necessary for NextGen.

The goal of this document is to cover the high-level system architecture and design. It provides guidelines on component functionalities and how they are related to each other at the architectural level. The detailed designs that support the architecture are documented separately in the individual component design documents.

The current document starts with the system architecture, followed by various architectural topics, such as content packaging model, data migration strategy, business process flows, etc.

1.2 System Design Document - SDD

The system design document (SDD) for FDsys consists of multiple volumes of individual design documents. In addition to the current document (Volume I), which focuses on high-level architecture and design, separate detailed design documents are created for each of the major components of the system and data management documents for each type of the publications that are managed by the system. The volume numbers for the SDD are in general consecutive, but gaps do exist because some of the originally planned documents were either consolidated or superseded as the design and implementation proceeded.

1.3 Audience

The design documentation is in general for anyone who wants to understand the system architecture and design of FDsys. The following groups are in particular the intended audience of the document.

- FDsys program managers – to evaluate that the system architecture and design support the requirements
- FDsys development engineers – to understand the architecture and follow the design to build the system
- FDsys administrators – to understand the internal workings of the system in order to administer the system effectively
- FDsys operators – to improve productivity while using the system on daily basis
- FDsys maintenance engineers – to understand how the system was built in order to be able to perform any enhancement or reengineering work.

1.4 Document Organization

The current document is organized as follows.

Section	Purpose
Section 2: System Overview	To describe the purpose of the system, and provide a conceptual design, along with some high-level design considerations.
Section 3: Scope of Release 1C.2	To describe the scope of the R1C2, and the incremental development approach for FDsys implementation.
Section 4: System Architecture	To present the system architecture of FDsys, by viewing the system from various perspectives.
Section 5: Content Packaging	To describe the content packaging concept of FDsys.
Section 6. Data Model	To describe the conceptual data model of FDsys, and implementation strategy.
Section 7: OAIS Functional Model Implementation	To describe how the OAIS model is implemented in FDsys
Section 8: Business Process Implementation	To describe how content flows in the system, and the workflows that support daily FDsys operations.
Section 9: Data Migration	To describe how to migrate the content data to FDsys
Section 10: Application Integrations	To provide guidelines on application integration with FDsys
Section 11: Hardware Architecture	To summarize hardware architecture for FDsys

1.5 References

Concept of Operation for the Federal Digital System.
Federal Digital System Requirements Document 3.2.
Reference Model for an Open Archival Information System (OAIS).

2. System Overview

2.1 Purpose of the FDsys

As stated in the Concept of Operations, FDsys will enable federal content originators to easily submit to GPO the content that can be authenticated, versioned, preserved, and delivered upon request. FDsys is the content management system for federal publications within the GPO enterprise, and is at the core of one of the three pillars of the GPO information systems – the Content Management Systems (CMS) pillar. Figure 2.1-1 depicts the conceptual three pillars. The CMS pillar interoperates with the Business Information pillar for financial transaction and ordering management, and with the Digital Production pillar for content and printing specification management. In addition to the core content management functionalities, the CMS pillar is also responsible for managing the acceptance of printing orders and the transmission of content and orders to internal and external service providers.

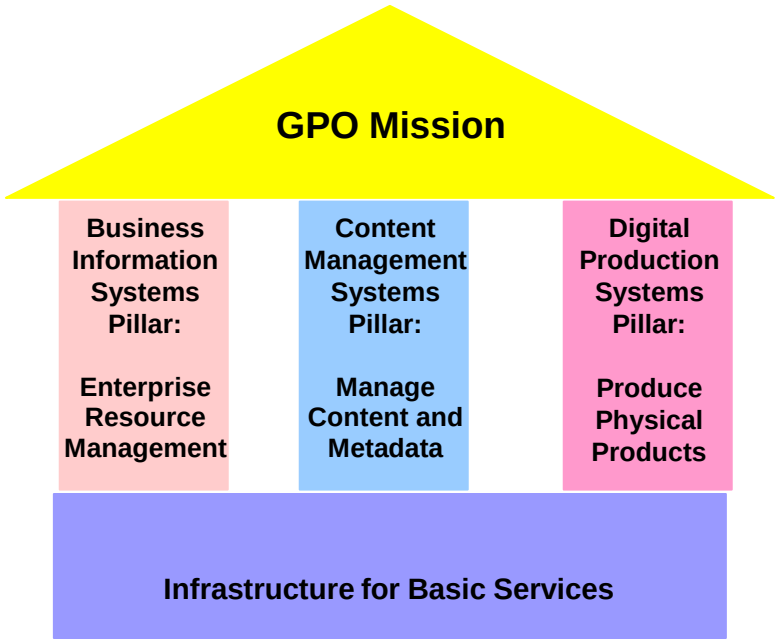

Figure 2.1-1: Three Information Systems Pillars

Within the content management systems pillar, FDsys provides a platform to support business functionalities in the following three areas: content management, content preservation and content access.

Content Management

This is to support daily operations of content management, including content submission, versioning, update on content renditions and metadata. In addition to the content management, FDsys also has the flexibility to handle if necessary, in post R1C2 release, collecting content-related business process information,

such as content ordering. Therefore interactions with applications in other two information systems pillars in later releases are also supported in this functional area. As of June 2015, the functionality is planned to be handled by the GPO DASH system and not in FDsys.

Content Preservation

While content management is at the heart of its functionality sets, FDsys goes beyond what the standard enterprise content management systems provide. One of the critical missions of FDsys is to preserve the content in its original form and to perform preservation processing on the content and technology refreshment to achieve the goal of making the content permanently accessible.

Content Access

In addition to the strategic mission of long-term content preservation, another critical mission for FDsys is to become the next generation of the GPO Access for content access and dissemination. The current GPO Access system was built more than a decade ago with a primary focus on making the government publications available online. Its architecture and enabling technologies have shown serious weakness to efficiently support its business functionalities without frequent and intensive manual interventions.

The access component of FDsys have subsumed functionalities of the current GPO Access with a new architecture and design supported by modern technologies. Since its high visibility is currently supported by a failing architecture along with the complexity of the processes involved in daily operations, the current GPO Access was replaced as one of the high priority features for the first public release of FDsys, the R1C2 release. Note: GPO Access was retired in March 2010.

2.2 FDsys Conceptual Design

2.2.1 Three Subsystems View

To accomplish its missions, conceptually FDsys has three major subsystems as depicted in Figure 2.2-1. Two separate content repositories are created respectively for the content management and content preservation subsystems. These two subsystems are accessible only within the GPO intranet. The repository for the content management is to support daily operations of FDsys, such as accepting content submission, updating existing content and metadata in the system, and publishing content and metadata for public access. The archival repository is to support content preservation. Preservation processes in post R1C2 will all be performed on the archival repository.
The two repositories communicate with each other when necessary, but each has its own independent storage for the content.

The access subsystem is in the DMZ for public content access and dissemination. The publicly accessible FDsys packages are published from the content management repository to the access subsystem, which processes the content and associated metadata and make them available online for general public access.

This high level conceptual view of the three subsystems will be reflected in the system architecture and application designs throughout the system design documentation.

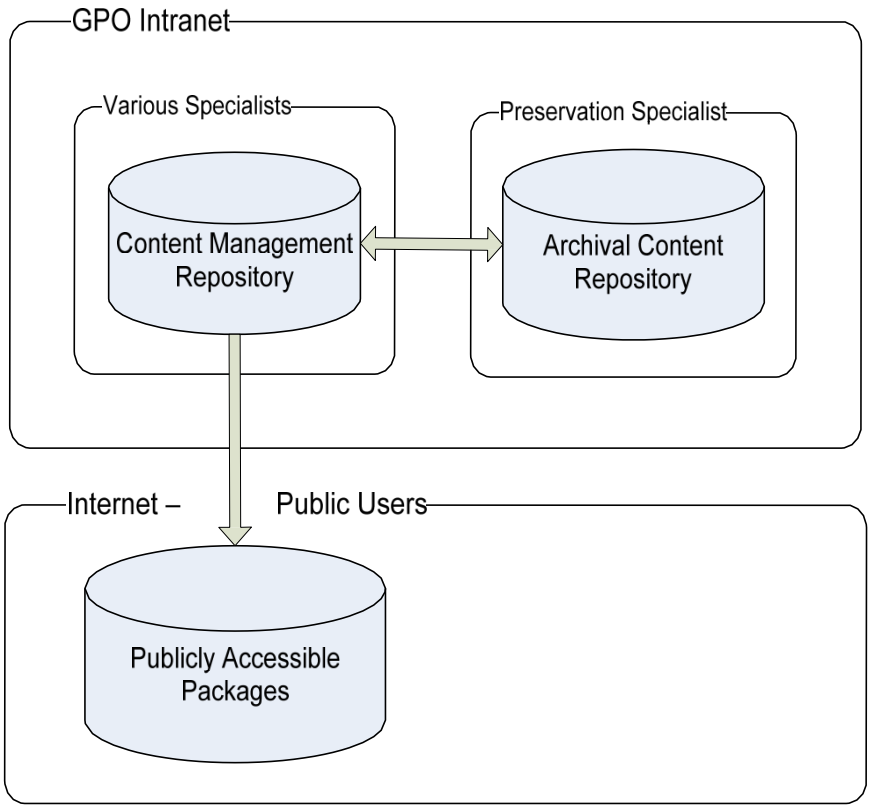

Figure 2.2-1 FDsys Conceptual Design

2.2.2 User Characteristics

FDsys supports two categories of users: authorized users and public users. The content management and preservation subsystems are only accessible to the authorized users. Authorized users are further categorized to functional specialists and system administrators or managers. The following lists the specialists and managers that are supported in R1C2, but due to shifting business priorities, functionality has not been implemented for select user classes indicated below.

- Internal Service Provider
- Congressional Publishing Specialist
- Web Management Specialist
- Cataloging Specialist (not implemented)
- Collection/Content Management Specialist
- Collection/Content Manager
- Preservation Specialist
- System Administrator
- ESB Administrator (not implemented)
- Workflow Administrator
- Business Workflow Manager (not implemented)
- System Administrator Manager
- Security Manager

Each authorized user will be granted roles that determine what the user can perform once successfully authenticated by the system. Preservation subsystem further limits its accessibility to authorized users with preservation privileges. Details of the application security and the user roles and content groups that enforce the access security measures are addressed in detail in the Repository Design document.

The access subsystem is open to public users without authentication. In post R1C2, optional personalization will be provided to public users to customize their web pages appearance. As of October 2015, this functionality has not been implemented.

2.3 Design Considerations

To meet the requirements, FDsys is designed and implemented with the following considerations.

- To follow the Open Archival Information System (OAIS) reference model
- To manage metadata in XML
- To align with the GPO Enterprise Architecture (Note: System design occurred in 2008.)

2.3.1 FDsys and OAIS Model

2.3.1.1 The OAIS Reference Model

The OAIS reference model was developed by the Consultative Committee for Space Data Systems (CCSDS), to serve as a standard for digital repositories for preservation purposes. The recommendation of the reference model issued by CCSDS in 2002 was adopted by the International Organization for Standardization as the ISO 14721:2003. According to the ISO 14721:2003, an OAIS is “an archive, consisting of an organization of people and systems, that has accepted the responsibility to preserve information and make it available for a Designated Community”. Note: ISO 14721 was updated in 2012.

Aiming to be a context-neutral and high-level reference model, the OAIS model describes the environment, functional and information models, and responsibilities that apply to an OAIS-compliant archival system.

The OAIS environment model is concerned with the producers and consumers of materials that are delivered to and obtained from the OAIS. It also covers the management of the OAIS itself. A special class of the consumers is the Designated Community, which should be the primary user group of the OAIS, and the OAIS must be able to appreciate the community’s knowledge base in order for information supplied by the OAIS to be understandable by this user group.

The OAIS functional model addresses functional activities for the following entities:

- Ingest
- Archival Storage
- Data Management
- Administration
- Preservation Planning
- Access

The OAIS information model describes the concept of Information Package and defines what should be included in the Information Package. The OAIS model proposes three Information

Packages: Submission Information Package (SIP), Archival Information Package (AIP), and Dissemination Information Package (DIP). Each information package includes digital objects to be preserved, metadata required to describe the digital objects, and the packaging information that associates the digital objects with their describing metadata.

Finally, the responsibilities of an OAIS archive required by the OAIS model are:

- Negotiate and accept information from Producers
- Determine which community should become the Designated Community
- Ensure the Information Packages are independently understandable
- Ensure the Information Packages are preserved
- Make the preserved Information Packages available

2.3.1.2 Impact of OAIS Model on FDsys Design

As clearly indicated in the ConOps and specified by a set of specific requirements, FDsys will follow the OAIS reference model to manage the content lifecycle. While some of the OAIS entities, such as Data Management for the archive, can be mapped to implementations of relevant functionalities from the commercially available enterprise content management systems (CMS), the commercial CMS products are in general not designed to conform to the OAIS model - the OAIS information model for long term preservation in particular. This presents a challenge for FDsys to implement the reference model by using the out-of-box features of a commercial CMS product.

Every commercial CMS product has its own proprietary data model for the content it manages. Though the implementation approaches vary from one product to another, the separation between the content and metadata is common to all data models of the COTS CMS products. While the content may be stored in various storage devices such as file system, the metadata are normally stored in a persistence store such as relational database. How the association between the metadata and content is modeled and managed varies widely between the CMS products, and has become one of the key differentiators between the competing products.

A simplest and easiest implementation of a content management system would be to use the out-of-box content data model of a COTS CMS, and leverage the application tools usually bundled with the CMS offering to manage the content lifecycle with little customization. Apparently the implementation of this type creates a total dependence on the underlying CMS, and has little flexibility to adapt to technology evolutions over time. This approach, therefore, will be unable to fully meet the FDsys requirements for its independence of the underlying supporting technology.

By the information package concept, the OAIS model proposes a high level abstraction that creates the opportunity for an implementation independent packaging scheme. This is especially beneficial for long-term preservation, which normally has to outlive the lifecycle of the underlying technology that facilitates the preservation process. Through its carefully designed content data model and self-describing archival package, FDsys provides an implementation of the OAIS AIP by leveraging the content management capabilities of the COTS CMS product with an XML-based abstraction layer to minimize the dependence on the underlying CMS product. Details of the FDsys implementation strategy for the OAIS model can be found in Section 5 and 7.

2.3.2 XML in FDsys

Metadata management is at the heart of all content management systems, and is one of the most key functionalities of FDsys. Unfortunately the commercial CMS products, as mentioned earlier, all have their own proprietary metadata models, which in fact have become one of the critical

differentiators between the competing products. The non-standards based metadata models present a problem for FDsys to accomplish its mission - to preserve and disseminate the content and metadata over an indefinitely long period of time. This mission requires that FDsys implementation remain flexible and not tied to any proprietary CMS implementation, and must to be able to adapt to technology changes over time.

To achieve this goal for long-term preservation, the FDsys requirements specify that all metadata for FDsys content must be in XML form, promoting an implementation that manages the metadata for FDsys content independently of the metadata model of the underlying CMS product. The requirements also reflected the fact that most metadata standards from library and other information management communities are in XML form. Managing metadata in XML enables FDsys to easily interface with other systems when needed.

It is noted that managing the metadata solely in XML and independently of the underlying CMS metadata model requires extensive customization. FDsys meets this set of requirements, along with the OAIS packaging requirements, by implementing an abstracted packaging service on top of the supporting commercial CMS product. The XML files containing the FDsys metadata are treated by the underlying CMS as regular content files, and the packaging service extracts the descriptive metadata from the CMS metadata model and populates them to the XML metadata files in the archival package. With all descriptive and technical information stored in and available from the XML files for a package, the package becomes self-describing and independent of the CMS that is used to create the package.

It should be pointed out that these XML files are for metadata only, majority of the FDsys content are not in XML form, but in file formats like plain text, PDF, TIFF, etc. Therefore the native XML applications offered by a few commercial CMS products (e.g. Documentum) for the XML content management are not applicable to the FDsys content in this category.

FDsys may accept submissions where content themselves are in XML or SGML format. For R1C2, the content in these XML-like formats will also be treated like any other content files (e.g. PDF).

2.3.3 FDsys and GPO Enterprise Architecture

While the GPO Enterprise Architecture is still being developed, FDsys will conform to the established models within the evolving enterprise architecture. The operating platforms for hardware and software, development languages, persistence store for FDsys all conform to the specifications in the TRM (Technical Reference Model) of the enterprise architecture. Specific tools in the TRM for enterprise use, such as LDAP for user authentication, and ESB for enterprise application integration, may also be utilized in FDsys, as necessary.

It should be noted that because of the evolving nature of the enterprise architecture, FDsys might select technologies or services that are not covered by the enterprise architecture yet. In this case, the selection will be approved by the Technical Change Control Board. Once approved, the selected technologies or services will be considered for incorporation into the evolving Enterprise Architecture.

3. Scope of Release 1C.2

According to the FDsys requirements, the scope of FDsys covers a large number of business functional areas. It is impractical, if not impossible, to develop and deploy the whole system of FDsys at once. An incremental approach for the FDsys development must be adopted to reduce the high risk associated with the all-at-once approach in terms of cost, schedule and overall success of the system implementation.

The Program Management Office (PMO) for FDsys has performed a detailed analysis on the requirements, and divided the requirements into feature groups with priority assignment. The priority assignment has taken into account the inter-dependency of the feature groups. For example, the infrastructure must be laid out first as a platform to enable FDsys operations. The packaging scheme following the OAIS model must be ready before FDsys can accept and process any content for preservation and public access. As a first public release, R1C2 focused on the following feature groups.

3.1 FDsys Infrastructure

The features in this group are to provide a system infrastructure onto which the FDsys is deployed and performs its operations. The features include hardware infrastructure, security architecture, system availability, backup and monitoring, and integration for FDsys to interface with external systems.

3.2 FDsys Information Packages and Metadata Management

This feature group is for FDsys to implement the concept of the OAIS Information Packages and to manage the metadata in XML with a set of established metadata standards. It must be implemented before any content and metadata can be accepted and processed by FDsys. The implementation of this feature group must be complete to cover content of various types that are ingested into FDsys at the first release or later releases, because modifications to the content packaging scheme in later releases are highly undesirable and will be prohibitively costly. One exception to the completeness is the supported metadata standards and the system must be designed in such a way that FDsys is able to support new metadata standards introduced in the future. This feature group is considered the fundamental foundation of FDsys for content preservation and public access. It will be one of the primary focuses of the R1C2 development.

Once the packaging scheme and metadata management are ready, FDsys is able to create and preserve the Archival Information Packages (AIP). While supporting functionalities for preservation processes, such as content refreshment, are planned for later releases, R1C2 release will have the AIP created and preserved in an archival repository that is separate from the working repository for daily content and metadata management as shown in the FDsys conceptual design. The preservation processes in later releases will operate on the archival repository created at R1C2.

3.3 Content Submission

The content source for R1C2 will be primarily from Congressional and OFR submission by GPO Plant Operations. The existing business processes established in the Plant Operations will continue to process the content files up to the point where the subsequent processing is to produce WAIS-specific file formats. The content currently available from the WAIS database will be migrated to FDsys, and a submission tool for the migration will be developed in R1C2.

In addition to the migration submission, two types of submissions will be supported in R1C2 for the day-forward content. One is the interactive submission where the authorized users upload the content files and metadata to FDsys through a browser-based FDsys user interface. The second type is a folder-based submission where content files are placed to a predefined hot folder for submission to FDsys. Once the content files are submitted through the hot folder, the authorized users will use the same browser-based FDsys user interface as in the interactive submission case to manage the content files before the final submission for ingest. For select collections, an automatic submission job has been enabled and is run on behalf of authorized users.

The content submission for R1C2 is further discussed as part of the GPO Access replacement, and in the Repository Design document. Individual collections are further described in the Data Management Definition (DMD) document for each collection.

3.4 GPO Access Replacement

This group of features is to subsume functionalities available from the current GPO Access, with additional significant improvement features. This feature set depends on the successful implementation of the packaging and metadata management features. It is another primary focus of the R1C2 release that was initially deployed to Production on January 15, 2009.

3.4.1 Incremental Approach

GPO Access operated on a WAIS infrastructure, a client-server text search system that offers little metadata management capabilities. A few metadata pieces in the WAIS content data are created by the GPO homegrown utility programs before the content is passed over to WAIS indexing process. Some metadata information is embedded in the content directory structure or file names. The lack of a consistent and complete metadata set in GPO Access implies that all the metadata required by the FDsys information packages must be systematically extracted before the WAIS content can be migrated to and processed by FDsys.

Metadata extraction from the content is a challenging task, and a few commercial tools are available but mostly for content of forms with predictable format patterns. FDsys adopts the approach to parse the content and extract the required metadata information through custom parsers. Because of the complexity of the publication structures and large number of variations of the structures from one publication collection to another, the parsers have to be developed for each of the collections while maximizing the usage of common metadata elements that can be applied across collections whenever possible. The parser development is a tedious and time-consuming process; multiple iterations are usually required to complete a parser for a particular collection with an acceptable level of accuracy rate.

As such, FDsys development takes a phased approach for GPO Access replacement to meet the target objectives for each release. For release R1C2, the objectives are:

- To migrate all existing GPO Access content to FDsys. Once ingested the content are managed in the form of FDsys information packages.

- To extract from the content a complete set of metadata elements that are required by the preservation and access for the selected high priority collections. For these collections, FDsys will be able to subsume functionalities available from the current GPO Access with additional improvement features.

- To extract from the content a basic set of metadata elements for the mandatory preservation and access needs for the remaining collections. For these collections, FDsys will be able to subsume most functionality available from the current GPO Access while the additional improvement features will be added in later releases because of the incomplete metadata set.

- To maintain granule-level access to publications available from the current GPO Access for the selected high priority collections.

- To provide backward compatibility for access capabilities, such as getDoc, currently utilized by systems external to GPO.

A successful completion of the above objectives will move the current GPO Access off the WAIS infrastructure, which has been the origin for many notorious problems. This phased approach requires a careful design of the system to support adding features in later releases to the content already in FDsys archival repository and also published for general public access. By this phased approach, the features that are planned for later releases are primarily concerned with adding additional metadata and relationships between the publications to the packages. This seemingly extra effort is in fact also covered by the FDsys requirements for post R1C2 releases – ability to edit metadata for content already in the system and to manage the relationships between publications. Note: The NextGen Publication Linking feature has been developed to manage relationships between publications.

3.4.2 Transition to FDsys

Figure 3.4.2-1 shows a schematic diagram of the current processes whereby the content are processed and made available online at the gpoaccess.gov website. The diagram shows only the main activities that concern the content migration and process transition to FDsys, and is not meant to describe all and detailed processing steps.

The current content processing involves many manual steps to invoke GPO homegrown tools or scripts to perform the processing tasks. The functionalities of the tools range from simply formatting the text files with appropriate return characters to processing the locator and SGML files and adding tags to the files. These content composition functionalities are addressed by a separate project currently under development – the Composition System Replacement (CSR). As such, subsuming functionalities of the tools used in the current processing is out of scope for FDsys.

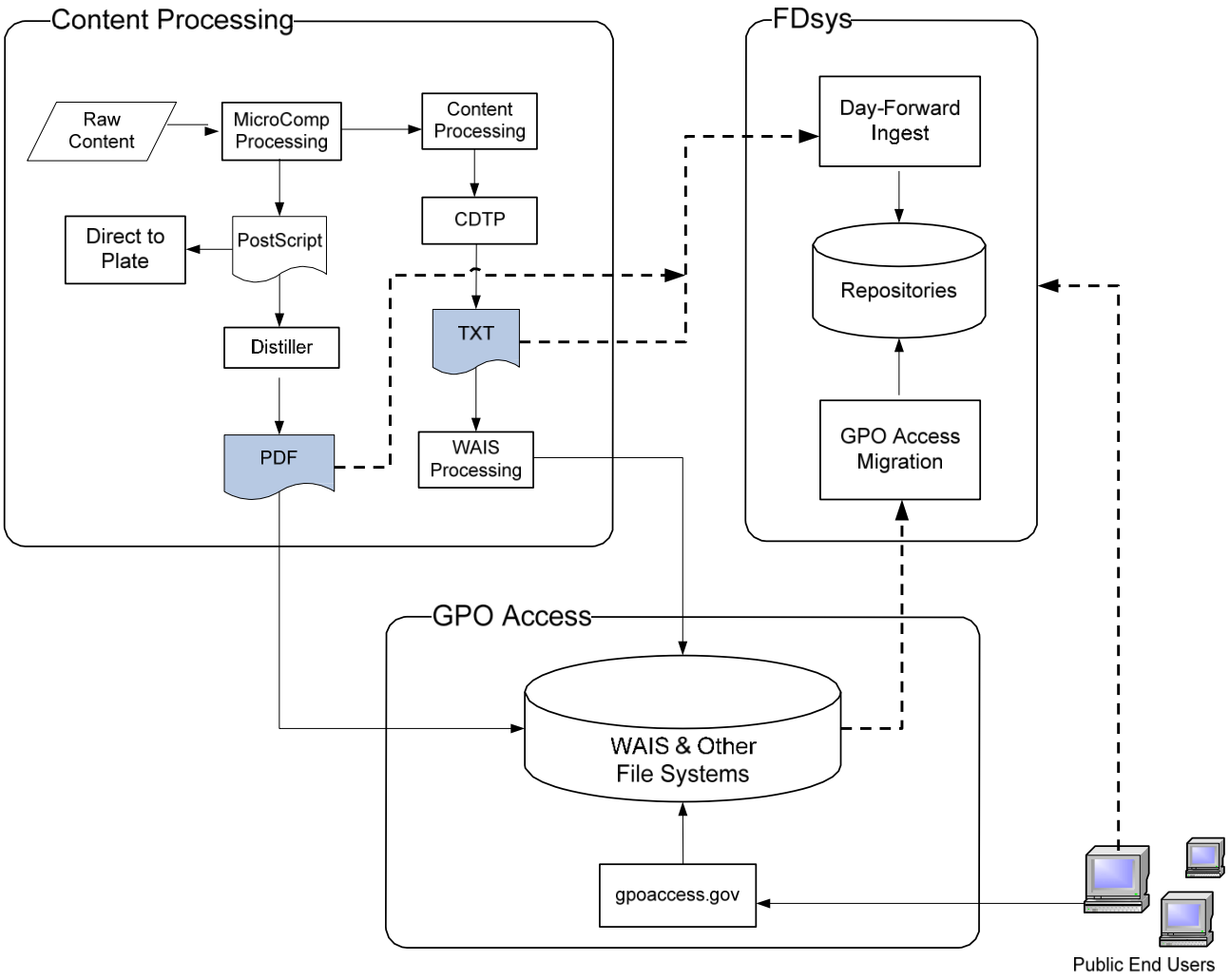

Figure 3.4.2-1 Current GPO Access Processing and Planned Transition

GPO Access replacement involves two major parts: migrating exiting GPO Access content to FDsys, and enabling a process and ingesting new or day-forward content into FDsys, as indicated by the dashed lines in Figure 3.4.2-1. For migration, the content source includes the content files from the WAIS infrastructure and some additional files from other file systems (Jukebox, Alpha3) currently maintained by the Plant Operations. A migration tool will be developed to traverse the migration files in the staging area for all collections, and feeding the content and a few metadata (that is only available from the original directory structure) for ingest to FDsys. Details of the tool design and the ingest process will be addressed in a later section.

For the day-forward content submission, R1C2 will leverage the current content processing up to the selected steps. One such step is after the creation of the screen-optimized PDF file. The PDF files are created manually using the Adobe Distiller in the current processing, along with a table mapping the PDF files to the corresponding text version of the files. Another step is after the homegrown tool – CDTP is invoked. The output of the CDTP (i.e. .done file) is a cleanly formatted text file with necessary tags inserted (i.e. <doc>), if applicable, to indicate the granule separation of the publication. Processing steps after these two are either specifically to prepare the text files for WAIS

indexing or to make the PDF along with other files available to content subscribers. For R1C2, processing steps to further prepare the files for WAIS can be retired after the transition period.

As shown in the dashed arrows in Figure 3.4.2-1, interactive and folder-based tools will be developed to submit the PDF and the CDTP output file, along with other appropriate renditions in other file formats (such as locator, SGML when applicable), to FDsys. Similar to the migration case, the parsers for the day-forward content will be applied to the text version of the content and the extracted metadata along with the original content files submitted for ingest. Details of the day-forward content submission and ingest will be addressed in a later section.

Since the current GPO Access is a live system, the industry best practices have shown that a transition period is required to switch from a live system to its replacement to mitigate the high risks associated with a sudden switch. During the transition period, the current GPO Access and FDsys will be in operation in parallel, providing similar functionalities for public access, while FDsys will have improved features in selected areas. It should be noted that the transition period concerns the smooth launch of the replacing system, and in principle should have little impact on the architecture and design of the new system. A similar strategy will be followed with the launch of the Next Generation public website (NextGen).

3.5 Summary

In summary, the scope of the release R1C2 is to complete implementations of the above listed feature sets to build the foundation of FDsys packaging scheme for preservation and public access and to replace the current GPO Access with the access subsystem. The content will be managed in the form of FDsys information packages, and also stored as AIPs in a separate archival repository of FDsys. From the implementation point of view, R1C2 will serve as a foundation for later releases to add more features or to enhance the functionalities of the system, completing the FDsys development in the incremental releases. R1C2 was launched in January 2009 and GPO Access was retired in March 2012. NextGen hardware refresh launched in December 2015 and the NextGen UI launched as a public beta in January 2016.

4. System Architecture

FDsys is a software intensive system, consisting of primarily COTS products and customizations that are needed to meet the requirements. System architecture concerns are about the overall structure, the components and their relationships in the system. For complex systems like FDsys, it is rarely sufficient to describe the architecture from a single perspective. The architecture is therefore presented by various views or perspectives of the system.

4.1 Application View

The application view describes the application components of the system, including the selected COTS or open source products, and the customizations that need to be developed. It also depicts the relationships between the components, and their respective roles in the system. Figure 4.1-1 shows the application view. The COTS products were selected by a series of trade studies. From the technical perspective, the key selection factors were how well the candidate product meets the FDsys requirements by the out-of-box features, and how well the programming interfaces are defined by the product for customizations. The FDsys custom applications are developed to leverage the capabilities of the COTS products so the requirements are fully met by the system. Notice that in order to provide a view with all major involved components, Figure 4.1-1 also includes a few existing systems, such as ILS, that FDsys may need to interact with in the future to perform its operations.

Figure 4.1-1 can be viewed as a detailed version of the FDsys conceptual design shown in Figure 2.2-1, but filled with the selected COTS products and planned FDsys applications for each of the subsystems. Functionalities are provided by both the COTS products and FDsys customizations that are required to fully meet the FDsys requirements. Open source technologies are also utilized where feasible in NextGen (e.g. Apache Jena, Drupal)

4.1.1 Content Management Subsystem

A Documentum content repository is setup to perform daily content management operations. The capabilities of Documentum are used for the following operations.

- User authentication – to authenticate users and enforce application level security by the Documentum built-in security capabilities. To store and enforce the security measures, Documentum provides a built-in integration with LDAP, which will be configured for FDsys to manage the application security controls.

- Workflows – the BPM (Business Process Management) of Documentum will be configured to execute FDsys business workflows.

- Content Search and Retrieval – out-of-box capability of Documentum to search and retrieve the content in the repository for authorized users.

- WebTop User Interface – to minimize FDsys customization by leveraging the Documentum out-of-box WebTop application as the standard user interface for authorized users to access the repository.

The FDsys applications are developed to extend or customize the capabilities provided by Documentum. Each of the applications is briefly described below.

Content Parsing and Processing

The purpose of the content parsing and processing is twofold. First it is responsible for extracting metadata from content for preservation and access needs. A set of operation interfaces will be defined to provide input and retrieve output from the parsers. All parsers, regardless of whether common to all publications or specialized to a particular publication, must implement the operation interfaces. The parsing framework also provides a mechanism to plug-in new parsers or to remove (and hence capability of replacing) parsers from the service. This is to enable a phased approach for parser development.

The second purpose of the content parsing is to handle publication granules. The content source for R1C2 is limited to the existing WAIS content and day-forward content from the Plant Operations. The publication granules are currently created by the established manual process and tagged in the input content files. So the remaining job for the content parsing is to interpret the tagged granules, that are normally publication specific, and to save the granules to separate files for access.

It should be noted that the content parsing for publication granules in R1C2 is designed to interpret and transform the granules that are manually created during the GPO Access content processing. Until granules are programmatically identified and generated, this design will continue to serve its purpose in later releases. At present, no tools have been found available to automatically create the granules that meet GPO's granule requirements.

Packaging Application

The packaging application is responsible for managing the FDsys information packages following the concept model of the OAIS. The AIP in FDsys is stored and accessed only for preservation purposes, and a separate archival storage is allocated for the AIPs to achieve this goal. To accomplish another mission of FDsys for content dissemination for public access, FDsys creates an additional package – Access Content Package (ACP) for daily content management. ACP also serves as the source for the content dissemination to the public.

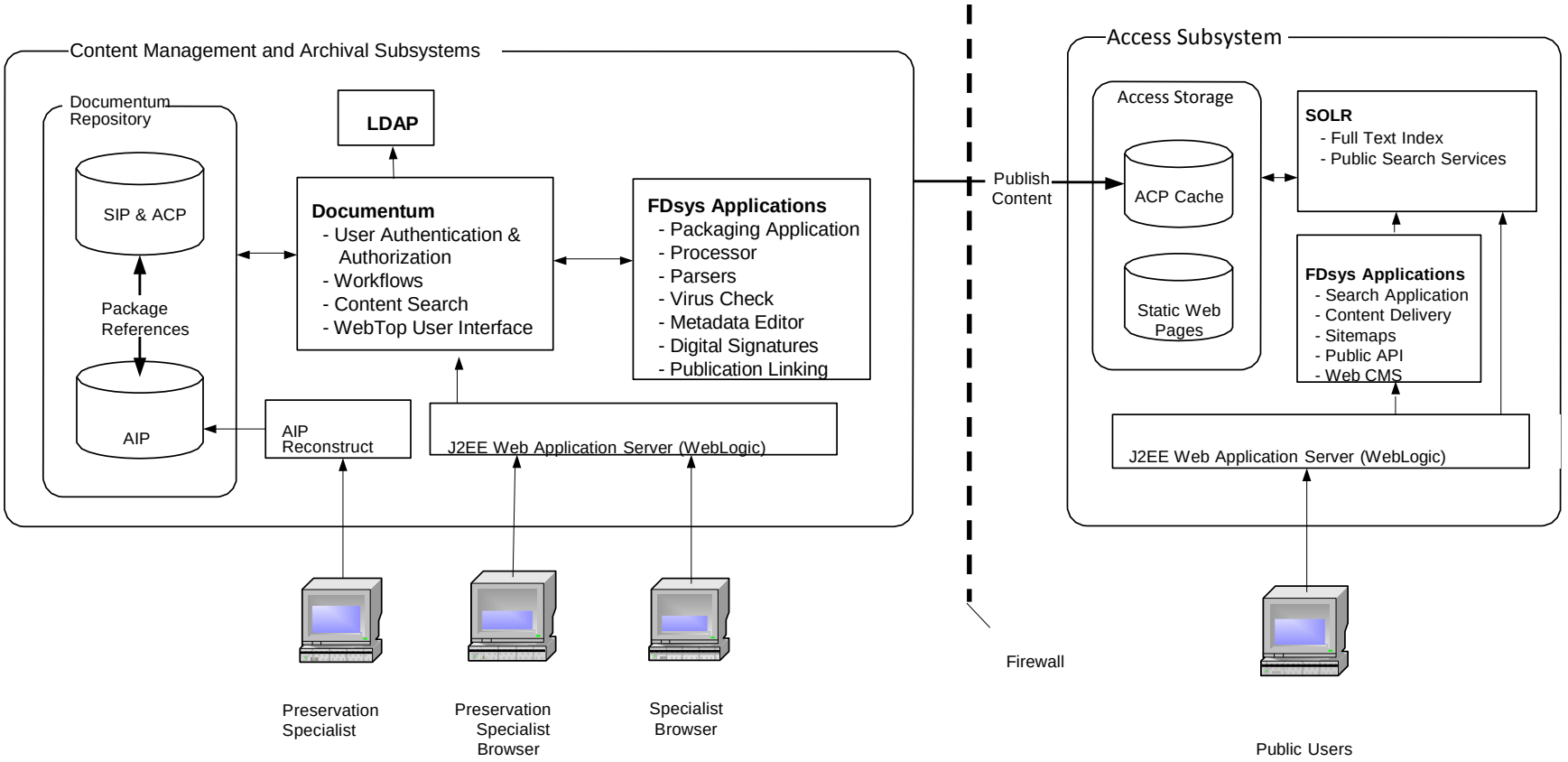

Figure 4.1-1 FDsys Application Architecture

Content Source Module

FDsys has to be able to accept content from a variety of sources, which can vary widely in terms of the way in which content are stored, the availability and format of descriptive information, etc. Examples include the initial migration of content from the WAIS and Jukebox file system, where the directory structures change from one publication to another or from one year to the next for the same publication. It is highly undesirable for FDsys to develop a customized ingest process to glean the content source information each time when a new content source needs to be covered.

The purpose of the content source module is to serve as a front-end application to the FDsys submission process. It probes the content source storage structure, gathers the content source information and returns the collected information to the submission process in a pre-defined and unified form. In other words, FDsys can expect the input information in the same form gathered by the content source module from various sources with completely different characteristics. CMS was used to support the migration from GPO Access to FDsys.

4.1.2 Archival Subsystem

An independent archival repository is setup for the AIPs. Similar to the content management subsystem, Documentum capabilities are leveraged to manage the repository and provide the authorized users access to the repository. Note the content data model in the archival repository is different from that in the content management subsystem, to fully meet the preservation requirements.

The packaging application in R1C2 is to create the AIPs conforming to the archival content data model, and ingest the AIPs to the repository. The preservation processing will perform the basic preservation related metadata management, such as recording the provenance events to the AIPs.

The AIP reconstruct utility is developed as a tool, independent of the CMS, to verify and ensure that the AIP is truly self-describing in that an AIP in the archival repository can be re-constructed to a self-contained package that is truly independent of the underlying CMS which creates and manages the AIPs in the archival repository.

4.1.3 Access Subsystem

For NextGen, SOLR was selected to replace FAST ESP as the search platform for FDsys to make the published content accessible to general public. The search engine performs the content indexing, and provides the full-text search capabilities for publicly accessible content of FDsys. SOLR will be configured to process and index the content and metadata published from the content management repository.

SOLR provides a set of APIs for customizations to submit queries and retrieve the search results. The search front-end for public access is thus a major area of FDsys customizations. WAI-ARIA accessibility will be applied to custom web pages for the access subsystem.

The access subsystem also provides web content management capabilities through Drupal, an open-source web content management system, to allow the authorized users to manage the editable pages for the public site. The examples of the editable pages include posting collection-specific news to the web pages for the collection. In addition, the user help will be available from the web applications for the access subsystem.

4.1.4 ILS Integration

GPO has deployed an ILS (Integrated Library System) to maintain a catalog of the federal government publications since 2002. As mentioned earlier, the ILS and FDsys both belong to the Content Management pillar within the GPO enterprise information systems. From the implementation point view, the ILS is, however, an existing system external to FDsys. FDsys thus has to have the capability to integrate with the ILS for bibliographic information synchronization, to keep both systems up to date about the content available from GPO. The integration between the two systems has been planned to be accomplished through the enterprise service bus.

The ILS integration is planned for implementation post Release 1, as of October 2010. As of June 8, 2015, ILS has not been integrated with FDsys.

4.2 Network and Storage View

This view shows network configurations that enable the two distinct access entries to FDsys, including access security to the storages. Figure 4.2-1 depicts the network and storage view of the system architecture.

There are three logically and physically separated storages in FDsys. AIP storage in NAS (Network Attached Storage) is dedicated to storing the archival packages with the most restrictive access control. Only a small group of users with preservation privileges is granted access. SIP and ACP storage is to support daily content management operations by authorized users, including service specialists, service providers, system administrators, etc. The third storage is for the ACP cache – the published ACP. This storage supports the public-facing web application, providing content access to the general public. The security measures in this architecture are enforced by the VLAN (virtual LAN) configurations and user access roles, in addition to the logical and physical separation of the storages themselves.

For the content management and access subsystems, the content is stored in NAS, but the DAS (Direct Attached Storage) is used to store the search engine indexes on the content for the system performance consideration.

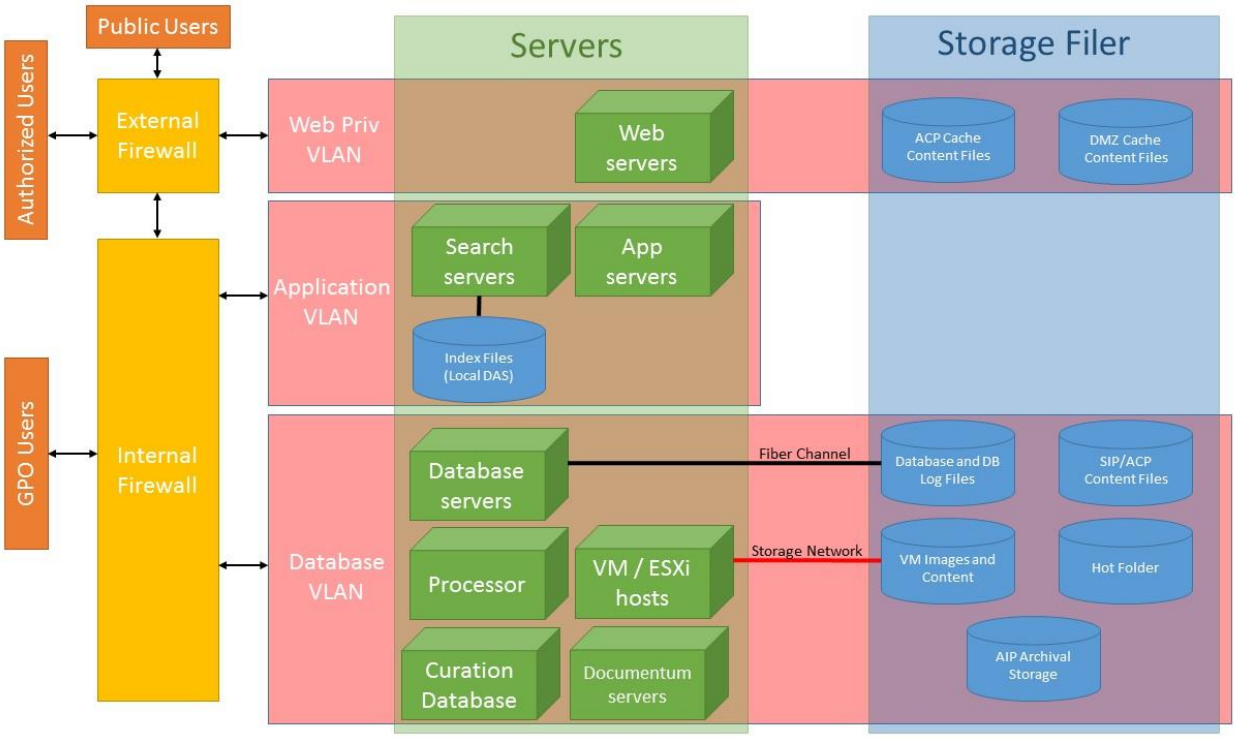

Figure 4.2-1 Network and Storage View

The authorized user access to FDsys through the public-facing web application, indicated by the dashed connection line in the diagram, is not implemented in R1C2. It was originally shown to present a more complete picture of access to FDsys, and for reference to potential features of subsequent releases.

4.3 Application Security View

The application security concerns with the security control over which users could perform what functions on which data set in the system. Following industry best practices, FDsys adopts the role-based approach to enforce the access control for the authorized users.

The role-based approach involves two concepts: roles and groups. A role defines a set of functions a user can perform. For example, a Content Submitter role enables a user to manage the content and metadata in the SIP before submission. An authorized user can be assigned with one or more roles, and thus granted with a set of functions that is the union of all functions permitted by the assigned roles.

An authorized user is assigned with one or more roles, along with one or more groups. The roles specify what the user can do, and the group for which content the user has access. In fact, a user must be assigned at least with one content group; otherwise the user has no access to any content in the system at all. Similarly, a user must at least have one role for the user to be able to perform any operations. The roles and groups will be populated to the LDAP directory information tree and stored in a LDAP compliant server. Figure 4.3-1 shows a view of the LDAP integration for FDsys.

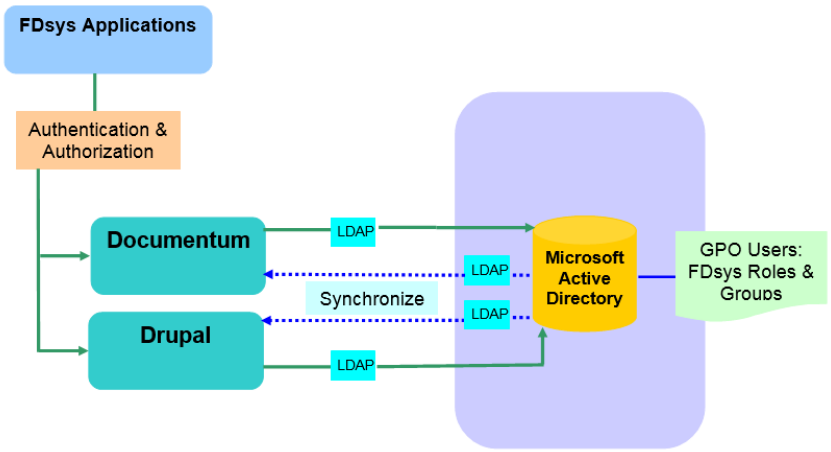

Figure 4.3-1 FDsys LDAP Integration

The role-based security measures need to be enforced by Documentum for direct content related access control, and by FDsys custom applications for functionality control, which indirectly relates to the content as well. As part of the out-the-box features, Documentum delivers a built-in LDAP integration for applications to enforce the role-based security measures. The built-in integration includes the configurable periodic update of the roles and groups from the LDAP server (shown as the dashed line in Figure 4.3-1). To maximize the usage of available capabilities from the COTS products and avoid reinventing the wheel, the current application architecture enforces all security controls through the Documentum LDAP integration, as shown in Figure 4.3-1. This architectural choice has the following benefits:

- FDsys has no need to develop another LDAP integration layer within the FDsys custom applications.

- No synchronization issues between FDsys applications arise as LDAP integration is handled at one place. The out-of-sync directory information in a complex system like FDsys could be a very challenging problem to diagnose and resolve.

- FDsys is a content-centric system, use of Documentum as the central LDAP integration simplifies the content related access control.

Figure 4.3-1 shows two directory servers, one is the Microsoft Active Directory that stores all current GPO user information on the GPO intranet network. The user accounts for R1C2 will be exclusively from the Active Directory, with additional FDsys specific attributes, such as a flag to indicate whether the user is activated for FDsys access. The other directory server is the Oracle Internet Directory, which is part of the Oracle IAMS that has been selected as GPO enterprise identity management. For post R1C2, user accounts for agency users will be stored in this directory. As of September 22, 2014, Oracle IAMS has not been integrated with FDsys and external users do not have direct access to Documentum.

Details of the LDAP schema and Documentum synchronization with GPO directory servers for the roles and groups are covered in the Repository Design document.

Public users access the FDsys public search application without the need to login. To enable the personalization, the subsequent releases of FDsys will provide the optional self-registering process for public users to set personal web page preference. Additional application security consideration are discussed in the System Security Plan and individual design documents for each component.

4.4 Deployment View

Two instances of FDsys will be deployed, one as a primary – Production instance, and another as a standby disaster recovery – COOP instance. A Production instance will be deployed outside GPO headquarters where the Production instance is hosted. The two instances are identical in terms of application architecture and software packages. A high degree of redundancy of the servers for major applications, such as Documentum and SOLR, is provided at the Production instance to distribute the system load and to ensure high availability of the system. The purpose of the COOP instance is for disaster recovery should the Production instance is hit by a catastrophic event, in which case the operations are failed over to the COOP instance. Figure 4.4-1 shows the deployment view.

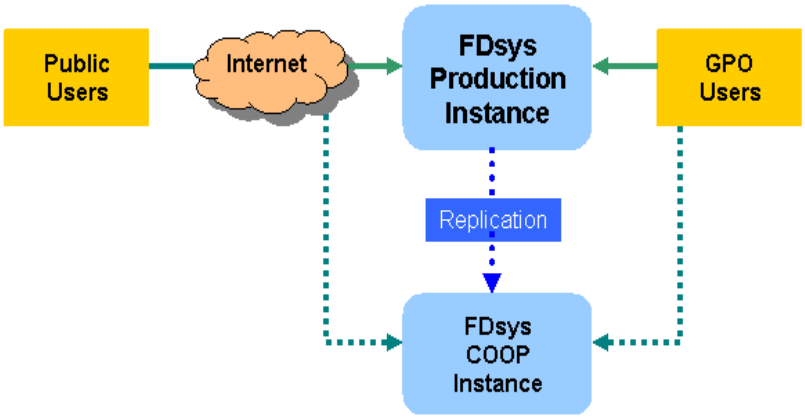

Figure 4.4-1 Deployment View

Refer to SDD volume LVIII for details of the COOP instance design.

4.5 System Component Model

The system component model concerns the high-level software components that implement and support the application architecture described in section 4.1. Following the content lifecycle, the system can be logically divided into four major components: submission, ingest, access and system infrastructure. Each component will be consisting of the combination of primarily COTS applications plus integration customizations when applicable. The components communicate with each other mostly by predefined system events or the API-based application interfaces that are supported by vendors of the selected COTS products.

Fundamental to all the components is the data model, which controls how data flow through the system, as well as the management information that impacts how the system behaves.

Figure 4.5-1 shows the high-level software component diagram for the system. Notice that only the main applications and functionalities are listed for each component, the more detailed lists are described below for each of the components. Also for the infrastructure component, only the software part is depicted in the diagram.

4.5.1 Submission Component

This component is responsible for handling content submission to the system by implementing a number of submission scenarios. The scenarios in R1C2 include GPO Access migration and day-forward content submission from the GPO Plant Operations.

The primary COTS applications in the submission component include:

- BPM-based Documentum workflow for the submission workflows
- WebTop-based user interface for content submission
- Built-in Documentum LDAP integration for authentication and authorization
- Documentum scheduled jobs to support the folder-based submission
- .

The required integration customizations for this component include:

- Workflow activity implementations for the submission workflows
- Submission user interfaces to be deployed to the WebTop framework
- Submission monitor tool for the Documentum scheduled job for the folder-based submission
- Documentum scheduled jobs to ingest content from external APIs
- Content source module to probe and gather package structure information for subsequent ingest and migration toolset to facilitate migration of GPO Access data to FDsys (completed).

4.5.2 Ingest Component

Following the OAIS model, this component accepts the SIPs from content producers, and performs content processing, managing the package lifecycle from the SIP to the AIP and ACP. After the submission component completes its tasks, the ingest takes over the system process flow. The submission and ingest components are communicated through the system events for the workflows.

The main COTS and open source applications in the ingest component are:

- BPM-based Documentum workflow for the processing workflow and associated processes
- WebTop-based user interface for manual actions in the FDsys workflows
- Built-in Documentum LDAP integration for authentication and authorization
- Apache Jena API for building the RDF store
- Apache Fuseki Server for storing Publication Linking information

The main FDsys integration customizations in this component include:

- Implementations for the processing workflow
- Content parsing – to extract metadata and granules from the content files

- Package management for creating AIP for the archival storage during Ingest
- Archival process to create preservation rendition during Ingest
- User interfaces for the WebTop to support the configured manual actions in the workflow
- Schema registry for supported metadata standards
- XSLT-based XML file transformations
- ILS integration (not implemented)

4.5.3 Access Component

This component makes the FDsys content publicly accessible online, and is the replacement for the current GPO Access. The content and metadata are published to the access component after the ingest component completes its tasks. The interface between the two components is by the FDsys integration customization that is based the supported APIs from Documentum and SOLR - the two primary solutions selected by FDsys.

The primary COTS and open source applications for the access component include:

- SOLR search engine for full text and structured XML metadata search
- SOLR APIs for content indexing and query
- Documentum APIs for content publishing and status synchronization
- Drupal web content management system for editable area on select web pages

The main FDsys customizations for this component include:

- Content Publisher to publish content from Documentum and synchronize the publish status
- Search schema and metadata normalization for the schema
- Full-fledged web application for public search based on the SOLR APIs
- Content delivery upon user request
- Section 508 compliance
- User help

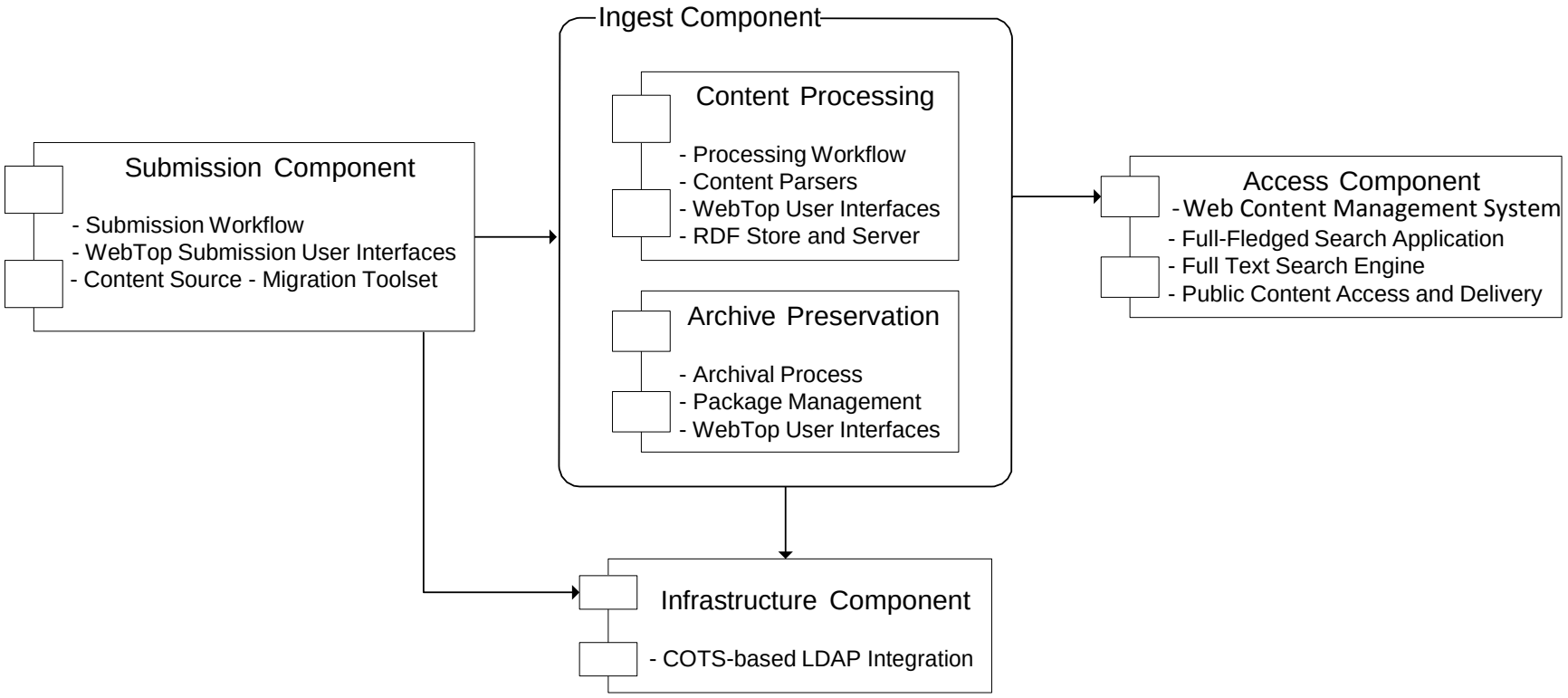

Figure 4.5-1 High-Level Software Components

4.5.4 Infrastructure Component

In addition to hardware, sizing and performance, and application allocation to servers, this component is also responsible for the security infrastructure and monitoring for FDsys. Additional detail about monitoring activities are captured in the Infrastructure Management Document. FDsys adopts the role- based approach to enforce the application level security control, and the roles are stored and managed in the LDAP-compliant directory server as part of the GPO enterprise security infrastructure.

It is worth pointing out that while FDsys will leverage the Documentum built-in LDAP integration capability for user authentication and authorization, the user information and the roles are not stored nor managed by Documentum.

The security-related COTS applications for this component include:

- LDAP-compliant directory servers – the existing Microsoft Active Directory for the current users on the GPO intranet, and Oracle Internet Directory for FDsys roles and content groups as required to support future functionality based on business need
- Documentum and Drupal LDAP integration for user authentication and authorization

No FDsys customization is required for the security enforcement in this component, except a custom LDAP schema may be needed to accommodate the needed LDAP attributes for FDsys. The details are described in FDsys Repository Design Document. The Documentum LDAP integration to the directory servers is accomplished by configuration. Drupal LDAP integration is included in the Responsive Search Web Application design document.

4.5.5 Data Model

The data or content managed by FDsys is the federal government publications. The data model for FDsys primarily concerns another set of data – metadata that describe the managed content in the system. The FDsys data model also contains attributes that help enable the efficient management of the content and control system behavior such as role-based application security.

The impacts of the data model can be found at every stage of the content lifecycle in FDsys, and the data model is thus the foundation for all high-level system components. For submission component, the data model indicates what metadata elements about the content need to be collected, by parsing the content or through the user input, to enable the successful downstream content processing. The data model has direct impact on how the content exist in the system – the FDsys package structure specification as an implementation of the OAIS information package model. The attributes in the data model may be further normalized to facilitate the structured and efficient search capability provided by the search engine (SOLR) for the access component. The security portion of the data model will enable the role-based application security control by Documentum through configuration without the need for customization development.

4.5.6 Architecture Update – June 2009

The approach, until June 2009, to implement the application architecture shown in Figure 4.1-1 was as follows.

- The Documentum object models were constructed in such a way that they were collection specific, and all the metadata elements, including those for the granules, for the collections were mapped to the repository attributes.

- The file processing tasks (i.e. content parsing, granule generation and digital signature, etc) were all performed directly as part of the workflow activities and executed within the Documentum infrastructure.

While the implementation approach met the overall functional requirements, it presented severe challenges in several key aspects of the system.

- The repository deployment had been a challenging task ever since the initial launch with a small set of collections. It was understood that the large and complex object model was a primary contributor to the deployment problems.

- The file processing within Documentum turned out to be extremely challenging as more collections were added to the system. FDsys packages require extensive file processing before they are published for public access, and the Documentum Process Engine, designed for support of majority manual workflow activities, proved unable to meet the massive automatically processing needs of FDsys. For collections with large packages (i.e. large number of granule files, HOB and CRI for example), the processing took 4 to 6 hours to complete for one package, which was apparently unacceptable by any measure.

The collection-specific object model and in-Documentum processing required, and would continue to require, significant development effort whenever new collections are added to the system. The object mode has to be updated and deployed whenever new collections are added as well, which presents a significant limit on extensibility and maintainability of the system.

To address these issues, the repository implementation design was revisited for the object model and processing architecture. An independent assessment by EMC corporation (the vendor of Documentum) was also performed for the object model and associated deployment and scalability issues. The assessment confirmed the need for a re-factoring of the implementation approach.

It is worthwhile pointing out that though the implementation approach was re-factored, the high-level application architecture shown in Figure 4.1-1 remains the same, in terms of major system components and their relationships to each other. This section describes the re-factored implementation approach, and the details of the re-factored repository design are documented in volume II of the SDD – the FDsys Repository Design.

4.5.6.1 Repository Architecture – June 2009

Figure 4.5.6-1 shows the re-factored repository architecture.

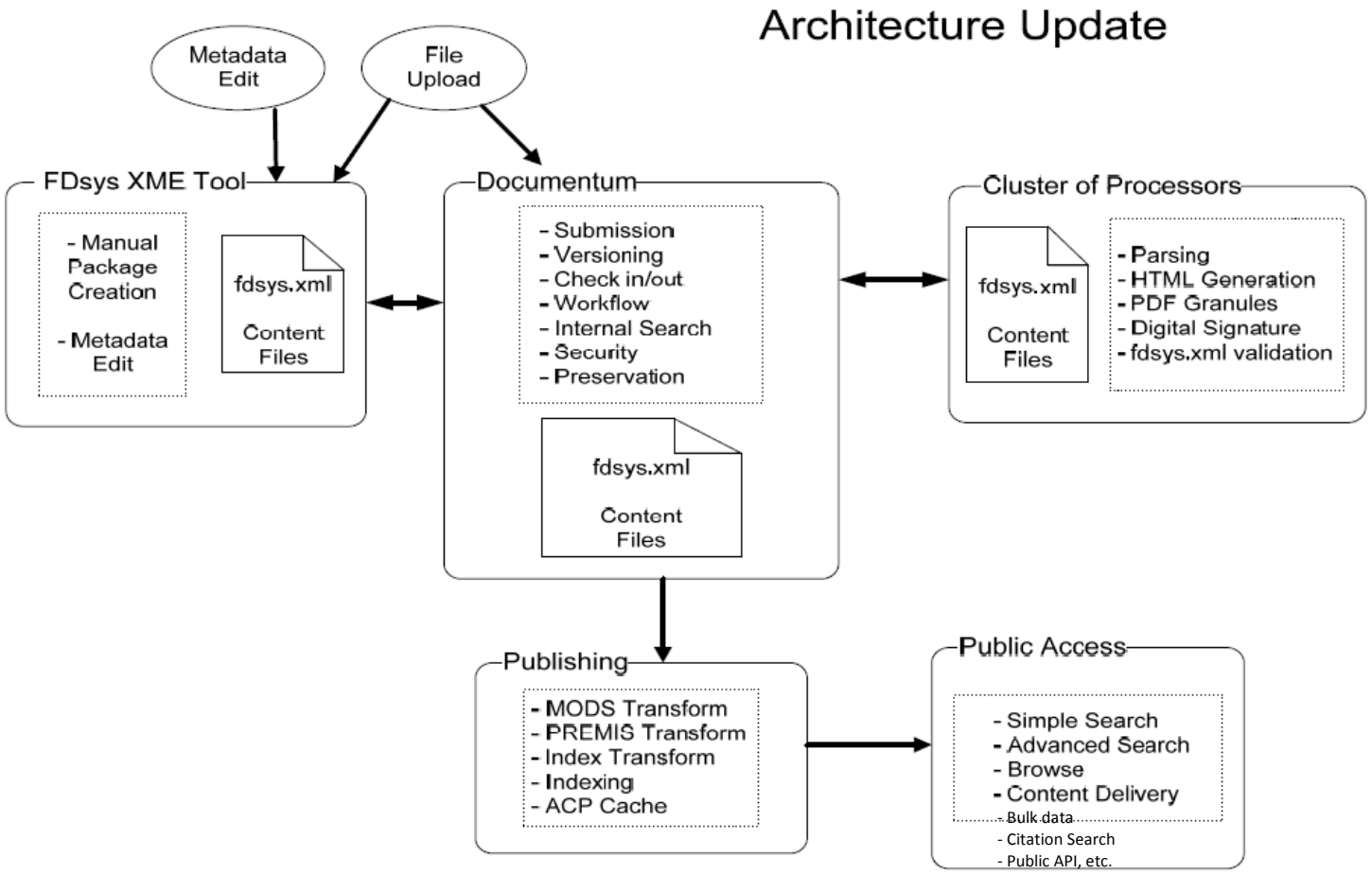

Figure 4.5.6-1 Repository Architecture

The main changes are:

- The repository object model is collection-neutral (see Vol. II of the SDD for details)
- The package processing is outside Documentum, supported by a cluster of file processors.
- Parser framework is called by File Processors.

- The metadata edit is supported by a custom application to work directly with the fdsys.xml

Under this architecture, all FDsys-specific package processing is performed by the processors, each of which is specialized in a particular collection. Documentum provides functionalities of what is designed for with much less customization. The object model is collection-neutral, and collection specific elements of the processing are covered by the specialized file processors. This provides the needed extensibility – to add a new collection to the system only requires a new file processor (which has to be collection specific) without the need to update the object model design and implementation.

The fdsys.xml is communicated directly between the repository and processors, eliminating the once required transformations in and out of the repository data model attributes from and to fdsys,xml. This not only makes the implementation much simpler, but also provides significant performance improvement.

To enable metadata editing, the FDsys XME (Xform based Metadata Edit) tool is developed to support direct edits of the metadata from the fdsys.xml. This tool serves the purpose of what a few commercial or open source XML tools provide – to update the data elements from an XML file. It also enables manual creation of packages ready for submission to FDsys. Until a COTS tool is required and selected for future release, the FDsys XME tool will be integrated with Documentum for metadata editing from fdsys.xml, and package creation.

4.5.6.2 Clustered Processing Architecture

It has turned out that FDsys packages require extensive file processing, to provide the rich search and browse features, before they are published for public access. The processing tasks include content parsing to extract a large set of provenance and access metadata, granule generation (HTML and PDF), digital signature and schema validation. The extensive processing needs seem to have reached the limit of what a single-process can handle for acceptable performance and scalability. A clustered processing architecture is thus implemented to meet the FDsys needs, as shown in Figure 4.5.6-2. The characteristics of the architecture are as follows.

Cluster of Processors – instead of one process to handle all processing needs, a cluster of processors is used. Each processor is specialized in one collection.

Parallel Processing – because of the specialized processors, the processing executes in parallel for various collections. In other words, no processing jobs for one collection need to lineup behind another collection.

Distributable – the processor cluster can be deployed in multiple physical servers, if necessary, to distribute the demand for computing resources.

Highly Configurable – depending on the computing resource needs, one instance of the processor manager can be configured to host single or multiple processors.

Plug-and-Play Capable – when an update is required for a processor, a new build of the processor can be deployed to the relevant instance of the processor manager without impact on other processors. This applies to newly developed processors as well in the future.

Loosely-Coupled Interaction with Repository – the processors interact with the repository through a queuing model, providing the flexible foundation for the cluster configuration and plug-and-play capability.

All the features, combined together, provide the much-needed performance, scalability and extensibility of the system. It simplifies maintenance as well, since many problems and desired enhancements are associated with the content parsers, the clustered processing architecture can support the parser updates easily with the plug-and-play feature.

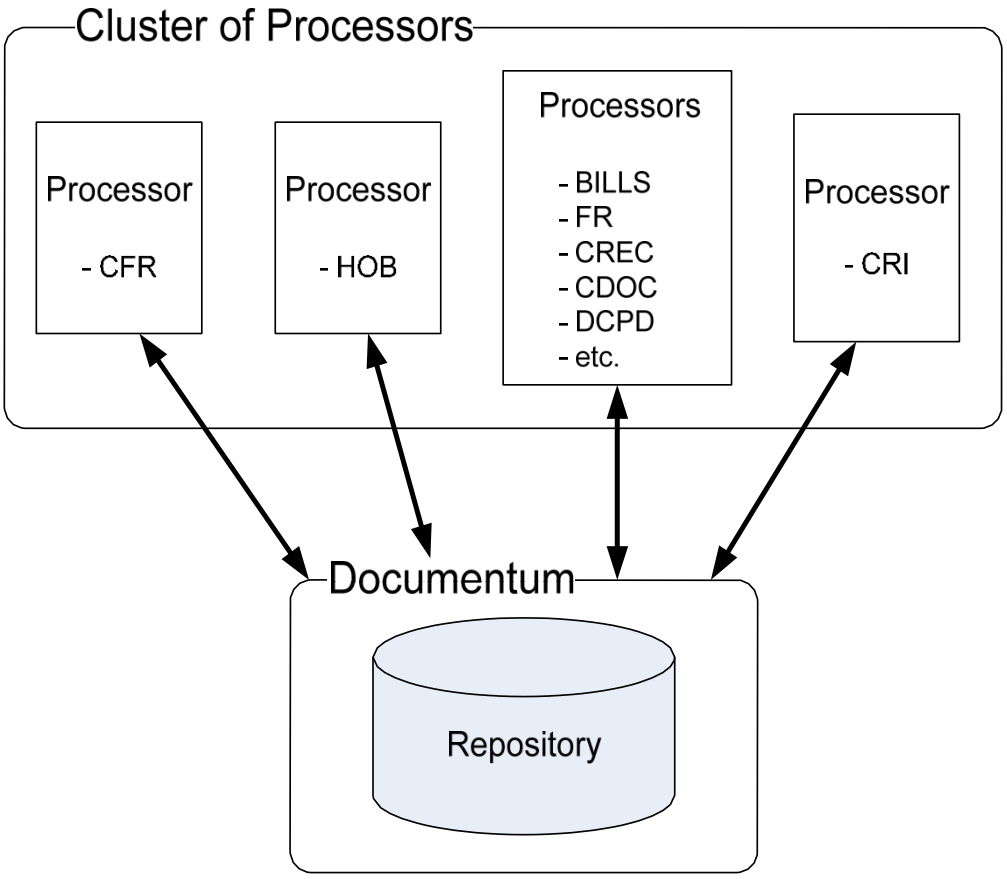

Figure 4.5.6-2 Clustered Processing Architecture

4.5.7 Architecture Update – June 2015

4.5.7.1 SOLR Search Engine

As part of NextGen, FAST was replaced by SOLR including the Publisher, SearchAPI, and Search Engine components. Please see the Search Engine design document, Publisher design document, and Search API design document for more detailed information. Additional information about the search engine replacement plan is available in the SERP – Concept ad Technology Refresh Plan document and the Search Engine Gap Analysis document.

4.5.7.2 Publication Linking

The FDsys publication linking feature provides end users with easy access to the related content of a particular publication on the FDsys public site. The content relationships are derived from the package metadata and can be different for different publications. To ensure fast and consistent access to content relationships, the FDsys publication linking repository will be implemented to persistently store FDsys content relationships, which will be served to the FDsys web application via an FDsys publication linking API.

Semantic Web is a collection of standards and technologies that allow machines to “understand” the meaning of information on the Internet, and enable people to share online content beyond the boundaries of websites. First, relevant information must be stored persistently and published in a standard structure, and then an agent is built to serve such information to the Internet. There are several frameworks to build Semantic Web and linked-data applications. Apache Jena will be used for the FDsys publication linking implementation (a Java framework). Apache Fuseki (SPARQL HTTP protocol) will be used as a Web service layer to service SPARQL query over HTTP. Both software are open source projects, and are well-known in the Semantic Web community.

The building block of Semantic Web is the RDF standard. The goal of Resource Description Framework (RDF) is to describe any resources and their relationships existing in the real world with a predefined unambiguous vocabulary. Applications following the same specification can share the information stored in RDF format and serve them over the Internet or within organizations.

Ontologies are considered another pillar of the Semantic Web. On the Semantic Web, vocabularies define the concepts and relationships (also referred to as “terms”) used to describe and represent an area of interest. Vocabularies are used to classify the terms that can be used in a particular application, characterize possible relationships, and define possible constraints on using those terms. In practice, vocabularies can be very complex (with several thousands of terms) or very simple (describing one or two concepts only).

FDsys plans to extend a few ontologies to describe Federal government publication domain knowledge. Content relationships stored in the RDF store are in a structured format. Initially, such relationships will be used only by FDsys internal applications. It is also possible to share RDF information with external users or organizations if it becomes a requirement in the future.

Please see the Publication Linking design document for more detailed information including architecture and software components.

4.5.7.3 Responsive Search Web Application

The Responsive Search Web Application (RSWAP) is the next generation public facing User Interface. RSAWP is developed to meet GPO requirements for a responsive UI that improves the user experience across form factors from smartphones to traditional desktop monitors and future methods of access and dissemination. An additional objective of RSWAR is to ensure FDsys content is displayed in a way that prioritizes responsive formats where possible.

The Responsive Search Web Application is a multi-tiered public facing Web application. Conceptually the RSWAP could be divided on the following tier: Presentation Tier, Business Tier, Persistence Tier, Content Delivery Tier, Web Statistic Reporting Tier and Content Delivery Network Tier. Figure 3.1 and Figure 3.2 depicts the high level deployment architecture and the layered architecture for the Responsive Search Web Application respectively.

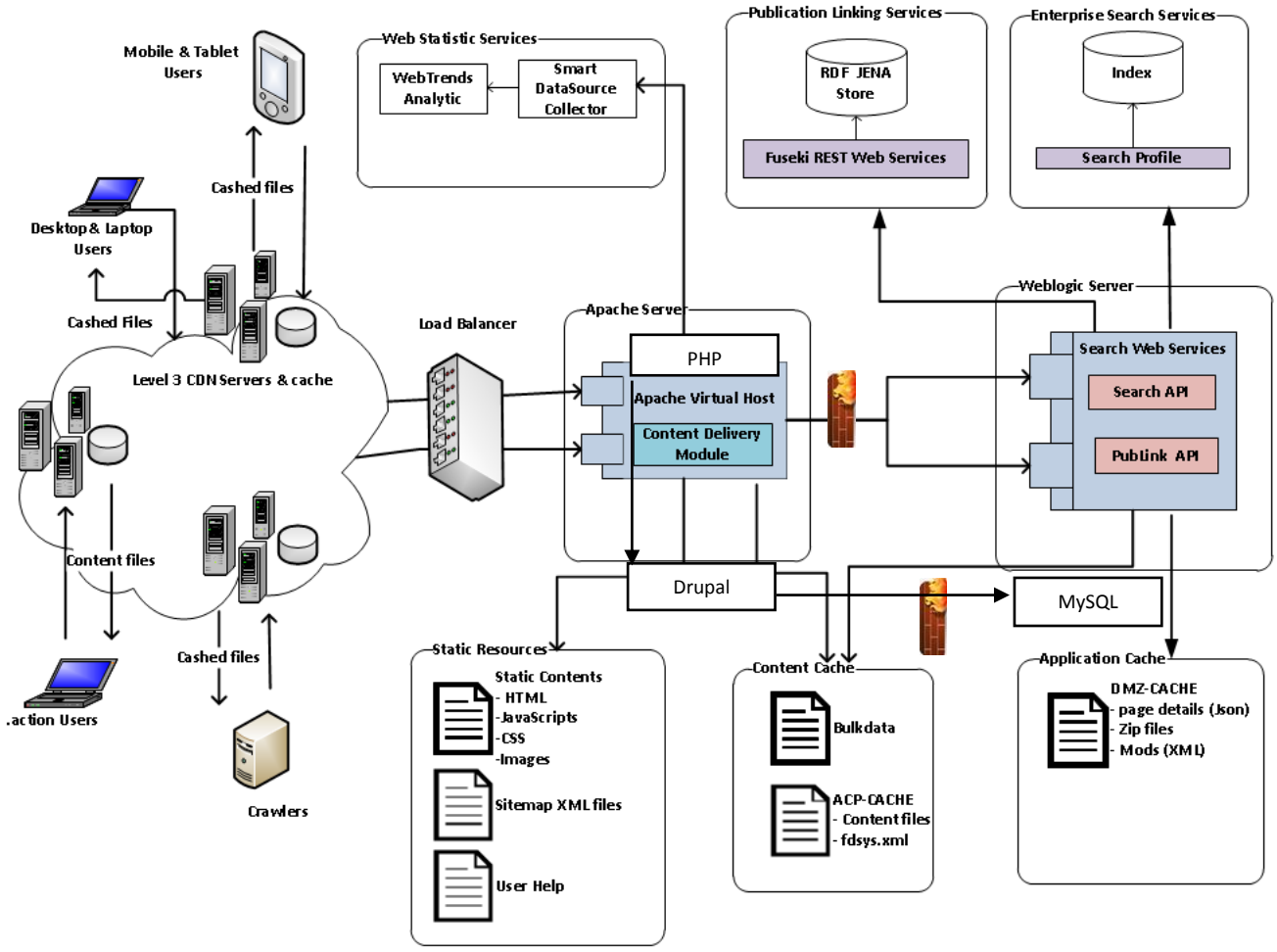

Figure 4.5.7.3-1 RSWAP High Level Deployment Architecture

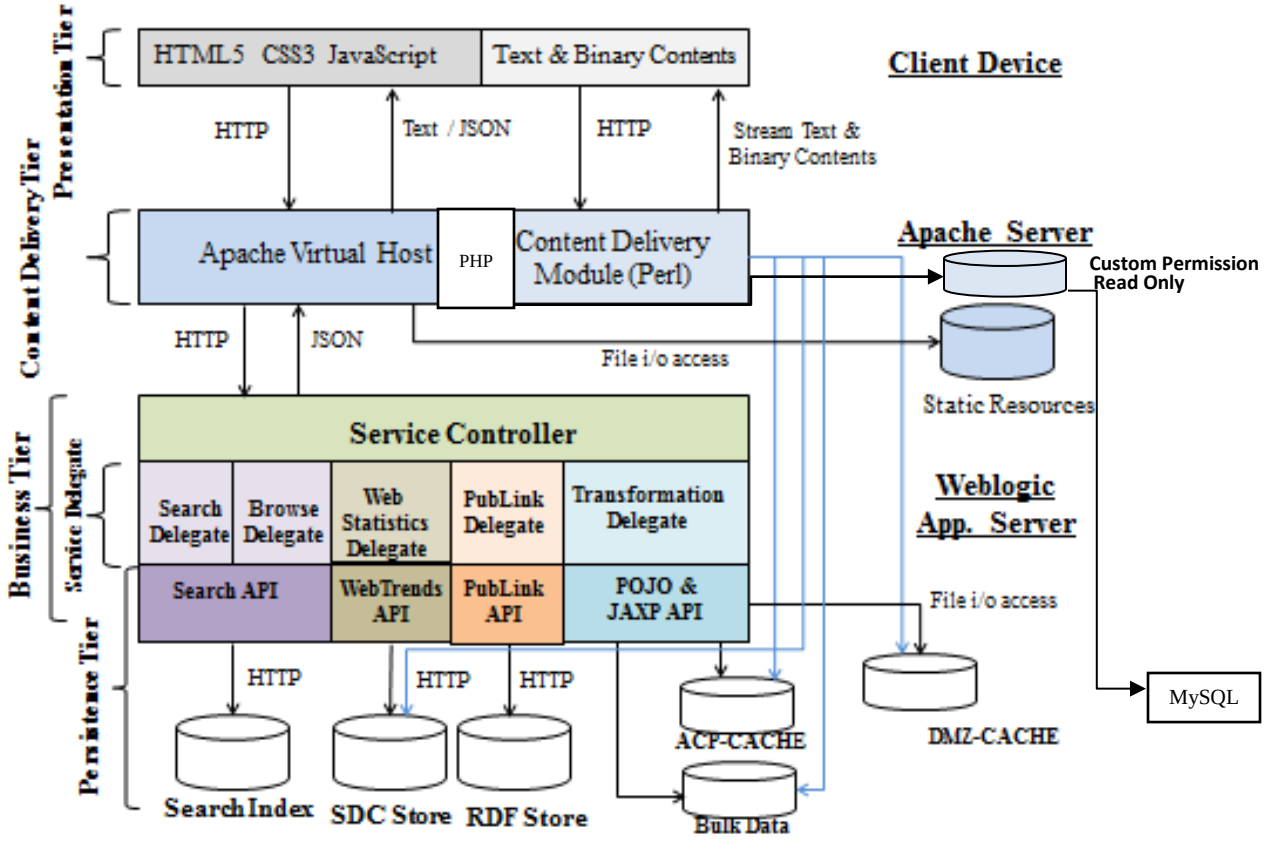

Figure 4.5.7.3-1 RSWAP High Level Layered Architecture

4.5.7.4 Web Content Management System

The Drupal Web Content Management system is used to manage static content in FDsys. Additional information about the design is found in the Responsive Web Application Design Document.

4.5.7.5 Network and Storage Architecture

As part of NextGen, the FDsys network and storage architecture were updated. Please see sections 4.2 and 11 for updated information.

5. Content Packaging

The content packaging is the most fundamental functionality in FDsys to implement the OAIS information package concept. The design of the packaging must be therefore able to fully support the critical missions of FDsys: content authentication, versioning, preservation and permanent access. Following the OAIS reference model, content flows into, managed in and delivered to consumers by FDsys in the form of information packages. The content package therefore provides the most essential foundation upon which the content and metadata evolve through their lifecycle within FDsys. The utilization of the package for managing the content and metadata also distinguishes FDsys from the standard content management systems that usually have little focus on long-term preservation of the content.

5.1 Conceptual Package Model

Although the term of package has been used throughout the document up to this point, it is now formally defined from the data model point of view for the FDsys content.

All content exist in FDsys in the form of a package, which is at the top of the hierarchy of identifiable unit for the FDsys content. A package consists of three elements: content files, metadata about the contents, and a special packaging scheme. The elements are also referred to as digital objects when both the content and metadata files are implied. Figure 5.1-1 shows the conceptual package model for the FDsys content.

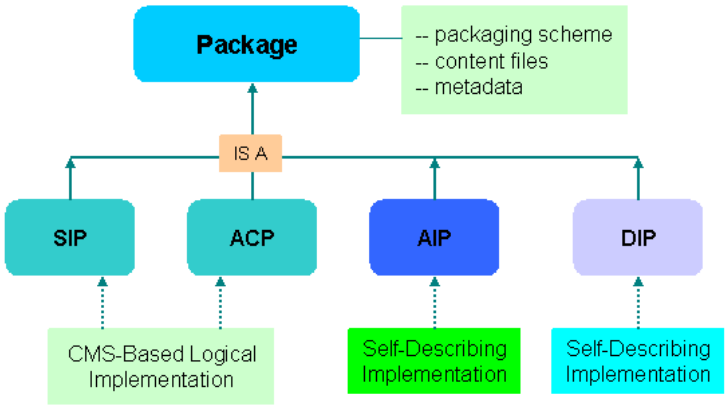

Figure 5.1-1 FDsys Conceptual Package Model

The content files in a package are further categorized into renditions. A rendition is a complete representation of the full publication with a primary file format. For example, a package containing a congressional bill may have two sets (i.e. renditions) of content files, one set in text format and another in PDF format, while both sets have complete content for the bill. A rendition is not entirely defined by the file format of the content files, since a complete representation of a publication may often require files in multiple formats. Examples include HTML rendition (in text) with image files (in jpeg).

As shown in Figure 5.1-1, there are four instances of the package: SIP, ACP, AIP and DIP. They all have the three elements to be qualified as a package, but individual instances of the package may differ by having content files or metadata specific to the relevant package. For example, ACP may have access-oriented renditions that are not in other types of the packages. The content files in a package can be of any types, ranging from plain text file to files in popular formats like PDF and to completely opaque binary data file.

In addition to the package itself that is uniquely identifiable, the digital objects in the package can be uniquely identified as well. The packages or the digital objects within a package can reference each other within FDsys through the unique identifiers. It is worth pointing out that even individual digital objects in a package are identifiable, they are never managed by the underlying CMS independently of its parent package – a distinct difference between FDsys that follows the OAIS reference model and a standard CMS implementation.

5.2 Package Implementation Approaches

As each type of packages serves different purposes, the implementations of the packages vary from one type to another in FDsys. The following sections address how the OAIS information packages are implemented in FDsys.

5.2.1 SIP

The SIP is defined by the OAIS model as follows. The SIP is that package that sent to an OAIS by a Producer. Its form and detailed content are typically negotiated between the Producer and the OAIS. The SIPs are transferred to AIPs by the Ingest functional entity. The OAIS model allows one-to-one, one-to-many, many-to-one, and one-to-none relationships between SIP and AIP to meet various archival needs. FDsys implements the one-to-one relationship between the SIP and AIP. A SIP in FDsys is transferred to one and only one AIP in FDsys by the ingest component.

The content submission to FDsys is from the GPO Plant Operations and other identified Producers such as the Office of the Federal Register. A SIP begins its lifecycle in FDsys when a set of files is received from the Producer. FDsys also facilitates a single or multiple transfer sessions to complete a SIP. Upon submission of a completed SIP, the SIP is ingested to the system and transferred to AIP and ACP in FDsys.

For post R1C2 release, if the agency submission is performed through FDsys user interfaces, such as a browser based web user interface, the current approach for the SIP implementation still applies because the SIP will be created by FDsys on behalf of the content originator. A complete SIP in an agreed-upon package form could be created by agencies for submission to FDsys if desirable in the future. In that case, a technical SIP specification needs to be developed for post R1C2 release.

The SIP is a transitory package, and its lifecycle ends when it is ingested into the system. The design goal for the SIP is thus to maximize the use of CMS capabilities while still logically conform to the conceptual package model. Figure 5.2-1 shows the package structure for the SIP as implemented for content submission from the Plant Operations in R1C2.

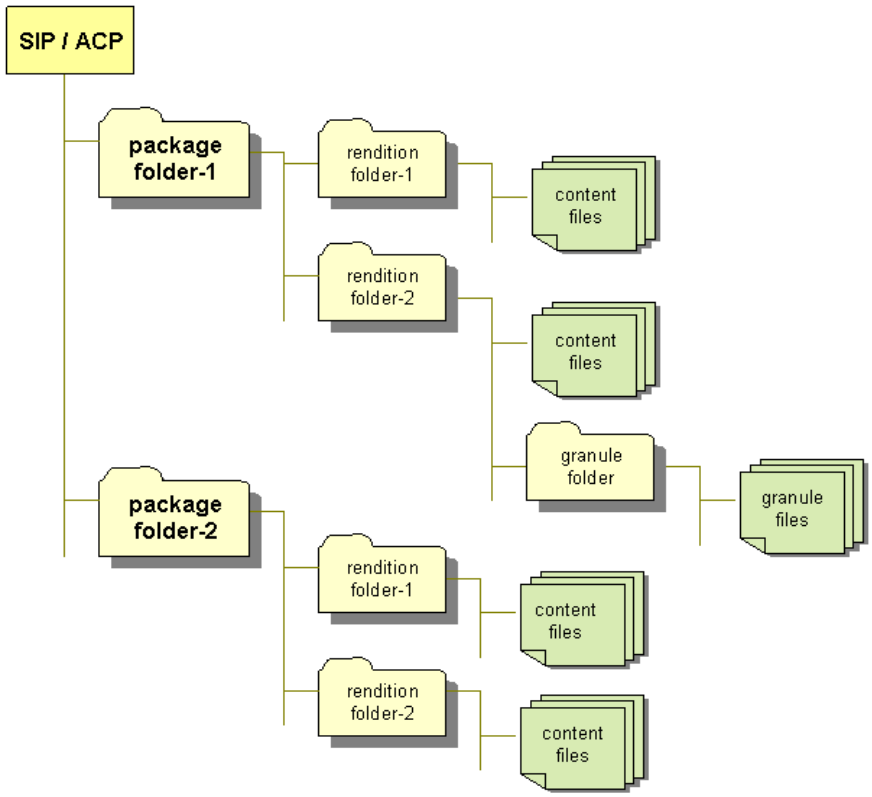

Figure 5.2-1 Package Structure for SIP and ACP

5.2.2 ACP

The ACP is a GPO extension to the OAIS model to facilitate daily content management for the packages and public access to the content. The ACP also provides another layer of protection for the AIP by eliminating the otherwise frequent access to and operations on the AIP, and enabling an installation of a much stronger security control that is completely separate from other operational packages in terms of both user access and content storage.

The implementation for the ACP will take the similar approach for the SIP, maximizing use of CMS capabilities for the content management while conforming logically to the conceptual package model. Figure 5.2-1 applies to ACP implementation as well.

5.2.3 AIP

In the OAIS model, the AIP is defined to provide a concise way of referring to a set of information that has, in principle, all the qualities needed for permanent, or indefinite, Long Term Preservation of a designated information object. The AIP itself is an information object that is a container of other information objects.

To fully meet the FDsys metadata and preservation requirements, the design goals for the archival content packaging are as follows.

- The archival package (AIP) must be fully self-describing to support the long-term preservation mission. The self-describing nature ensures that the package is independently understandable to the designated community, which is one of the mandatory responsibilities for OAIS-compliant archive. Technically the package can be reconstructed through the enclosed self-description by tools that were not used to create and manage the package. Therefore the package will have little dependence on how and where it is preserved. It will also be portable from one archive to another when necessary without requiring that the two archives be hosted in the same technology environment.

- The implementation of the archival package must be independent of the underlying technology that supports the package design. As stated in the concept of operations, the archival packages will outlive the lifecycle of the underlying technologies that host and support preservation processes on the packages. The packages must be able to adapt to technology evolutions with minimal maintenance effort. To achieve this design goal is thus critical to the success of the FDsys implementation.

These design goals challenge, and in fact effectively exclude, the utilization of commercial CMS products as the FDsys archival repository without customization. Every commercial CMS product has its own proprietary approach to associate metadata with the managed content, and thus none of the commercial CMS products would accomplish the design goals in terms of maintaining the self-describing nature of the archival packages, and independence on the underlying supporting technologies.

To achieve the self-describing design goal for the archival package, an XML based packaging scheme is used in FDsys to implement the AIP. With the flexibility and extensibility of XML, the AIP will be portable and be able to survive technology evolutions over time. The METS (Metadata Encoding and Transmission Standard) and its supported extension schemas are explicitly specified in the FDsys requirements for the FDsys implementation of the OAIS reference model. Figure 5.2-2 shows the package structure for the AIP.

In contrast to the SIP and ACP package structure, the package for the AIP contains a set of XML files that make the AIP self-describing and understandable by designated community as specified in the OAIS model. The three XML files are for descriptive metadata for the content files in the package (mods.xml), provenance information about the package (premis.xml), and the packaging XML file (aip.xml) that conforms to the METS schema. Both MODS and PREMIS schemas are supported extensions schemas by METS.

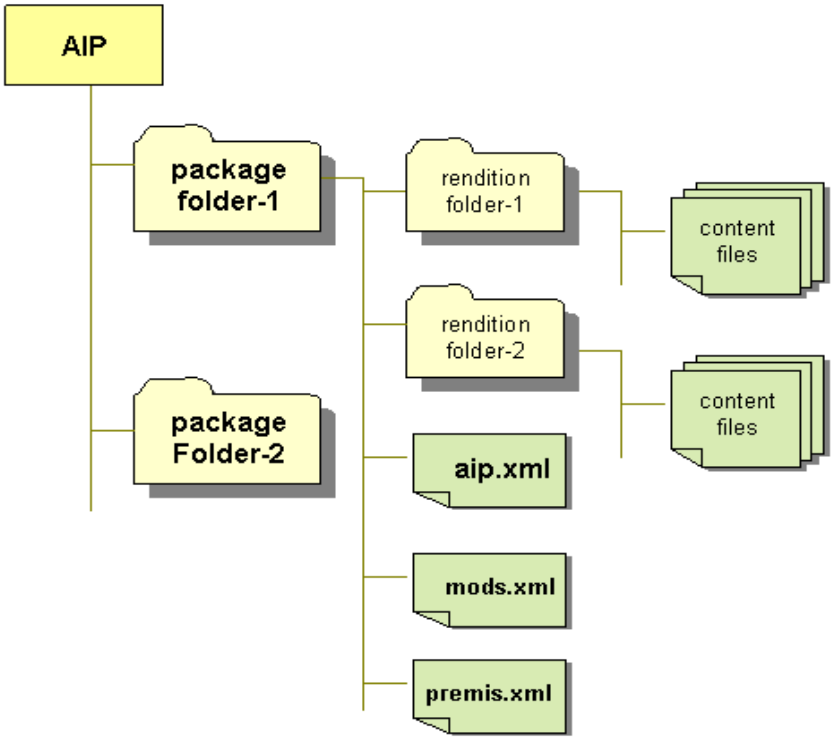

Figure 5.2-2 Package Structure for AIP

The METS file (aip.xml) describes every digital object that belongs to the archival package. In addition, the associations between the metadata files and the content files are also maintained in the METS file.

5.2.4 DIP

Of the four instances of the package, the elements in the DIP vary depending on the request of the user or application that receives the DIP. It could contain just content file(s), or metadata file(s), or both. When both content and metadata files are present, the packaging file will be included as well to describe the association between the metadata and content within the package.

5.3 Package Lifecycle

A package in FDsys begins its lifecycle with the SIP. In FDsys, one SIP maps to one and only one AIP as mentioned before. So the information that is mandatory for preservation must be available or derivable from the SIP before the AIP is created from it. Three scenarios need to be considered for FDsys to collect the required information for a complete SIP.

In the first scenario, the content originators or service providers interact with FDsys through the FDsys user interface to submit content and associated information for a SIP. In this case, the SIP is created and managed by FDsys from the very beginning, and FDsys will store the user inputs, i.e. uploaded content and metadata information, in the form of logical package structure as described earlier. The submission process of FDsys is responsible to accept and process the user inputs in

multiple sessions and maintain the user inputs in the form of the SIP, until the user is ready to submit the SIP to FDsys for ingest. When a user uploads the first content file, a SIP is created with the uploaded file as its content. The user can work on the SIP in separate login sessions to upload more or update (i.e. modify or delete) existing renditions and metadata information in the same SIP. The SIP remains editable until the user submits it, when the SIP validation occurs. This scenario will be implemented in R1C2 for the submission workflow.

The second scenario is a case that involves no user interaction with FDsys to ingest a bulk of content. Since no user input is involved, the information required for a SIP and thus the corresponding AIP must be completely prepared or collected programmatically. The automated bulk input effectively takes over the responsibility of the submission process as in the first scenario with user interactions. It may thus directly communicate with FDsys as if a SIP is ready for ingest. Bypassing the SIP maintenance process could result in significant system performance improvement.

The last scenario is for submission of a complete SIP that is created outside FDsys in the agreed-upon form between GPO and the submission parties. GPO will need to publish a technical SIP specification and any conforming SIPs created at the agency systems or other applications within GPO should be acceptable by FDsys. Upon receiving the submission of a complete SIP, FDsys will validate and parse the SIP for ingest. Figure 5.3-1 shows the lifecycle of the FDsys information packages.

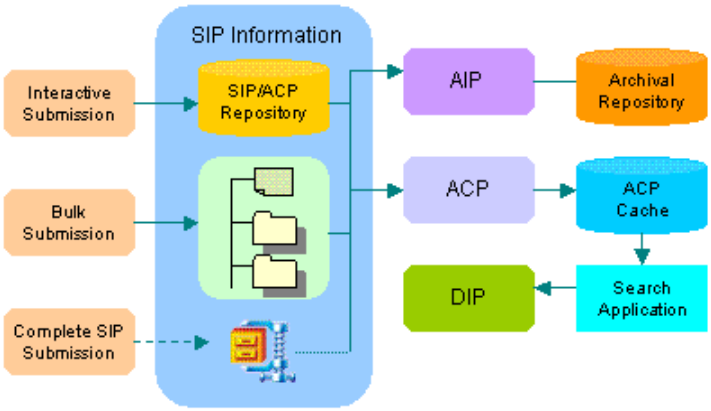

Figure 5.3-1 Information Package Lifecycle

Upon receiving a SIP submission, through one of the three scenarios described above, FDsys validates and transforms the SIP into the ACP, and AIP if it is in scope, respectively by the ingest component. The publicly accessible content is published to the ACP cache storage for public access. A DIP is created from the ACP cache upon request by public users for the content dissemination. The SIP and DIP are transitory packages; SIP lifecycle ends when it is transformed to the AIP and ACP. DIP is a temporary package created solely for content delivery, and FDsys does not retain DIP in storage after delivery. The AIP and ACP are the long-term packages, and will be updated as necessary and maintained in the system to support permanent access to the content.

Though not shown in Figure 5.1.2-1, a DIP can also be created from the ACP for authorized users. To transfer AIP from one archive to another, a DIP can be created from the AIP, in which case the DIP will be the exact copy of the original AIP.

6. Data Model

As a content management and preservation system, the primary data managed in FDsys are the federal government publications. The data model addresses another set of data elements that affect how the publications are managed in the system. In addition to the metadata for the publication content, the data model also concerns with functional attributes that dictate the system behavior such as application security control.

The data model consists of logical and physical models. The logical data model focuses on the data entity definitions and relationships between the entities. How the entities are implemented (i.e. where they are stored and how they are managed) is determined by the corresponding physical data model that is described in the second part of this section.

It is noted that details of the data model vary from one collection to another. The current document is thus limited to providing an overview of the data model that can be seen common to many collections. In addition, the elements of data models for individual collections may differ, the general structures and relationships between entities are the same as described in this document. The detailed data model for each collection is documented in individual Data Management Definition documents (DMDs).

6.1 Logical Data Model

The logical data model for R1C2 concerns the metadata elements that describe the content managed in FDsys, the attributes for content-related information to enable efficient accessibility, the functional roles and content groups that enforce the application security control for FDsys, and the data elements that need to be collected to support the reports.

The logical data model thus has four separate parts: content, security, report, and publication linking models. The content data model describes the metadata and access-oriented attributes for the content packages managed in FDsys. The security model defines the functional roles and content groups to enforce the application security for FDsys. The report model defines the data elements that need to be collected to support the related business reports. The publication linking model defines relationships among FDsys packages as Resource Description Framework (RDF) statements.

6.1.1 Content Data Model

Following the packaging implementation strategy discussed earlier, a complete FDsys package is modeled by three entities with hierarchical relationships: a package, renditions that belong to the package, and granules that belongs to a rendition. Figure 6.1-1 shows a presentation of the ERD (Entity Relationship Diagram) for the content logical data model. The attributes that can have multiple values are marked with the (+) symbol, and the mandatory descriptive metadata with the (*) symbol.

The package entity includes a list of descriptive metadata and a set of attributes to manage the system behavior. It is at the top of the hierarchy with attributes applying to all sub-entities within the package. For example, the package level title should apply to all content files within the package. The rendition entity is to model the various content renditions for the package. For instance, a package may have a list of PDF files as one rendition and a set of text files as another rendition, both containing the same content of the publication but in different formats. In addition to all attributes inherited from its parent package, the rendition entity has its own rendition-specific attributes. The content files within a rendition are modeled by the digital object entity with attributes like file size, and hash value for the content file. A rendition may have granules, which inherit all attributes from the parent rendition and grandparent package. In addition to the file format for the granule files, the granule entity is associated with the granule object entity for the granule files themselves. The granule object captures attributes applying only to the granule content files. For publications that do not have granules, only the top two hierarchies (i.e. package and rendition) apply.

As can be seen from the ERD, the descriptive metadata are all at the package level, while attributes at the rendition and granule levels are mostly for enabling well-structured and efficient accessibility for the content. The attribute inheritance is particularly crucial to facilitating the powerful search capability of the selected search engine for the access subsystem.

In addition to the common attributes included at the top level, the package entity is also associated with a list of publication-specific attributes. Figure 6.1-1 shows an example for the congressional bills. This set of attributes is also access-oriented, and helps enhance the accessibility for the publications. The publication specific attributes may apply to the granule object as well for a subset of publications.

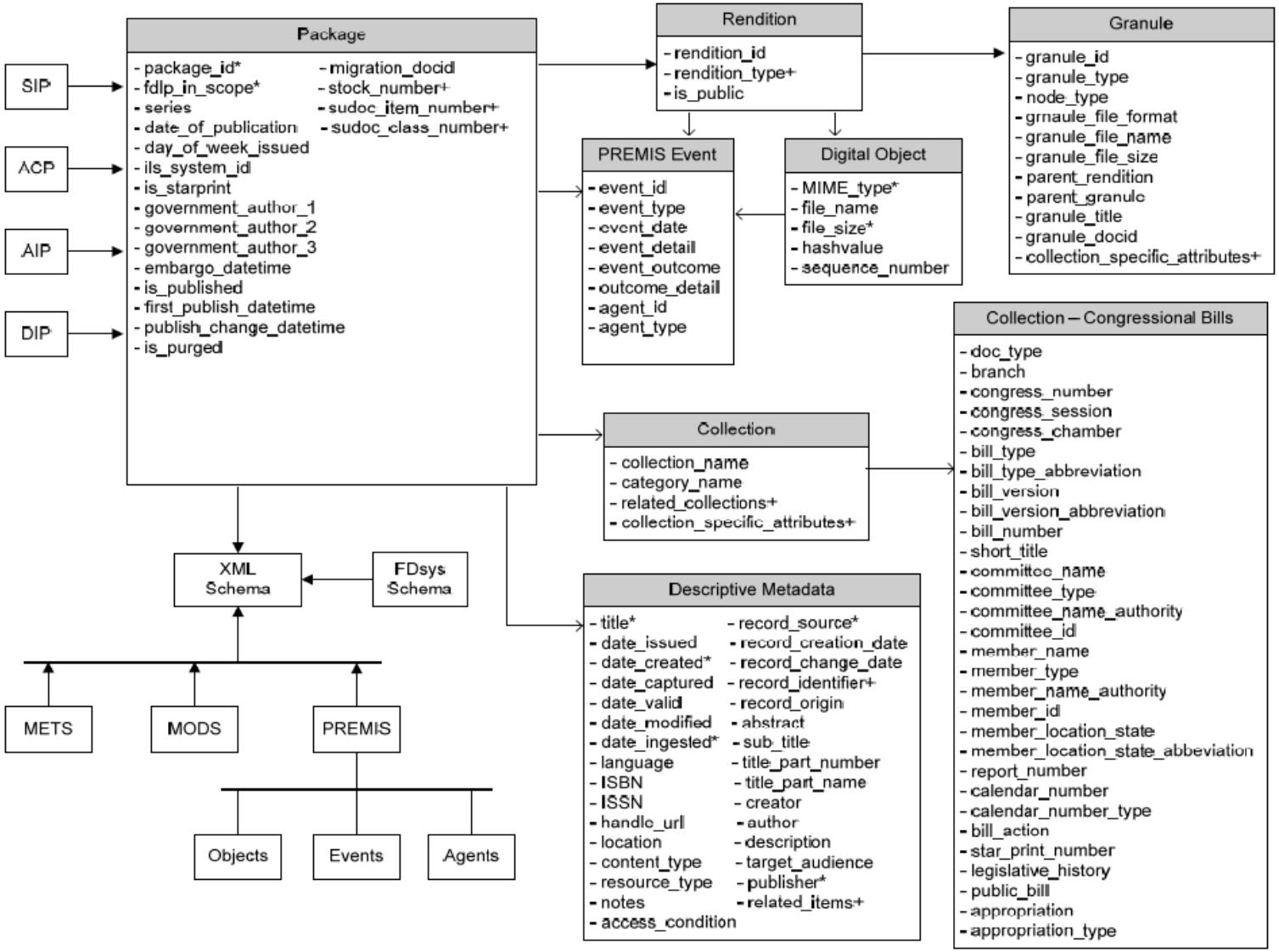

Figure 6.1-1 Content Logical Data Model

Another set of the attributes at the package level is the system attributes to manage the package. For example, the `is_published` is used by the system to synchronize the packages between the content management and access subsystems. The purpose and usage of this group of system attributes will be addressed in individual software components where the attributes are utilized.

Four types of FDsys packages (i.e. SIP, AIP, DIP and ACP) have similar logical characteristics in attributes as shown in Figure 6.1-1, but will be different in physical implementations. A set of metadata schemas for the supported metadata standards is shown in the logical model to indicate the association between the schemas and entity definitions for the content data model – a subset of the attribute values will be populated to the XML schemas to implement the AIP. The XML schema box on the diagram is to indicate a generic XML schema type. The custom schema (FDsys schema) is to accept the parsed metadata from the content parsers, and carry the descriptive metadata, selected provenance information and access-oriented attributes to the access subsystem to support the public access and delivery of the content.

Notice that as the schemas of the supported metadata standards are part of the logical model, the attributes in the ERD are thus limited to a subset of the elements of the schemas whose values must be captured during content submission and ingest processes. In addition, some attribute values apply to multiple metadata standards, because of the elements overlapping between those standards (e.g. MODS and PREMIS have a set of overlapping elements). As far as the metadata standards are concerned, the objective of the model is to capture the data that are required to populate the standard schemas.

6.1.2 Security Data Model

As discussed earlier, FDsys will follow the industry best practices to enforce the role-based application security control. The roles are associated with functionalities performed by the authorized users. For example, a content originator role allows users to upload content to FDsys and to update the content before final submission to FDsys. But the same role may not grant the user privileges to perform operations such as updating ACP, which requires the role of service specialist. The content groups in FDsys can be defined in a way that best meets the business needs. Each package in FDsys must at least belong to one content group; otherwise no user would be able to have access to it. A user can operate only on content that belong to the groups assigned to the user.

The security data model is to define the data entities for the FDsys roles and groups. Figure 6.1-2 is a presentation of the ERD for the security data model. Notice that the User, Organization and Address entities refer to the existing attributes in the GPO Active Directory (AD) server, and only a subset of the User attributes (such as user id, and status of authentication from the AD) will be utilized for authentication purpose for R1C2. Therefore the attribute list may be incomplete but it suffices to serve the presentation purpose of the current logical model.

The authorization of the security model is through the FDsys roles, groups and their assignment to users. The logical model for the application security control is straightforward, consisting of two separate entity definitions, one for the FDsys roles, and another for FDsys groups. FDsys users are assigned with the roles and groups by being on the list of the user role assignment entities. A user can be assigned with multiple FDsys roles and multiple FDsys groups. The effective roles for the relevant group will be the union of all roles the users are assigned to. A user will be assigned with a

default FDsys group. Upon successful login onto the system, a user with multiple roles will be able to perform functions to content that belong to the groups that the user is assigned with. In other words, all roles assigned to the user for different groups are enabled transparently by the system.

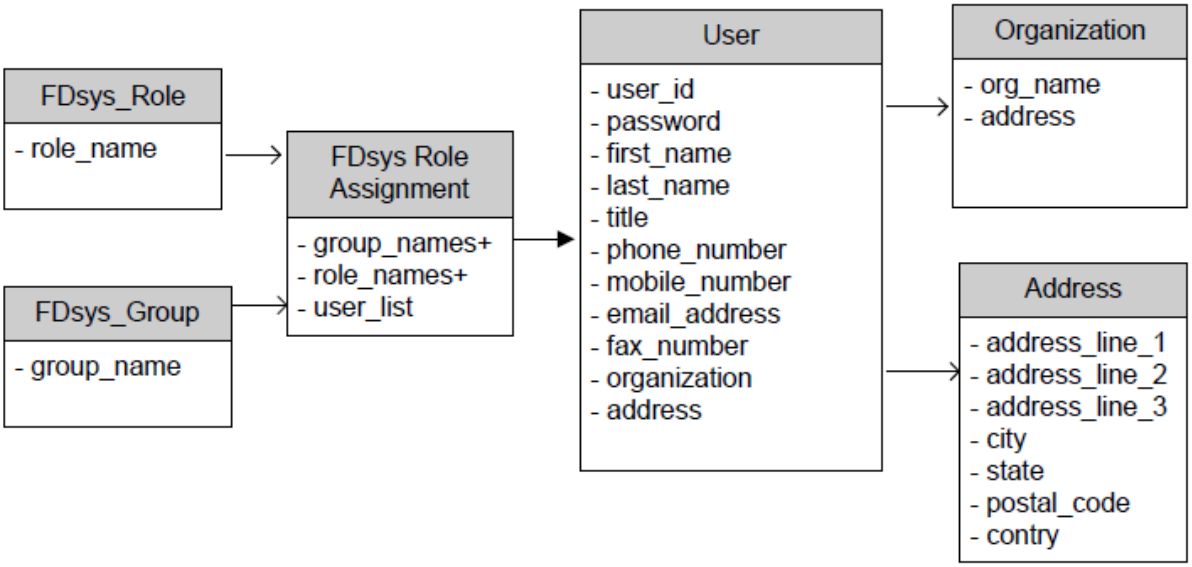

Figure 6.1-2 Security Logical Data Model

Though not explicitly addressed in the logical model, the implementation of the security model can be accomplished in such a way that the user roles are tied to groups. In other words, a user can effectively have different roles depending on the groups the use belongs to. One approach is to define the groups with access permissions attached, for example group A with delete privilege and group B with read-only privilege. A user belonging to group B with submission role will be unable to submit content to group A, which requires delete privilege that the user does not have. The physical implementation of the logical model describes in detailed how the dependence of the roles on groups is handled.

6.1.3 Report Data Model

GPO currently produces a number of content related business reports, such as the number of titles of the selected federal government publications that are made available online, and numerous web access statistics for the content. It is often the case that a current report requires data from various sources from different business units within GPO. Analysis of the current reports indicated that while FDsys will be able to provide partially the required data for a subset of the reports, FDsys would not have complete dataset for any of the current reports. As such, FDsys will collect the available data that are part of the existing reports, and deliver the data to the respective report authors who would combine the data from FDsys and other sources to create the complete reports.

The report data model thus defines the report entities for the data attributes to be collected and delivered to the business users by FDsys. Figure 6.1-3 shows a presentation of the ERD for the report logical data model.

At a high level, the reports are divided into two categories. One category is for the web statistics for online access of the content. The reports generated from the WebTrends will continuously meet this type of reporting needs. Another category is the data that are available from FDsys during content submission and processing phases. These data need to be collected and delivered to support some of the existing reports. Note: WebTrends is planned to be replaced after the initial launch of NextGen and will be documented in the Custom Application design document.

The report logical model shown in Figure 6.1-3 defines two report types; one is for the metrics of acquired titles of the collections on the monthly and annual basis. Another is for the online availability report for the congressional bills and other collections. This report type is further categorized into sub-reports based on organization (House or Senate) and reporting frequency (daily and weekly). In Figure 6.1-3, a child report type will inherit all data attributes of its parent type.

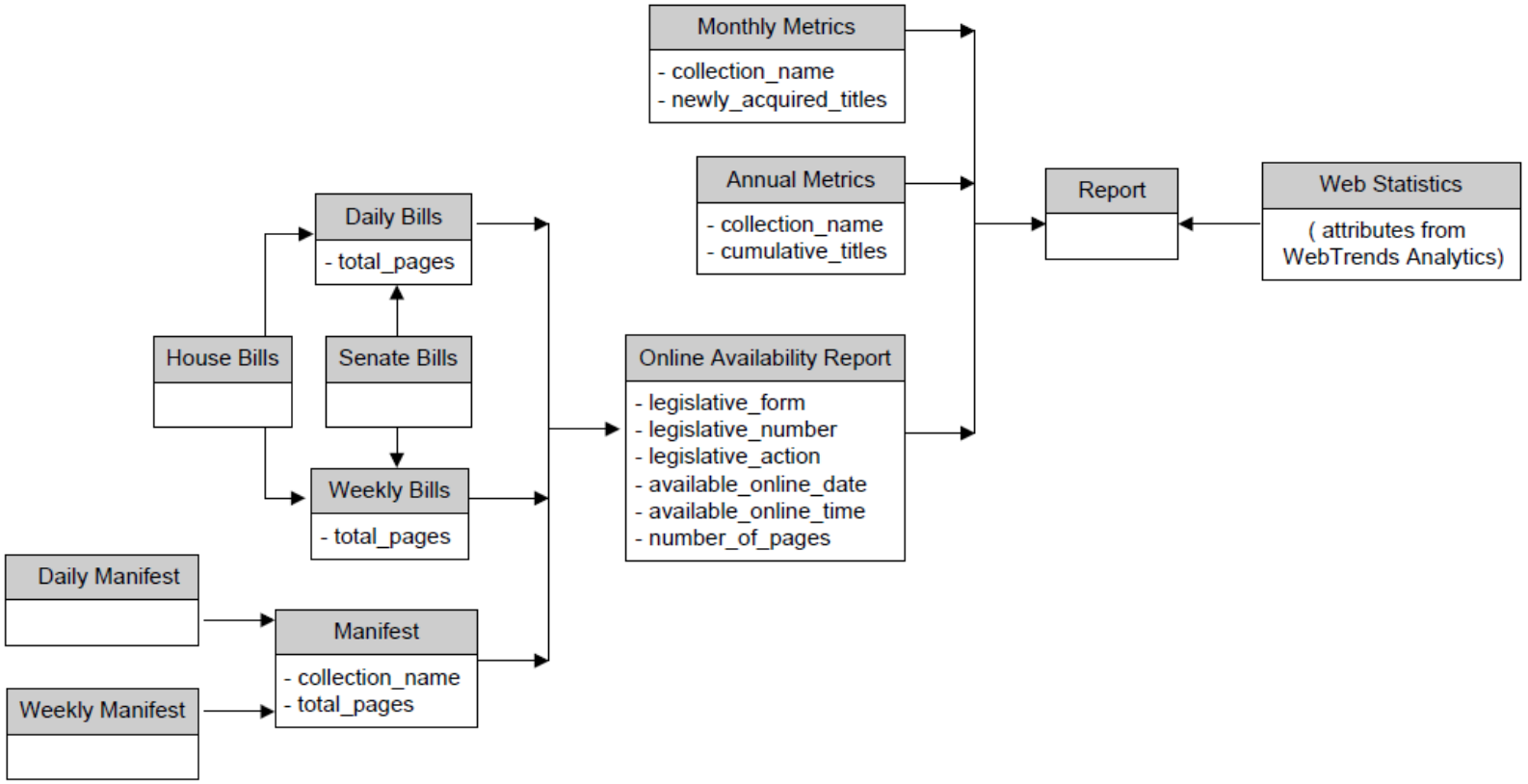

Figure 6.1-3 Report Logical Data Model

6.1.4 Publication Linking Data Model

The FDsys publication linking feature provides end users with easy access to the related content of a particular publication on the FDsys public site. Relationships are derived from descriptive package metadata and can be different for different publications. A hierarchical business object model is used to represent FDsys package, granule, and standard reference structures and relationships. Since the package and granule level attributes are easily accessible from fdsys.xml when an access ID is given, only minimum common attributes are stored in the package and granule level business model. The data model is further documented in the Publication Linking Design Document and specifically in the Relationship Management Definition (RMD) Appendix. The RMD defines technical specifications of all resources and relationships implemented in FDsys Publication Linking repository. As described in section 2.2 of the Publication Linking Design Document, FDsys publication linking repository stores both logic business entities and FDsys physical entities. Logic business entities are presented by standard references in FDsys, and physical entities are represented by FDsys accessId.

The package and granule business objects are collection neutral. The majority of relationships between FDsys objects are derived from standard references. Each standard reference XML element is translated to an individual Standard Reference Business Object. The Standard Reference Business Object Classes share common interface but is implemented specifically for individual collection. Only essential attributes necessary to uniquely identify a standard reference are modeled into the business object.

6.2 Physical Data Model

This section describes how the logical data models are implemented, in terms of the storage and management approach.

6.2.1 Content Data Model Implementation

The physical content data model implementation varies depending on the types of FDsys packages, as described in section 5.2. In addition to the implementations for the FDsys packages, a separate implementation is developed to specifically support the access subsystem. The physical implementations for the packages are accomplished in two ways: through customized Documentum object models for the SIP and ACP and by the XML files conforming to the schemas of the metadata standards for the AIP. No custom database tables are necessary for the physical implementation of the content logical data model.

SIP and ACP Implementation

As discussed earlier, the native content management capabilities of Documentum will be fully leveraged for the SIP and ACP implementations at the content management subsystem. The physical data model implementation for the SIP and ACP will therefore be to store all attributes in Documentum in the form of the custom Documentum object models. The attributes are thus managed automatically by the Documentum metadata management. The Documentum object models are created in such a way that they implement the logical structure of the package specification. For detailed Documentum objects models for the SIP and ACP, refer to the Repository Design document.

AIP Implementation

To implement the completely self-describing AIP for the archival repository, a subset of the attributes from the content logical data model are mapped and populated to the elements of the supported metadata standards, and the resulting metadata files in XML are stored as part of the AIP conforming to the package specification. Specifically, the descriptive metadata are stored in MODS, provenance metadata in PREMIS, and technical metadata to NISO technical metadata schema as implemented in PREMIS.

For AIP, descriptive metadata are also stored in Documentum for content management purpose – package identification and retrieval for example. But the metadata in the XML files are considered authoritative, and metadata in Documentum are considered cached values instead. The custom Documentum objects for the API and the packaging application that ensures the self-describing characteristics of the AIP are addressed in detail in the Repository Design document.

DIP Implementation

The DIP implementation depends on the request of the content delivery. It can be simply a list of content files, if no metadata delivery is requested. Upon request, the metadata will be delivered in XML forms. This is accomplished by mapping and populating the relevant attributes from the logical data model to the metadata standards schemas, and deliver the resulting XML files as part of the DIP.

FDsys Schema

The FDsys schema in the logical model represents the custom schema defined by FDsys to accept metadata extracted by the content parsers. It also defines elements for the publication granule information. As the most metadata in R1C2 is extracted by the content parsers, the data values in this schema thus become the primary source for all subsequent physical data model implementations – from the CMS object models to FDsys packages and to the search application (see below). In addition to the content parsers, the data source for this schema includes system generated values during ingest process, and also values entered manually depending on the publication types.

As indicated in the logical model, the FDsys schema will cover the data attributes that are common to all publication types as well as publication-specific ones. For the SIP and ACP implementation, the values from the FDsys schema will be populated to the Documentum object models, while the XML file in the FDsys schema will be stored to the SIP and ACP as the metadata file for packages.

The FDsys XML file is a completely internal working mechanism to facilitate communications between FDsys applications, from the content repository to the search engine and custom web application. It is populated, updated, and consumed at various stages in the content lifecycle within FDsys. The Data Management document for each collection or publication type will have mappings to the metadata elements defined in the FDsys schema.

6.2.2 Security Data Model Implementation – LDAP integration

As discussed in the application security, the LDAP-compliant directory server will be used to store the roles and groups information for FDsys to enforce the access control over the content as well as functions performed by authorized users. The Documentum repository will be configured to integrate with the LDAP directory servers using the Documentum built-in support for the LDAP integration. The FDsys roles and groups, as well as the details of the LDAP integration are addressed in the Repository Design document.

6.2.3 Report Data Model Implementation

The report data are collected in two ways: through the web analytics tool (WebTrends) and custom FDsys applications. Since WebTrends provides tools to configure and create the custom reports, the implementation will focus on the report data collected by FDsys applications.

The relevant reporting data need to be collected at different stages of the content lifecycle within FDsys. For example, the acquired titles in the logical model (i.e. publications in the form of the packages) are available during ingest stage, while the number of publications that actually become available online will not be collectable until all content processing completes and the package is published to the access subsystems. Details of the physical implementation of the report data model are also addressed in the Repository Design document.

6.2.3 Publication Linking Data Model Implementation

Relationships among FDsys packages are stored as Resource Description Framework (RDF) statements in the FDsys publication linking repository, and each statement represents a single fact of relationship between FDsys resources. RDF is a family of W3C specifications originally designed as a metadata data model. RDF has come to be used as a general method for conceptual description or modeling of information that is implemented in web resources, using a variety of syntax notations and data serialization formats. RDF uses an abstract model to

decompose information into small pieces with some simple rules about the semantics of each one of these pieces. This abstract model includes a statement, subject and object resources, and a predicate. Details of the implementation are in the Publication Linking Design document.

7. OAIS Functional Model Implementation

In addition to various common services that are generally available from the modern IT environment (e.g. security infrastructure, storage allocation and directory services), the OAIS model specifies a set of functional entities for a compliant archive system. Figure 7-1 shows the functional entities from the OAIS model.

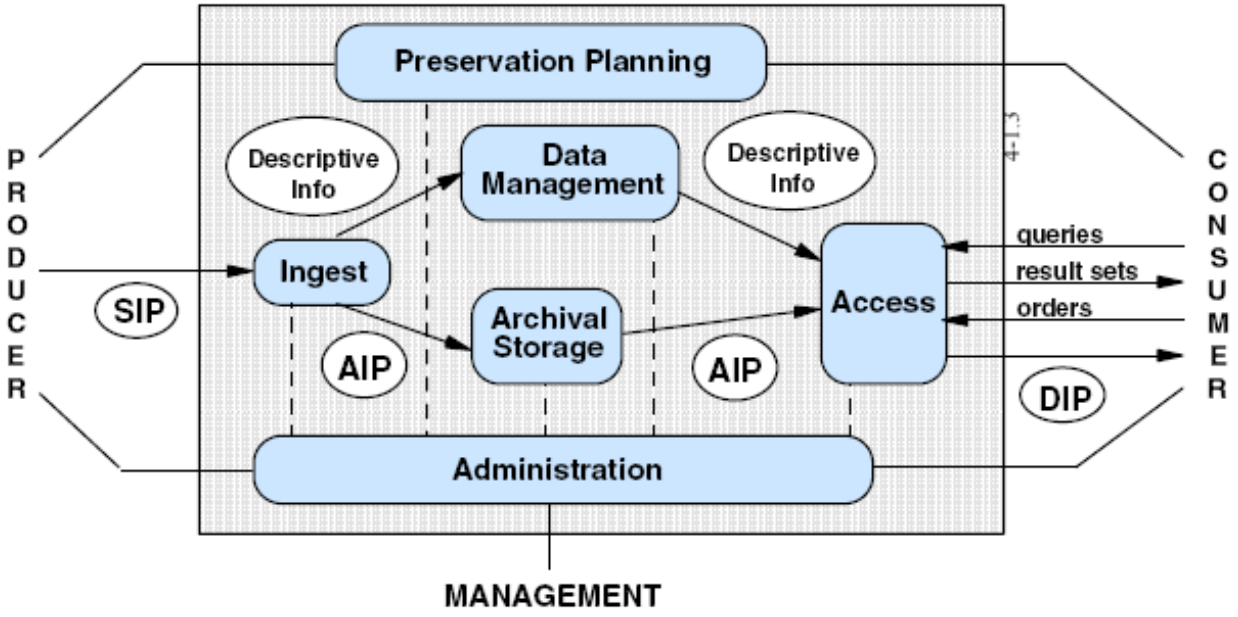

Figure 7-1 OAIS Functional Entities

The roles provided by the functional entities are as follows.

- Ingest – to provide the services and functions to accept SIPs from Producers and prepare the contents for storage and management within the archive.
- Archival Storage – to provide the services and functions for the storage, maintenance and retrieval of AIPs.
- Data Management – to provide the services and functions for populating, maintaining, and accessing both descriptive information that identifies and documents archive holdings and administrative data used to manage the archive.
- Administration – to provide the services and functions for the overall operation of the archive system.
- Preservation Planning – to provide the services and functions for monitoring the environment of the OAIS and providing recommendations to ensure that information stored

in the OAIS remains accessible to the designated user community over the long term, even if the original computing environment becomes obsolete.

- Access – to provide the services and functions that support consumers in determining the existence, description, location and availability of information stored in the OAIS, and allowing consumers to request and receive the information products.

Many of the functional elements are available from COTS CMS products, though often with different terminologies. This section attempts a mapping of the FDsys content management and archival functions to the OAIS functional model.

As can be seen from Figure 5.3-1, the OAIS functional model begins after the acceptance of the SIP by the system, the SIP submission in FDsys is thus covered first in the section below.

7.1 Content Submission

The R1C2 release supports three content submission scenarios: interactive and folder-based approaches for the day-forward submission, and bulk submission for GPO Access data migration.

The submission process in FDsys is responsible for accepting uploaded content files and optional user input for metadata information if applicable. It also invokes FDsys custom application tool – the content source module to arrange the content files into the rendition folders which conform to the package structure described earlier. The content source module performs its task of arranging the files into the renditions by following the pre-defined packaging rules for the relevant types of publications. While the content source module tries its best based on the rules, user manual intervention is supported to correct or override the outcome of the content source module in case of exceptions that were not covered by the rules. The details of the content source module are described in the Custom Application (SDD Vol. VI) design document.

For R1C2, FDsys supports multi-session user interaction with the system to upload, verify, modify and finally complete the SIP within a work-in-progress (WIP) area before submission. Note that the WIP is not a package. The WIP is an FDsys implementation to assist the authorized users to prepare the SIP for submission, and should not be confused with the SIP itself. The formal SIP in FDsys is defined as the package that is submitted by the authorized user for ingest. In the actual implementation for R1C2, the system creates an incomplete SIP object upon receiving the uploaded files from the user, who continues to prepare the (incomplete) SIP until ready for submission. So the "Create SIP" action in the illustrative diagrams in this subsection refers to the initial object creation for the incomplete SIP, which will be hosted in the WIP area until submission. Note: For BILLSUM, BILLSTATUS, and USCOURTS automatic submission is from an external data source.

Note that all submission scenarios end with the start of the Ingest process. The validation of the SIP is performed as the first step of the Ingest functions. The validation at this stage is to ensure:

- The required digital objects are available in the package according to the packaging rules.
- The metadata that need to be collected during submission have valid values.
- The SIP is not a duplicate of any package in the system.

7.1.1 Interactive Submission

This is an internal FDsys process interacting with browser-based user interfaces to support the interactive submission. The main objective of the process is to accept user uploaded content files and metadata information. Figure 7.1-1 shows the interactive submission process.

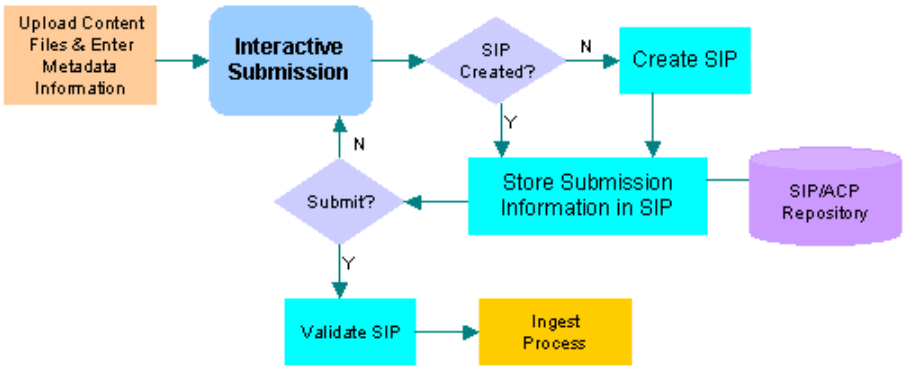

Figure 7.1-1 Interactive Submission

When a first content file or piece of metadata information is received, the submission process creates a CMS object to represent the SIP in the repository. Starting from its creation, the SIP stays in a constantly evolving state as the user continues to provide the submission information. The submission process ends when the user submits the SIP for ingest. The SIP information will no longer be available to users for update as a SIP after the submission.

The SIP is validated before the ingest process. If the validation fails, the user will be notified and the persistence for the SIP remains available in the WIP area for the user to perform correction and resubmission.

For R1C2, the interactive submission workflow uses this submission process to accept submission from the GPO Plant Operations. The submission workflow is addressed in the Repository Design document.

7.1.2 Folder-Based Submission

To facilitate a smooth transition from WAIS submission to FDsys submission for the day-forward content from the GPO Plant Operations, a special folder-based submission is supported for a set of selected collections.

In the folder-based submission, a hot folder is setup for the content files to be ingested to FDsys. The folder will have a pre-defined structure for the collections with file names conforming to the pre-defined name conventions. The folder is monitored by FDsys and the files are moved to the CMS repository and stored in the package form of the SIP, as if the files were uploaded by an authorized user through the interactive submission.

When an authorized user logins to FDsys via the same user interface for the interactive submission, SIPs (one for each package), that were pre-populated, by using the pre-defined packaging rules, with the content files that FDsys collected from the folder, will be available for manual check for misplaced or missing files, etc. The user can then submit the SIP for ingest. Figure 7.1-2 illustrates the folder-based submission.

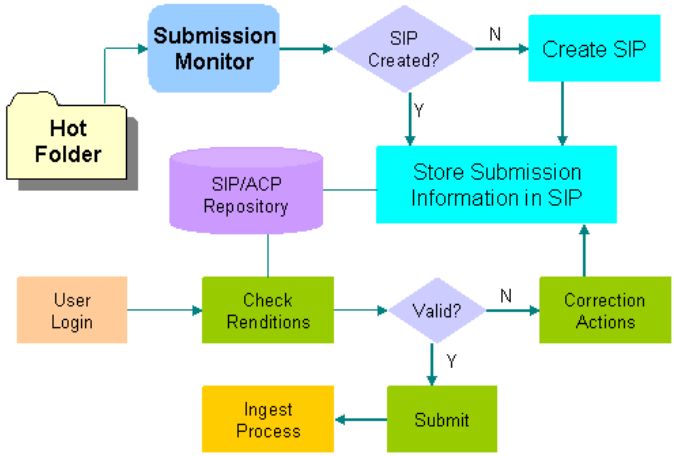

Figure 7.1-2 Folder-Based Submission

The difference between the folder-based and interactive submissions is only in how the content files are uploaded to FDsys. The main objective of the folder-based submission is to relieve user's burden of having to manually upload significant number of files (about 100 or more) to FDsys on the daily basis.

It is recognized that the folder-based submission is associated with an impact on the system architecture. For this approach to work, the hot folder must be accessible to both the users who place the files to the folder and the FDsys content server. Therefore FDsys must be deployed local to where the hot folder is, in terms of storage network. Folder based submission has been extended to include an automated submission job for select content including but not limited to the United States Courts Opinions collection. It is documented in the Repository design document.

7.1.3 Bulk Submission

The bulk submission process has the same objective as its interactive counterpart, but it interacts with automated bulk submission tool instead of users. It receives the metadata information and content files from the tool that in turn employs the content source module to gather and provide the required content source information to the submission process. Figure 7.1-3 shows the bulk submission process.

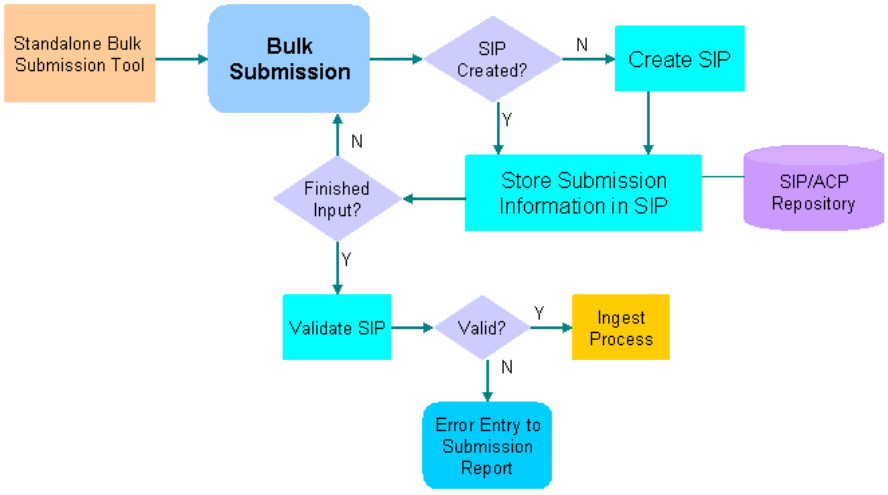

Figure 7.1-3 Bulk Submission

Like in the interactive case, the bulk submission process creates a SIP upon receiving the first input from the submission tool, and continues updating the SIP until all input information is processed and added to the SIP. Before the process flow control is passed over to the Ingest process, the SIP is validated. If the validation fails, an error entry is entered to the submission report that is created and updated constantly by the submission tool.

The bulk submission is aimed at automated submission scenarios with large volume of content files. For R1C2, the GPO Access migration process flow will be based on this design. Similarly, batch submission of files is also enabled through the GOVPUB collection.

7.2 Ingest

The Ingest functional entity in the OAIS model starts with receiving the SIP from the Producer. In FDsys, this corresponds to submission of the SIP to the system as illustrated as Ingest process in the figures of section 7.1. Figure 7.2-1 shows the functions of the Ingest as defined in the OAIS model.

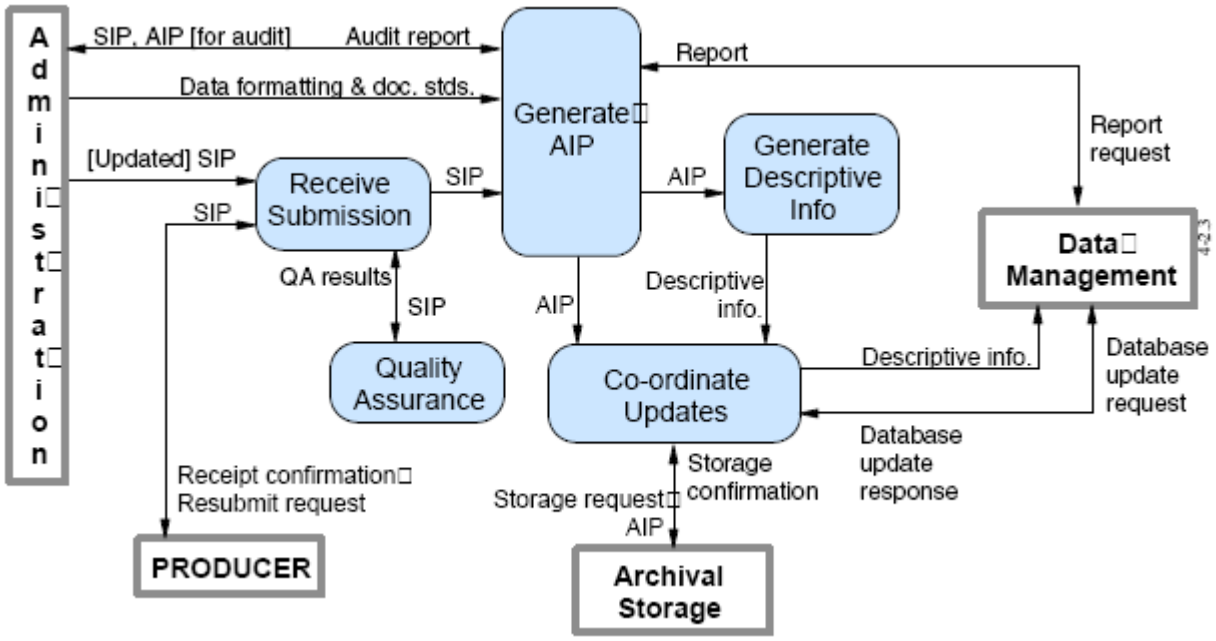

Figure 7.2-1 Functions of Ingest in OAIS

In FDsys, the **Receive Submission** function is performed by the system when the user submits the SIP or the submission job is enabled on behalf of the user, as described in section 7.1. For R1C2 submission, FDsys supports multi-session user interactions with the system to complete the SIP before submission. For this purpose, FDsys provides a work in progress (WIP) area for users to upload and validate content files. The system also validates the files in the package to ensure that they belong to the right rendition according to the pre-defined packaging rules. A globally unique (across SIP and AIP storages) identifier is assigned to all digital objects when they are received (or generated) by the system. The system performs virus check whenever a new content is uploaded. The checksum of a content file is calculated upon successful upload of the file to the system.

The first task of the FDsys ingest is to perform a (final) validation of the SIP, corresponding to the **Quality Assurance** function in the OAIS model. If found invalid according the pre-defined rules, the SIP is rejected for ingest and a notification is sent to a designated authorized user for actions, after which the SIP can be resubmitted. Otherwise the ingest process continues. Figure 7.2-2 shows the model of the FDsys Ingest process.

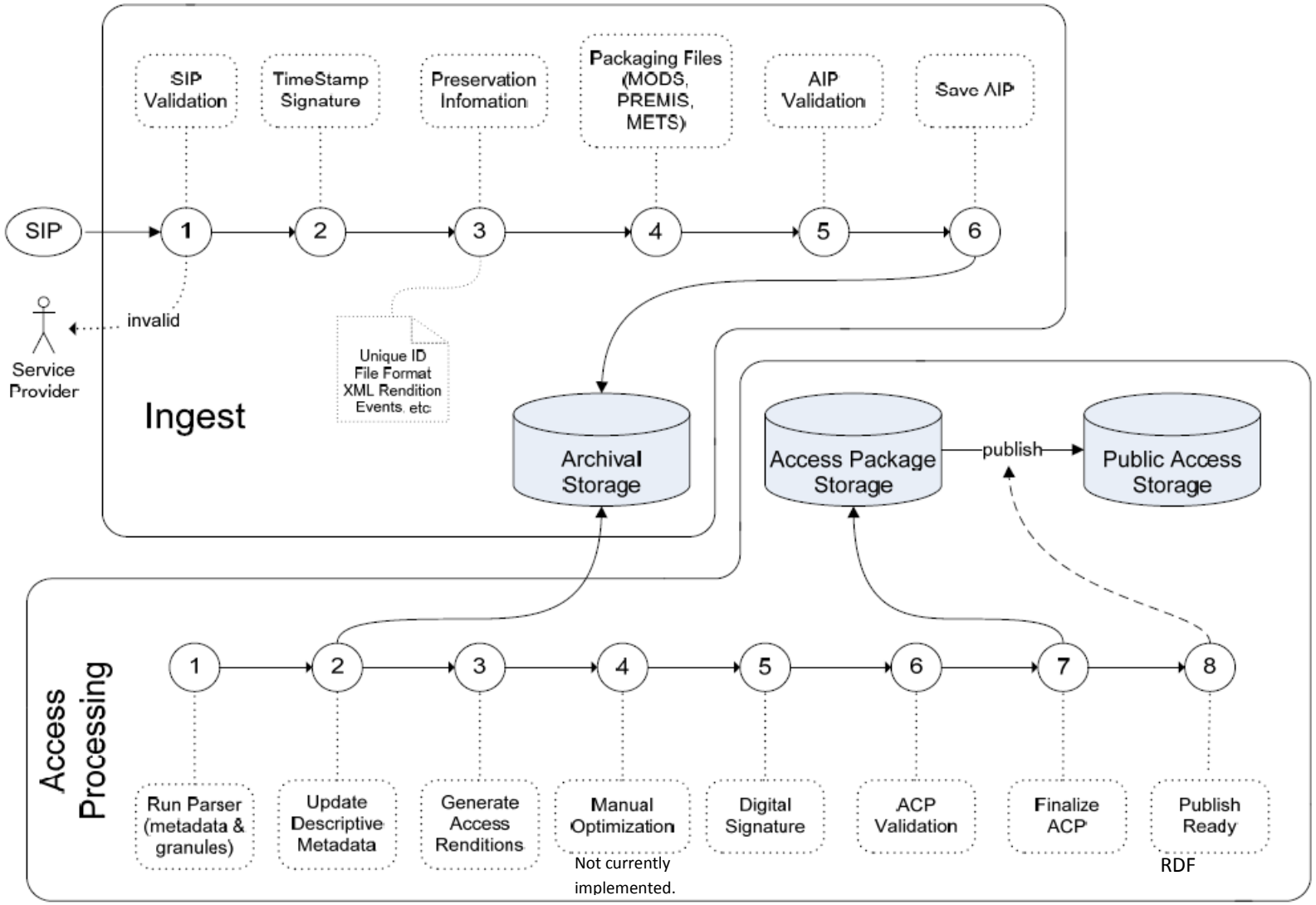

Figure 7.2-2 FDsys Ingest Process Model

The function of **Generate AIP** is implemented in FDsys in the ingest process in which the AIP is created from the SIP. The workflow is triggered by submission and successful validation of the SIP. A packaging utility is developed in FDsys for creating the AIP. The objectives of the utility are to transform the descriptive and provenance information from the SIP into XML files for the AIP. The XML files conform to the supported metadata standards in FDsys (i.e. MODS and PREMIS in R1C2). The utility also creates a binding XML file, conforming to METS schema, to hold all information about the AIP (content as well as metadata files), making the AIP fully self-describing.

It is noted that for R1C2, a set of descriptive metadata necessary for preservation and access purposes is not entered by the submitters, but captured by FDsys through the custom parsers. The AIP thus created will contain only a limited number of descriptive and technical metadata (e.g. unique ID, hash values for the content files, timestamp for the AIP, and file format). The PREMIS events will include: creation of the AIP from the SIP, submitter of the SIP for the package.

Upon successful creation and package validation, the AIP is written to the archival repository, which is completely separate from where the SIP was staged. The AIP storage information is stored both in Documentum, and the METS file as well (to make the AIP self-describing). These operations correspond to the **Coordinate Updates** function of the Ingest functional entity of the OAIS.

As discussed earlier and shown in Figure 7.2-2, FDsys extends the OAIS model by adding the ACP which is primarily for daily content management purpose and also provides a protection layer to its corresponding AIP in that operations on content files (such as extracting metadata) can be applied to the ACP instead of the AIP directly.

The ingest process ends with completion of a successful AIP creation, and storing the AIP to the archival storage. The ingest process is then followed by a separate workflow to process the package for the ACP – the access processing in Figure 7.2-2.

After creation of the ACP, the first task the processing workflow performs is to extract the descriptive metadata and generate granules files if applicable. The custom parsers are invoked for this purpose. The AIP will be updated with the extracted descriptive metadata (specifically the MODS file). These operations also correspond to the **Generate Descriptive Information** function in the OAIS Ingest functional model (see Figure 7.2-1). The parsing operations could be performed on the content files directly from the AIP, if there was no the ACP extension to the OAIS model in FDsys. So the ACP plays the protecting role for the original content files in the AIP.

Besides the descriptive metadata and original content files from the SIP, additional digital objects may be required in the ACP to fully support the public access to the content. For example a screen optimized PDF version of the original content, if applicable, may need to be added to the ACP if not provided as part of the SIP by the content originator or service provider. The additions to the ACP may or may not be stored to the corresponding AIP, depending on the purpose of the addition and its value for preservation. The digital objects that are required for the ACP are as follows.

- Access renditions of the content. The access renditions are the same content as the original publication but in different file formats. The examples include screen optimized PDF version, html version, or plain text version of the original content.

- Access-oriented metadata. For each type of publications, a set of metadata information will be extracted from the content to enhance the public accessibility to the content.

- Publication granules. To enable the granule level access to selected publications, the ACP will have to contain the granule information before it is published to the search application. While the search engine can handle the granules stored in logical form (such as tagged in an XML file), the logical granule search is not supported by the CMS for the internal authorized users. To consistently support the granule search capabilities for both the access and content management subsystems, it was decided to physically split the rendition files into individual granule files for both the text and PDF renditions. The individual granule files will be handled in both subsystems as independent, searchable content files.

- Access relationships. The relationships include how the granules are related to each other, and links or citations between publications, at the publication or granule level.

- Digital signature for the publicly accessible PDF files for select collections.

The required digital objects for the ACP may be included in the SIP by the content originator, the GPO internal service provider, or organizations with whom GPO has agreements in place to ingest content. The ACP processing will check the digital objects, and take appropriate actions if the required objects were found missing. The actions include invoking automated process, such as format transformation for access rendition, or involving manual tasks such as editing access oriented metadata, setting the embargo date or performing the final optimization if applicable.

For R1C2, the majority of tasks for processing the ACP will be performed by FDsys automatically. The access oriented metadata and granules will be handled by the custom parsers. The parsers will follow the specifications of the DMD to extract the metadata from the content, and process the granules tagged by current content processing of the Plant Operations. The access rendition, e.g. the screen optimized PDF version of the content, is manually created and also available from the existing content processing at the Plant Operations. For a subset of the collections, a rendition folder will be created to hold the individual granules files in the text format from the FDsys parsers. The PDF rendition will be created by Plant Operations utilizing the PDF Generator within the Adobe LiveCycle services.

Figure 7.2-2 shows the model of FDsys ingest process. The actual implementation of the process is described in the repository design document, with details not shown in the model here. In addition to the Archival Storage as defined in the OAIS model, FDsys has two more separate storages: one for the ACP (Access Package Storage) which serves in many cases as a surrogate to the AIP when operations are required on the content files. This storage facilitates the daily content management for the packages ingested to FDsys, and is accessible only to authorized users.

Another addition is the Public Access Storage (ie ACP Cache) where a subset of the digital objects – the access renditions of the ACP along with the descriptive and access-oriented metadata is stored. This is the only storage that supports the general public access to the government publications. Upon successful completion of the access processing, the ACP is flagged with ready for publish. The FDsys publish program, design in the SDD Vol. III, will get invoked to publish the publicly accessible renditions from the ACP to the Public Access Storage for the search engine to index and make available for public access.

7.3 Archival Storage

The Archival Storage in the OAIS model interacts three other functional entities. It receives AIP from the Ingest and writes it to a dedicated storage media. In FDsys, the storage media for AIP is separate from the media for the SIP and ACP, so a different and stronger security control can be enforced to the AIP storage. Other functions of the Archival Storage are primarily handled in FDsys by the hardware storage management, which is responsible for configuration (and partitioning), error checking, backup and disaster recovery of the storage media for the system.

The Archival Storage in the OAIS model is also responsible for providing data to Access to fulfill orders, which is implemented in FDsys as content delivery in response to user request. The content delivery implementation includes a few scenarios. For authorized users, the content is delivered from the ACP in the form of a DIP, which can take different forms depending on the delivery request. The DIP can be created from the AIP as well, if for example the request is to transfer the AIP from the current archive to another. For public users, the content delivery delivers data from the ACP Cache (stored in the Public Access Storage in Figure 7.2-2), also in the form of the DIP. The content delivery is addressed both in the Content Repository (SDD Vol. II) and Custom Application (SDD Vol. VI) design documents.

7.4 Data Management

The OAIS Data Management functions specify capabilities of performing queries against the archive, and generating reports in response to the queries. The report queries can come from any of the other functional entities within the OAIS. Maintaining and updating the database, which includes the descriptive information about the packages, are also identified as part of the Data Management functions in the OAIS model. These functions are implemented in FDsys by leveraging the native capabilities of the selected CMS product – Documentum. It is noted that for SIP and ACP, the Data Management functions are supported by the CMS, but for the AIP the descriptive and provenance information is available, independently of the CMS, from the package itself as well.

7.5 Access

The OAIS model defines Access as for coordinating access activities, generating DIP, and delivering DIP in response to delivery request. The FDsys implementation for Access supports two scenarios for different user classes.

For authorized users, the Documentum WebTop-based user interface is provided for users to submit search queries. The result set of the queries are presented to the users. Upon user request of the content, a DIP is created (by the CMS Export functionality) and delivered to the user.

Content dissemination to the general public is one of the critical missions of FDsys. FDsys subsumes the functionalities currently available from GPO Access. GPO extended the OAIS model and created a new type of package – the Access Content Package (ACP) to specifically serve the purpose of providing public users the access to the content in the FDsys archive. The ACP contains access renditions of the archival package along with access-oriented metadata to enhance the user experience of content access.

The FAST search engine was originally implemented and then replaced by SOLR to empower the content search, and FDsys implements a full-fledged custom search web application, serving as a front-end to the search engine. NextGen also includes an open source Web CMS implementation to

further enhance the access function. Users submit search queries and the system return the search result set to the user. The DIP is created and delivered upon initiation of a content delivery request by the user.

The search and content delivery for authorized users is addressed in the repository design document (SDD Vol. II), and for public users the details can be found in the Search API (SDD Vol. IV), Search Engine Configuration (SDD Vol. V), and Custom Application (SDD Vol. VI) design documents. Collection specific access requirements are further described in the DMD for each collection.

7.6 Administration

The Administration entity in the OAIS model covers the services and functions needed to control the operation of the other OAIS functional entities on a day-to-day basis. The functions include negotiating submission agreements with (content) producers, establishing and maintaining archive standards and polices, etc. It also includes system engineering functions to monitor and improve archive operations. These functions in FDsys are performed by selected COTS products for monitoring system performance and figuring out bottlenecks if any. For example, the SOLR search engine is delivered with an admin console that allows monitoring system health and performing some basic performance tuning tasks dynamically. The Library Services and Content Management (LSCM) organization within the Government Publishing Office (GPO) negotiates submission agreements with content providers. Additional administrative function are performed by GPO's Official Journals of Government (OJG) and Programs, Strategy, and Technology (PST) organizations. Submission agreements with agency and content creators are documented using GPO standard form 1 (SF-1), Congressional requisitions, or Congressional authorization letters.

7.7 Preservation Planning

The Preservation Planning includes functions to evaluate contents of the archive and periodically recommend archival information updates, etc.

The objective of the R1C2 is to create and store the AIP, with functionality for preservation processes, such as the ability to update the descriptive metadata of the AIP. Preservation planning functions are performed by PST and LSCM. Preservation Planning functions performed by LSCM include the monitor designated community function and the develop preservation strategies and standards function. For example, LSCM interacts with Federal Depository Library users through surveys, conferences, and webinars to monitor service requirements for FDsys. New requirements and requests for changes are provided to PST for implementation. Preservation Planning functions performed by PST include the monitor designated community function, the monitor technology function, the develop preservation strategies and standards function, and develop packaging designs and migration plans function. For example, PST tracks emerging technologies and implements prototypes (e.g. SOLR open source search engine). Please refer to the Repository Design document for additional technical information including information about preservation reports and the preservation user interface within the repository. The repository integrity check is performed annually.

8. Business Process Implementation

8.1 Overview

In FDsys, business processes are implemented in two ways, by system workflows orchestrated by a workflow engine, and by functional utilities executed in fully automated system processes. The differentiation between the two is critical as the two mechanisms have completely different characteristics in terms of the system resource consumption, performance impact and maintenance effort. The system workflow that is based on a workflow engine is by itself a heavyweight, but feature-rich application. In contrast, an execution of functional utilities can be easily facilitated by lightweight system processes in the modern multi-threaded enterprise applications.

The design decisions of whether to use the workflow or functional utility to implement a business process are made primarily based on the considerations of system performance and features that are required to support the business process. A full-fledged workflow engine is normally utilized to support the task management with complicated workflow joining and splitting options in a business process, which usually involves automated as well as manual actions by participants of various business roles. A workflow engine for a simple automated set of lightweight operations would be a misuse and may inversely impact the system performance because of the overhead involved in the workflow management itself. As a general rule, only the following types of business processes in FDsys are implemented by using a system workflow.

- Business processes that involve automated as well as manual activities, and its lifecycle ranges from minutes to days or longer.
- Business processes with complicated joining and splitting operations.
- Business processes with completely automated activities and a long lifecycle that can and needs to be monitored.

The FDsys business processes are all related to the content and metadata management, the Business Process Management (BPM) product of Documentum was selected to fulfill the FDsys workflow requirements. The Documentum BPM implements a general-purpose workflow engine to support a variety of business processes. Naturally it provides a built-in integration with the Documentum content management system, which benefits greatly to a content-centric system like FDsys. The BPM will be configured to host the FDsys system workflows.

By analyzing the FDsys business processes and applying the decision rules described above, the following system workflows and processes will be implemented in R1C2.

- Submission process
- Ingest process
- Access processing workflow
- Archival storage process
- AIP deletion workflow
- Package updating process

It is noted that all diagrams in this section depict high-level business process flows, and serve as the guidelines for detailed design and implementation. The workflows, exact names and order of the activities in the actual implementation may slightly differ, and are documented in the individual detailed design documents.

Figure 8-1 shows an overview of the business processes that are supported by system processes and workflows. The business processes are grouped into four areas: submission, processing, preservation and access.

For content submission, three types of submission are supported: interactive, folder-based, and bulk submissions. A standalone bulk submission tool was developed to support the data migration process. In R1C2, the interactive and folder-based approaches are to support submissions from GPO Plant Operations and organizations with whom GPO has agreements in place to ingest content. The content files are uploaded to the system through a hot folder or interactive user login session with FDsys. Once content files are loaded to the work-in-progress (WIP) area, both submission types use the same user interface for the user to verify and finalize the packages before submission of the SIP for ingest. For select collections (e.g. USCOURTS, BILLSUM, BILLSTATUS, ECFR), an automated submission job has been set up.

Upon submission, the SIP is validated against the pre-defined packaging rules, and the ingest process continues if no validation errors were reported. Otherwise, the submission is rejected and a notification is sent to the designated service specialists. The objective of the ingest process is to create and store the AIP to the archival storage. As majority of the descriptive metadata in R1C2 is yet to be extracted by the parser at a later stage, the AIP thus created will have limited metadata information. The OAIS ingest functional entity implementation ends after the AIP is created and stored into the archival storage.

The ingest process is followed by a separate workflow for access processing, starting with the creation of the ACP. The content parser is invoked as a first step against the relevant renditions. The descriptive metadata extracted in this step are also used to update the AIP that is created previously at the ingest process.

The ACP is further processed after the parser invocation. The activities include digital signature signing for select collections, manual optimization of access renditions if configured during submission. The publicly-accessible renditions, and granules if applicable, are then published to the access subsystem by a publishing process. Additional information is available in the File Processor Design Document (SDD Volume LII) and the Publisher Design Document (SDD Volume III).

Following the access processing workflow, the ILS integration for bibliographic information exchange between the two systems takes place. Note that this functionality has not been implemented.

In the access group, a full-fledged search application is developed to support content search and delivery for the public users. The access and retrieval of the ACPs by the authorized users are facilitated by the Documentum applications with necessary configuration and customization.

The solid arrows in Figure 8-1 indicate the system control flows from one process (or workflow) to the next by the system-managed transition. The dashed arrows are used to indicate that the relevant processes are normally initiated manually.

The remaining of the section describes a few main business processes that are implemented in R1C2. The descriptions in this document are regarded as high-level business operations. The implementation details of the workflows and processes are in the Repository Design document.

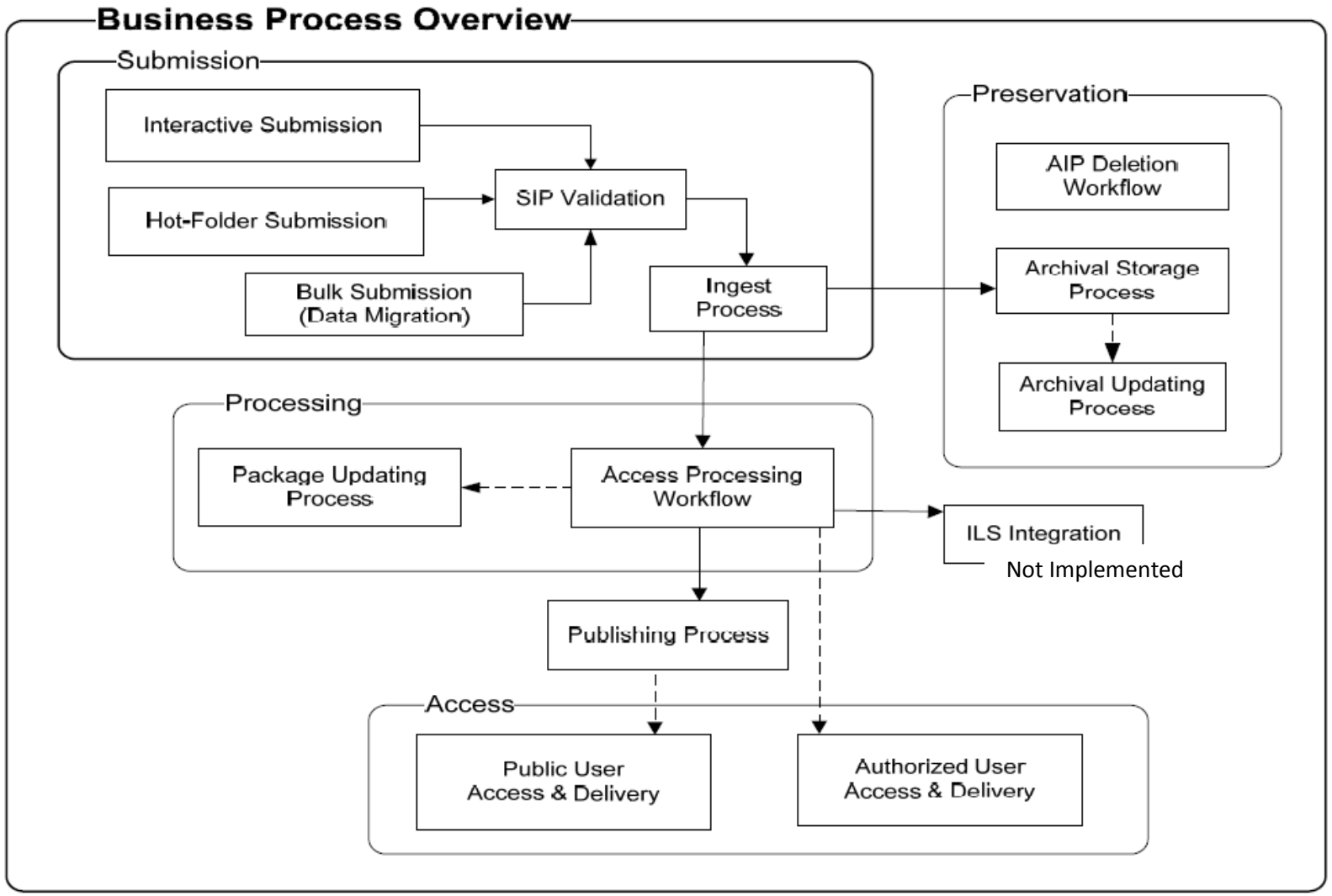

Figure 8-1 Business Process Overview

8.2 Submission

This is to support the day-forward content submission from the GPO Plant Operations to FDsys, as discussed in section 5. The submission can be categorized into two groups. In one group, a large number of files (around 100 or so) are submitted to FDsys on daily basis. For example, the Federal Register and Congressional Record belong to this group. In another group, the submission is much less frequent and also with a small number of files for submission. From the system architecture, development and maintenance effort perspective, submission through the interactive FDsys user interface is a preferred way to ingest the content into the system. However, because of the relatively large number of files involved and thus higher risk of manual submission errors in the first group, it was decided to develop a folder-based submission to support submission of a selected set of collections.

8.2.1 Folder-Based Submission

For R1C2, the following collections are supported by the folder-based submission:

- Congressional Bills
- Congressional Record
- Federal Register
- Congressional Reports
- Congressional Record Index
- History of Bills
- Congressional Calendars
- Congressional Documents
- Daily Presidential Documents
- Code of Federal Regulations
- Congressional Hearings
- Congressional Committee Prints
- Economic Indicators
- Public and Private Laws
- Public Papers of the Presidents
- GPO Collection
- GOVPUB Collection
- United States Courts Opinions Collection
- Statutes at Large
- Privacy Act Issuances
- List of CFR Sections Affected
- eCFR
- Bill Summary (Note: Retrieved from External API)
- Bill Status (Note: Retrieved from External API)

The submission for the remaining collections (i.e. collections not listed above) will be through the interactive FDsys user interface.

The submission process starts with placing the content files into an FDsys hot folder by the GPO Plant Operations or an organization with whom GPO has an agreement to ingest content (e.g. OFR, United States Courts). A separate folder will be established by FDsys for each of the selected collections, and the folders are accessible to users in Plant Operations or the organizations with whom GPO has an agreement to ingest content. The folder location, structure and applicable filename conventions are collection specific and are specified in the detailed implementation design. The hot folder (shown later in Figure 9-1) is a dedicated storage area with the accessibility granted to the CMS content server, as well as approved users. The content files placed in the hot folder will be picked up by a configured CMS job, and stored to the CMS repository in the form of a SIP, according to the pre-defined packaging rules for the collections.

When a specialist, who is responsible for submitting the collections in the hot folder, logins to FDsys, the specialist is presented with the SIPs (one for each collection) with content files (i.e. renditions) that were placed to the hot folder. Figure 8.2-1 shows the submission process flow for the folder-based approach.

The specialist verifies and validates the renditions in the SIPs. If content files were misplaced or missing from a collection, the specialist can move the misplaced files or upload missing ones for the collection. The validation continues until no errors were found, and then the specialist submits the SIP for ingest. For select collection (e.g. USCOURTS, eCFR, BILLSUM, BILLSTATUS) an automated submission job has been configured to submit content on behalf of the specialist. This functionality is further described in the Repository Design Document.

The ingest process starts with parsing the select content files for descriptive and technical metadata, and generating granule files if applicable. The metadata and granule files are added to the package and another validation is performed to ensure the mandatory metadata for the AIP, etc. The ingest process continues to transform the SIP into AIP and ACP upon successful validation. Otherwise the system control returns back to the specialist for manual corrective actions.

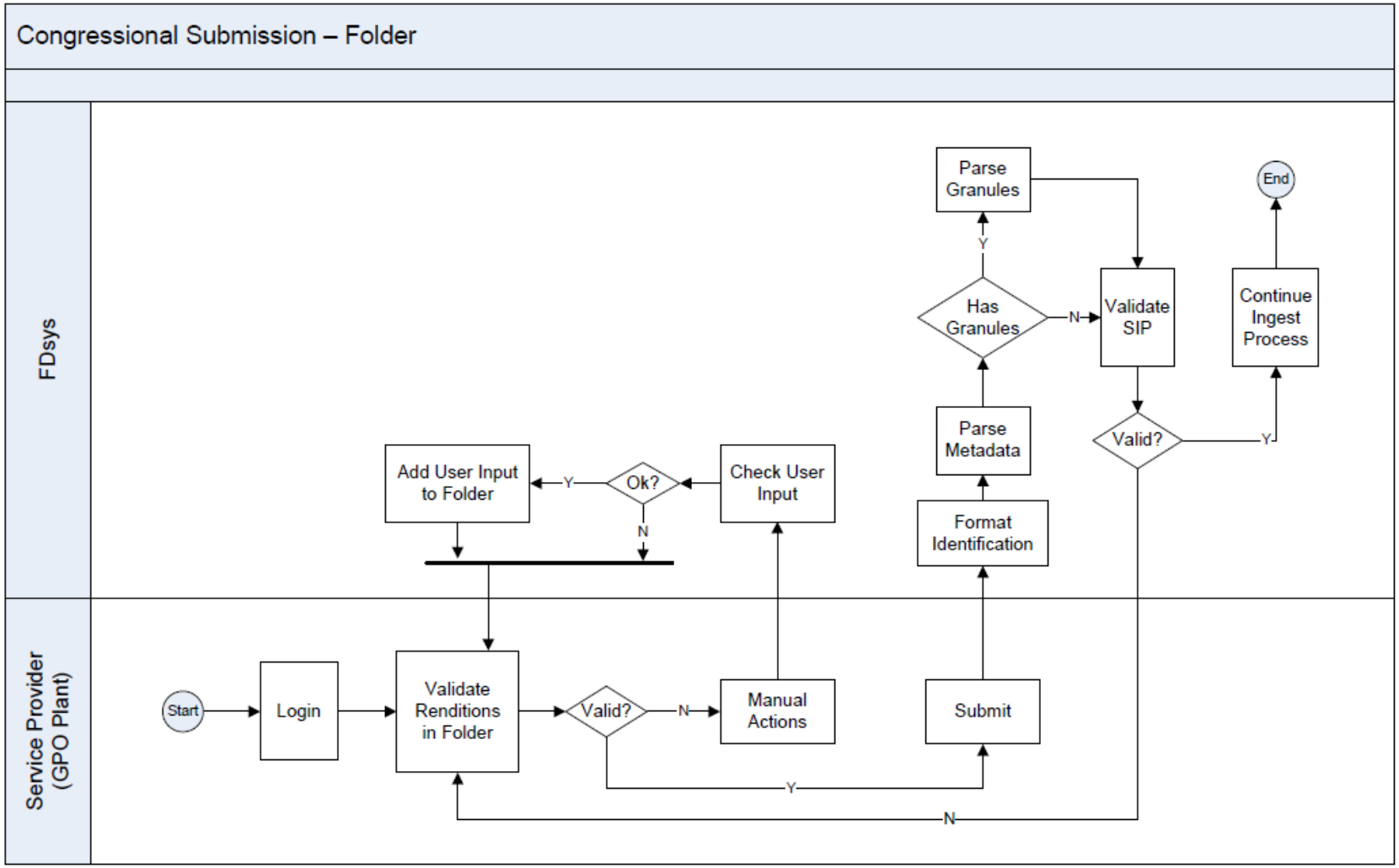

Figure 8.2-1 Folder-Based Submission Process

8.2.2 Interactive Submission

This is to support the interactive content submission from Plant Operations, PST, or LSCM through the FDsys graphic user interface. The collections that are not supported by the folder-based submission will be submitted to FDsys through this approach.

Figure 8.2-2 shows the process in the swimlane form. Upon receiving the first user input of an uploaded file or a piece of metadata, a package in the SIP form is created in the work-in-progress area. Subsequent user input is added to the package. The input process continues until the user submits the SIP.

The submission action of the user triggers a series background processes, ranging from parsing the metadata, content granules if applicable, to validating the SIP itself. If the SIP is valid, the ingest process continues. Otherwise, and an error message is sent to the submitter's inbox for further manual actions.

The two submissions differ only in how the content files are loaded to the system. Upon submission of the uploaded packages, the processes for the two scenarios are the same.

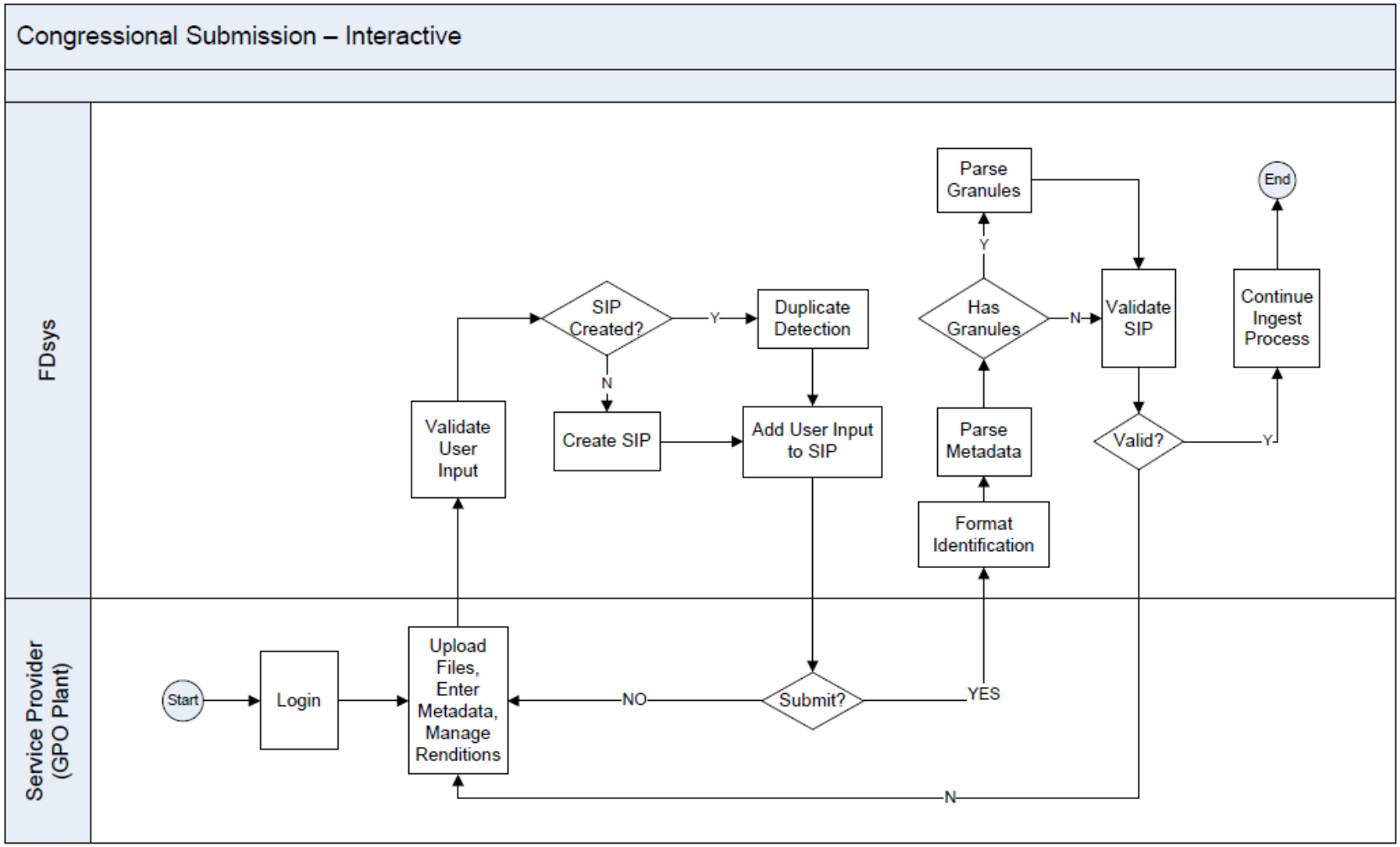

Figure 8.2-2 Interactive Submission Process

8.3 Ingest Process

Upon successful validation of the SIP, the ingest process continues with the AIP creation. Otherwise the submission is rejected and a notification is sent to the designated authorized user.

The first step for the AIP creation is to apply timestamp signature to the digital objects within the package. The timestamp signature is defined and calculated as follows. A first hash value is calculated based on the unique ID of the digital object and the hash value of the digital object itself. The timestamp signature is defined as the new hash value based on the first hash value plus the current time (in seconds, since January 1, 1970). The algorithm is illustrated in Figure 8.3-1.

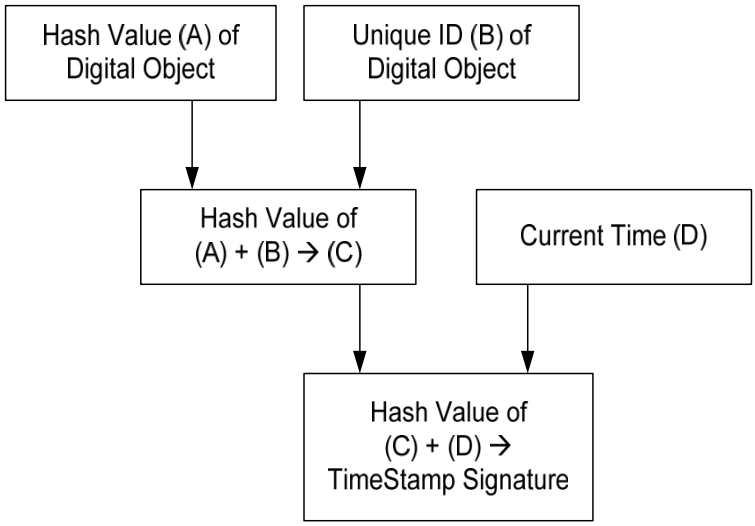

Figure 8.3-1 Algorithm of Timestamp Signature

Note that all elements (hash value of the digital object, unique ID, and the time) of the timestamp signature are themselves stored as metadata information for the AIP. The timestamp signature is stored in the package separately and used as an assurance of the fixity – changes to any of the mentioned elements (content, id, and time) will compromise the signature.

Figure 8.3-2 shows the ingest process for the AIP creation. After applying the timestamp signature, the process continues with gathering preservation information, such as file format, and preservation rendition in the XML form. The XML files of MODS, PREMIS and METS are generated for the AIP using the content files and metadata info available up to this point. After successful validation of the package, the AIP is written to the archival storage.

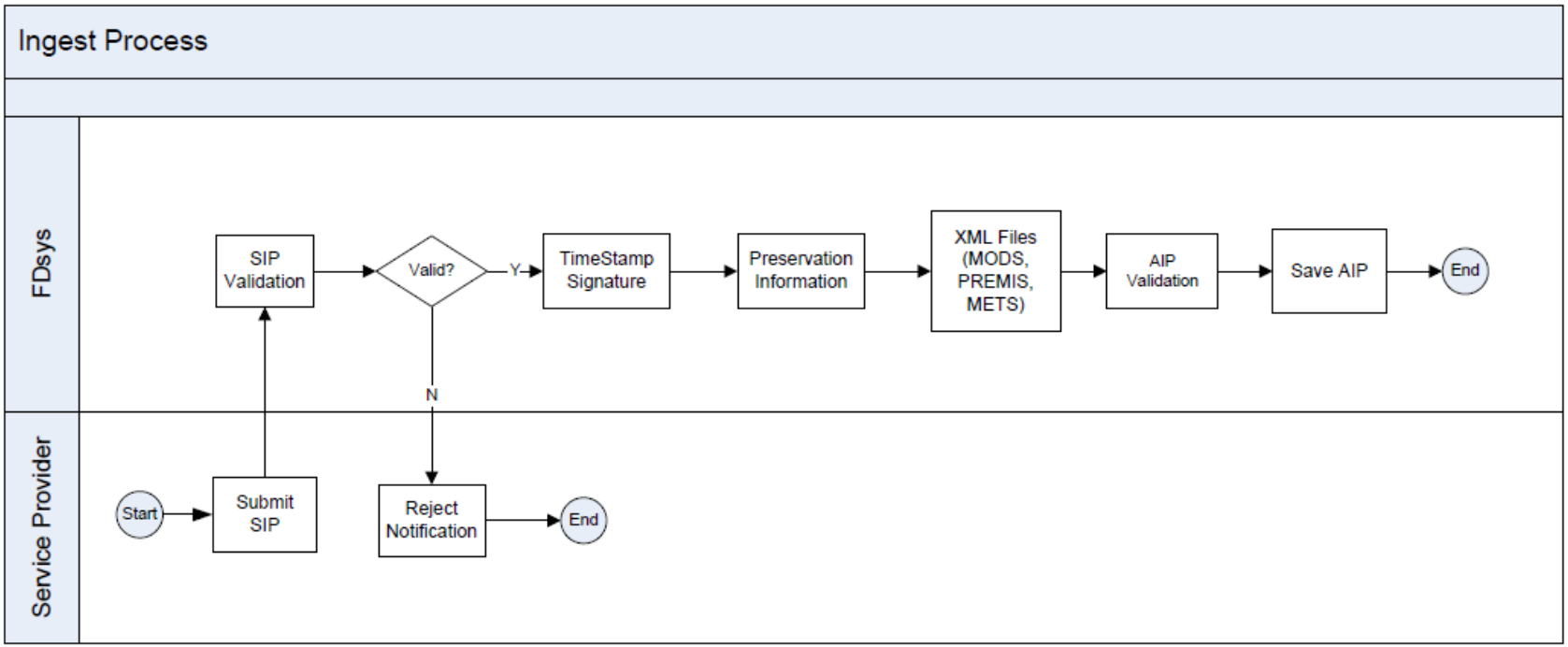

Figure 8.3-2 Ingest Process

8.4 Access Processing Workflow

Figure 8.4-1 shows the access processing workflow. The workflow starts with the creation of the ACP. The content parser is then invoked to extract the descriptive metadata and generate granule files if applicable. The system checks if the results from the parser is valid according to the pre-defined parsing rules. The parsing rules for this check include:

- Availability of the required metadata elements for the AIP and ACP
- Availability of the expected granule files if applicable

If any of the required metadata or granule information is missing because of the parsing errors, the designated authorized user is notified for manual actions. The actions could include: a) instructing the system to continue if the package is still acceptable with the missing information, b) manually making corrections to the missing metadata (and the workflow continues in this case), and c) terminating the processing workflow. In case c), the submission of the SIP will result in an AIP in the archival repository and ACP in a separate repository which has not been successfully processed because of the parser failure. The ACP will not be published for public access since the processing did not run through to the end. The workflow can be later re-started on the ACP already in the system when an improved version of the parser becomes available. The workflow in the repository must be designed to be able to support this capability.

The parsing errors can be result of a variety of causes. The discussions above assumed that the parsing errors were due to the parser's failure to perform its tasks. The parsing errors could be caused by the corrupted data as well, in which case the package (ACP) should be removed from the system, including the corresponding AIP as it provides no value to preserve the corrupted content from the very beginning. For parsing error due to corrupted data, FDsys provides an option for the authorized user to initiate a complete deletion of the packages (ACP and AIP) from the system. In fact, because the system has no knowledge about whether the data is corrupted or not, it is the user who has to confirm that the data is indeed corrupted by for example opening the content file before instructing the system for deletion. The designated authorized user is informed of the deletion and the SIP has to be resubmitted. Note that this type of deletion is performed as part of the extended ingest process to ensure data quality since it provides no value to preserve data that is known to be corrupted from the very beginning. So the deletion can take place without special approvals, which are required for deletion of the AIP whose corresponding ACP has been successfully processed and published for public access.

Upon successful parser invocation and validation of the parsing results, the processing workflow updates the corresponding AIP with the extracted descriptive metadata (specifically to the MODS file). The Generate Renditions activity in the workflow includes PDF chunking for select collections, and access rendition in html for the ACP, etc.

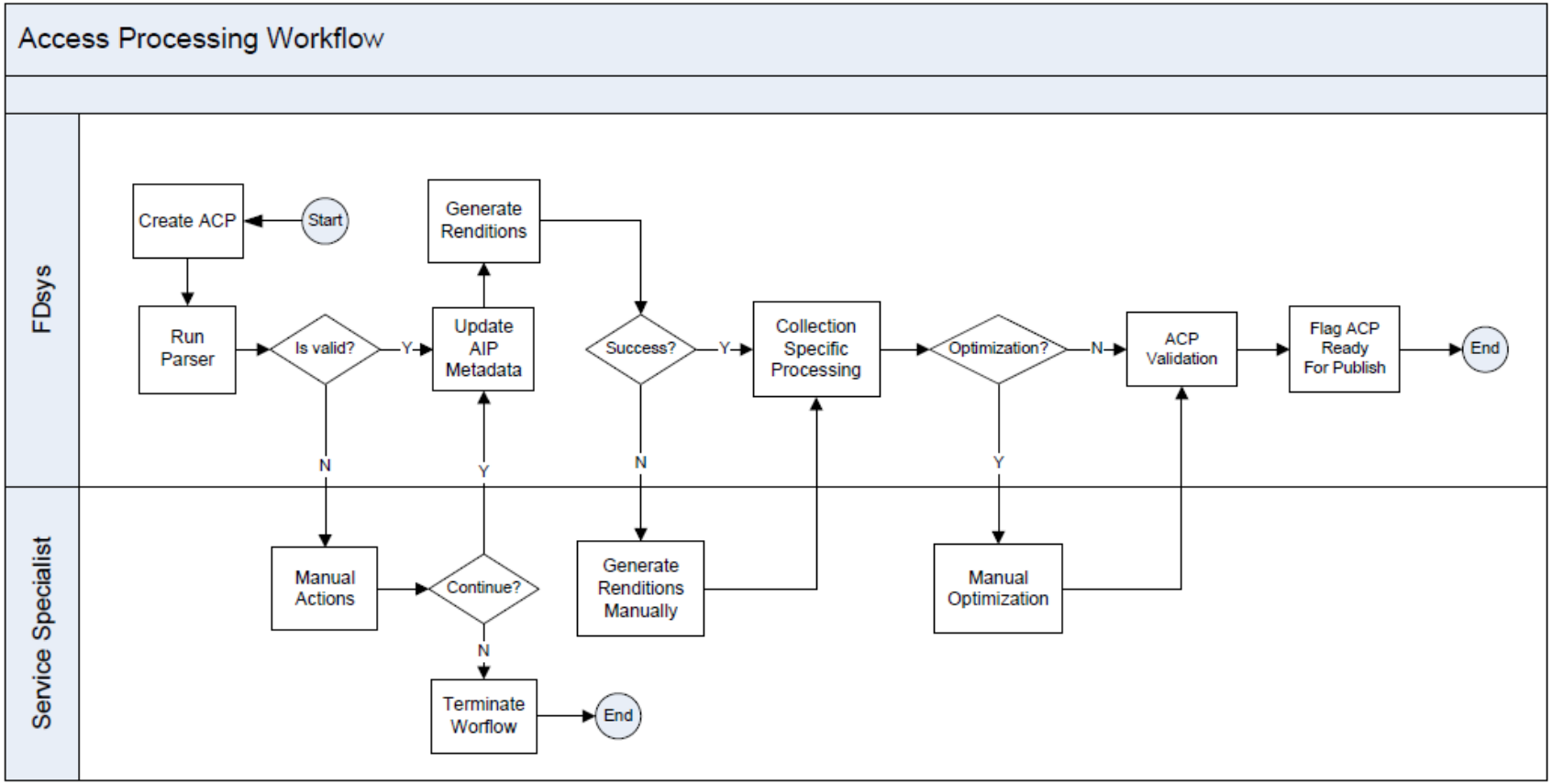

Figure 8.4-1 Access Processing Workflow

The collection-specific processing is facilitated by configurations for processing needs for the relevant collection. The digital signature signing for publicly accessible PDF files for select collection takes place at this step as well. If manual optimization is flagged during submission, the workflow pauses for manual actions to, for example, verify and improve the quality of the access renditions. The manual interactions in the workflow also include any automated functional errors during rendition generation or package validation (though not shown in the diagram for presentation clarity).

The package is validated again after the series of automated and manual processing activities. Following a successful validation, the ACP is marked as ready for publish. The descriptions here are from the business operations perspective. The actual workflow design and configuration details are documented in the SDD Vol. II: Content Repository.

8.5 Package Updating Process

This is to support the authorized users for updating the existing ACPs in the content management repository. ACP updates apply to both the rendition files and metadata. Use cases of the package updates include adding more access metadata to enable more efficient search or to replace with a better access rendition file for public access. Updates in the workflow cover operations of adding, removing and replacing for metadata as well as content files.

Figure 8.5-1 shows the package updating process for the ACP. The process starts by querying the ACP for update. The query is facilitated by the general search and retrieval capabilities of Documentum. The selected ACP for update is transparently checked out from the repository. The user performs metadata edits and/or rendition updates by uploading new or modified rendition files. Every updated input is checked and validated by the system to ensure that the input is legal and valid. For example, mandatory metadata can never be removed from the package, and uploaded rendition file must be able to pass the virus check. The updating actions continue until the user instructs the workflow that the update is completed.

Once the update is complete, the system checks whether the updated ACP requires the re-publish for public access. The ACP is re-published only if the publicly accessible rendition or metadata were updated.

The fact of updating the ACP is recorded as the PREMIS event to the corresponding AIP in the archival repository. In addition to recording the updating event itself, if the update was made to the metadata in the ACP, the relevant metadata elements in the AIP will be updated as well. In other words, the updates to the metadata to the ACP and AIP are automatically synchronized between the two repositories. The updates to the renditions, however, are independently made and maintained at the individual repositories.

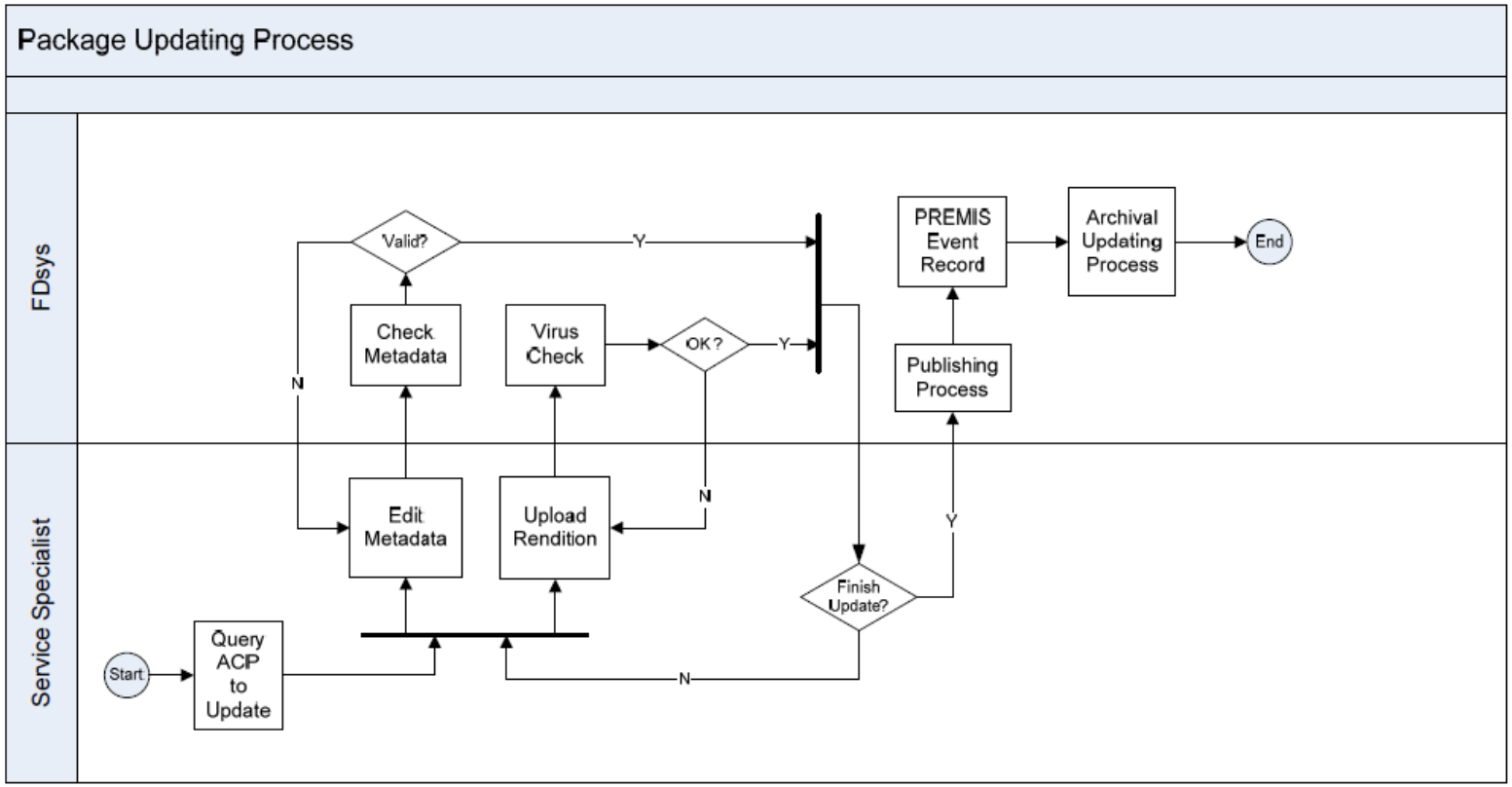

Figure 8.5-1 Package Updating Process

8.6 Archival Updating Process

This is to support updates to the AIPs in the archival storage. A process can be initiated in two ways. First when the metadata in the ACP is updated, the corresponding AIP is updated automatically with the new metadata information as well. This update requires no manual intervention. In the second scenario, a process is initiated when a preservation specialist issues an update request to the selected AIP from the archival storage.

Figure 8.6-1 shows the archival updating process in the swimlane form. Notice that FDsys is shown in two separate lanes, to cover two cases in which an updating process may apply.

Similar to the package updating process for the ACP, preservation specialist can query AIP for updating the metadata information. For rendition updates, the system creates a new AIP with the updated renditions. The new AIP is assigned with the unique ID of the old AIP, and the old AIP with a new unique ID. This is to ensure that the existing package reference now point to the updated AIP without the need to update the references in any referencing packages. All update actions are recorded as the PREMIS events in the relevant AIPs, old and new.

The preservation specialist can query AIP from the archival storage and request a deletion of the AIP from the system. As the AIP is highly protected, the deletion action requires extra processing and a separate workflow is configured for the AIP deletion described in the next section.

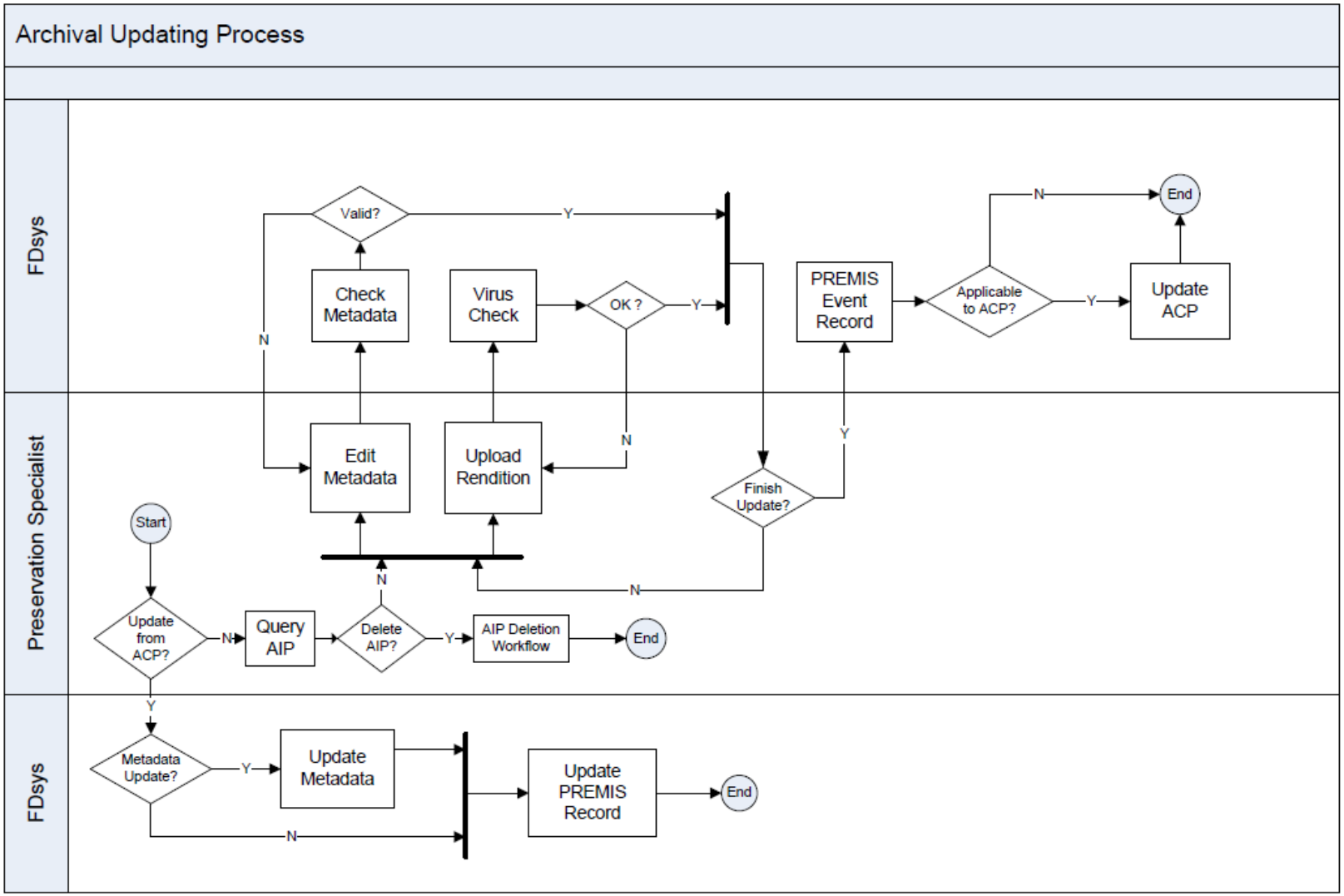

Figure 8.6-1 Archival Updating Process

8.7 AIP Deletion Workflow

The AIP in the archival storage is highly protected in FDsys. An AIP deletion from the system requires all of the three authorized approvals. The deletion requester can be one of the authorized approvals. The AIP deletion is expected to happen only in very rare cases, such as a wrong package (out of FDLP scope for example) was mistakenly ingested to the archival storage.

Figure 8.7-1 shows the AIP deletion workflow. Upon approval of all three, the corresponding ACP and ACP cache are deleted first, followed by the deletion of the AIP itself. If any of the three denies the deletion request, deletion rejection notification is sent to the requester, and the workflow ends.

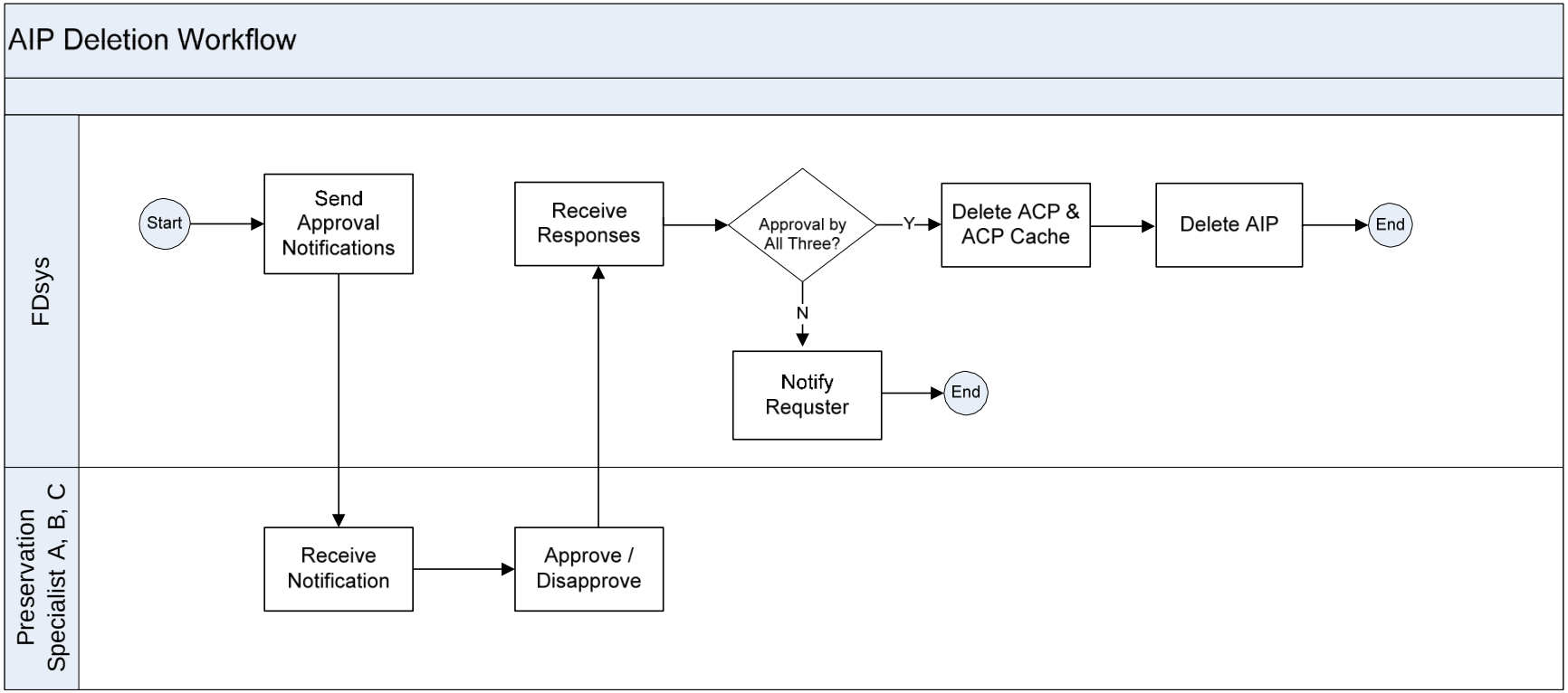

Figure 8.7-1 AIP Deletion Workflow

9. Data Migration

The goal of the data migration is to migrate content collections currently available from the GPO Access (i.e. WAIS file systems), and associated content files in various formats and images on Jukebox and Alpha3 servers to FDsys. Because of the diversity of the data sources and large variations in how the data are stored from one collection to another (and from one period of time to another for the same collection), the data migration process is divided into two steps. The first step will consolidate the data from various sources to a centralized storage area. At the second step, the content files will be re-arranged to the FDsys package folder structure – a unified structure that is understood by FDsys applications for submission and ingest.

While the details of the data migration design are yet to be developed as a separate design document, the rest of the section describes the current understanding of how the data migration needs to be carried out.

9.1 Data Analysis

For data migration, the purposes of the data analysis are:

- To create an inventory of data files of various sources for migration
- To establish the collection-specific packaging rules, including applicable exceptions
- To specify metadata fields that are only available from the original data folder structures
- To perform data cleansing if necessary to improve the quality of the migration data

The packaging rules include: expected file formats (and thus renditions) for the collections, file naming conventions used by the collections. These rules will be turned into the rules configurations by the content source module, which probes the migration data folders and returns the data source information in the form of the FDsys packages.

9.2 Migration Tool

The objectives of the data migration tool are:

- To configure the packaging rules for the migration data from various sources, including WAIS, Jukebox and Alpha3 severs
- To copy and arrange the content files into the package folders according to the configured rules
- To glean metadata information that is only available from the existing data folder structure
- To generate report of migration status

9.3 Migration Setup

GPO Access currently has about 350 GB data on the live WAIS infrastructure and other file system (Jukebox and Alpha3) with numerous variations in collection specific folder structures. To minimize the impact on the live system, a staging area is used to store the WAIS data with the exact copy from the live system. It also provides a central storage area to consolidate the migration data from the other two sources and associated them with those from the WAIS file systems. Figure 9-1 shows the schematic diagram for the migration setup.

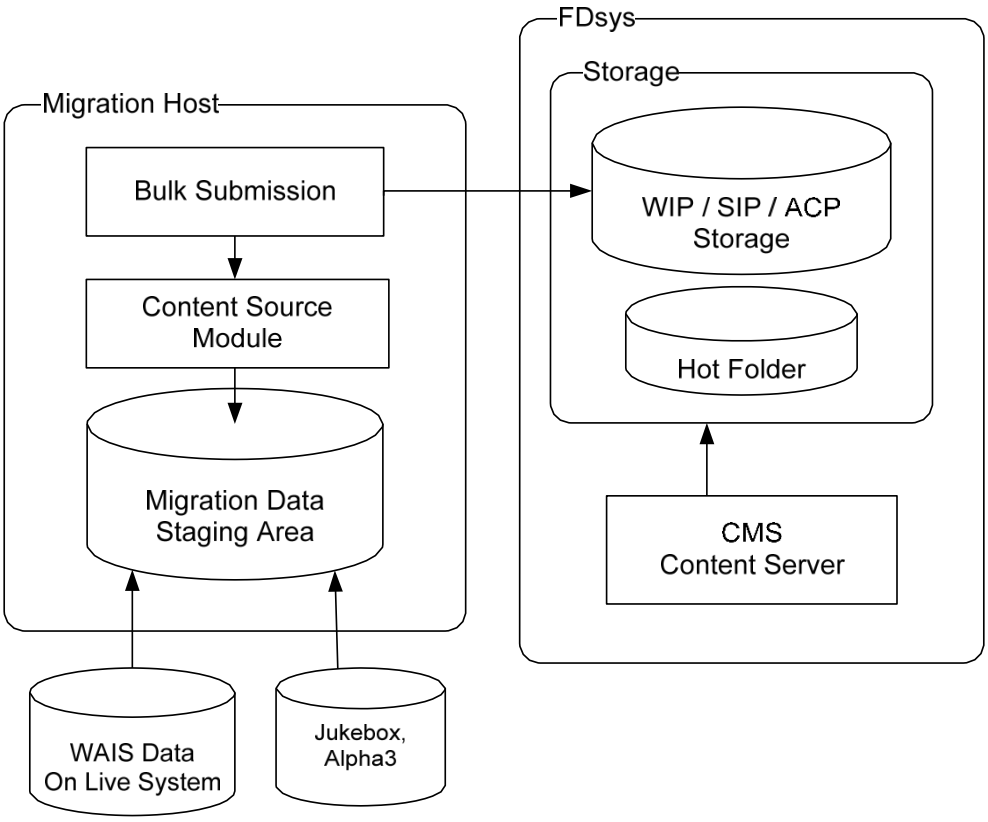

Figure 9-1: Data Migration Setup

To illustrate their different purposes, two storage areas on the FDsys side were drawn in Figure 9-1, but they may just be separate logical partitions on the same NAS storage device. While the CMS content server has access to all of them, the migration host will have access only to the WIP area. A storage area marked as hot folder is setup to support the folder-based congressional submission. The access to the hot folder is limited to the CMS and users from the Plant Operations who place the content files into the folder for submission.

With package information returned from the content source module, the Bulk Submission tool copies the original content files from the migration staging area to the WIP area of FDsys in the form of the package folders. The CMS content server can pick up the files already arranged in the package structure and performs ingest process, just like it does for the folder-based submission but without manual interaction to verify and update the renditions. When packages are picked up from the WIP area for migration, the ingest process starts immediately as if a user submitted the package for ingest.

The same ingest process designed for interactive and folder-based submission can be used for migration as well, so the content parsing, rendition processing, package archiving and publishing to access all happen within the content server. To minimize impact to the live production system, one option is to standup a separate Documentum instance for migration, and replicate the objects from the migration instance to production instance later.

10. Application Integrations

10.1 Integration Overview

FDsys is COTS based system, and integrations between the COTS products selected for FDsys are naturally of significant impact on the system architecture. To maximize the architectural extensibility of the system while minimize the performance impact, the integrations are categorized into two groups, depending on whether the integration needs to support one-way or two-way interaction. In the one-way interaction case, the service request or data flow goes from FDsys to the other integration endpoint only. The two-way integration covers cases in which FDsys and the other integration endpoint request services or data from each other.

The integration can also be categories as internal and external. Internal integration refers to cases where the COTS products are selected specifically for FDsys. External integration covers scenarios where FDsys needs to interact with existing systems for services or data exchange (e.g. Library of Congress' Congress.gov API).

For one-way integration, FDsys adopts the standard protocol or API based approach to maximize the performance gain by using the native application protocol to connect the two points. The same API-based approach is utilized for internal integration as well. This approach is feasible because the changes at the other integration endpoint will have minimal impact on FDsys as long as the communication protocol remains the same, or FDsys controls both the integration endpoints.

For two-way integration, the benefits of architectural flexibility outweigh the performance gain because FDsys has no control over the changes of the other endpoint of the integration and vise versa. In this case, the ESB can be used to mediate the communications between the integration endpoints.

The remaining of this section highlights the integration types and approaches adopted by FDsys for the integrations. The low-level interface details are described in the individual detailed design documents.

Figure 10.1-1 shows the FDsys integrations for R1C2. The solid arrows indicate the data flow between the integration endpoints, and dashed arrows show service request direction.

As can be seen from Figure 10.1-1, all integrations, except one, in FDsys are one-way including two external integrations. As Documentum manages the content repositories for FDsys, it is naturally one of the integration endpoints for all integrations.

Documentum – SOLR Integration

This is an internal integration that bridges the content management and access subsystems within FDsys. After a package is submitted and ingested into the content management subsystem (Documentum), the access renditions of the package are published to the access subsystem for the SOLR search engine to index and make it available for public access. It is a one-way integration since the data always flows from Documentum to SOLR. The integration is based on the Documentum APIs for the SOLR component to query and retrieve the files from the repository to the

access side. The Documentum APIs are one of the key deliverables of the COTS product, and the backward compatibility has been supported throughout the history of the product.

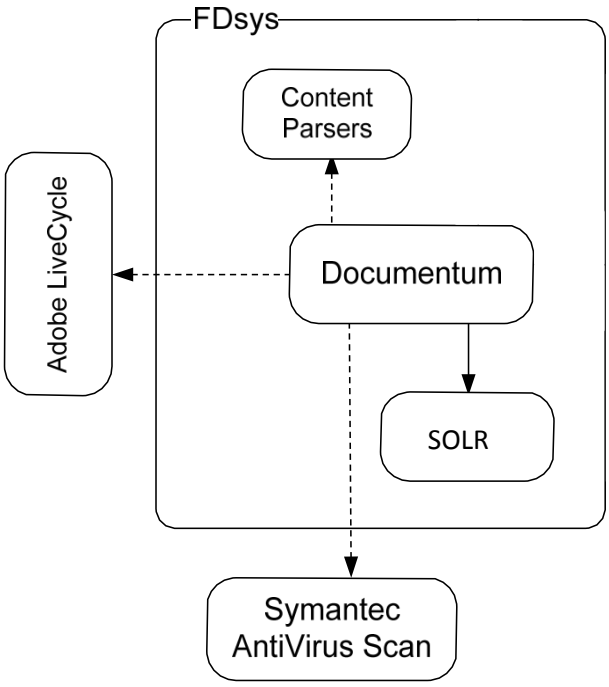

Figure 10.1-1 FDsys Integration Endpoints

Parser Integration

Because of the complexity and the required resources to develop the parsers, the content parsers are developed and tested in a separate, dedicated effort. The parsers are integrated with Documentum ingest process to extract the metadata information and generate granule files for a subset of collections. The integration is entirely internal, and is based on APIs defined by FDsys. The parsers are delivered in the form of a binary library that implements the FDsys-defined APIs. It is a one-way integration as the parsing request always goes from Documentum to the parsers that perform the requested task and return the results back to the Documentum.

Adobe LiveCycle Integration

The Adobe LiveCycle ES3 provides digital signature signing and a set of PDF manipulation (e.g. chunking, format conversion) services. The service request is always issued by FDsys to the LiveCycle which functions as a service provider to FDsys. The LiveCycle is delivered with a set of supported APIs that are specifically designed for system integration. To streamline digital signature signing and PDF manipulation process, FDsys adopts the API based approach to integrate the LiveCycle in the package ingest process. It is a one-way but external integration based on the application protocol that is supported by the COTS product.

ListServ & eDocket Server Integration

For R1C2, FDsys delivers the table of contents of the Federal Register to the ListServ and eDocket server to keep the links live from the systems external to GPO. The delivery is done through the standard FTP protocol which Documentum provides the built-in support. The integration is one-way in that FDsys drops the data over to the target location only. This was not implemented per business requirements.

Virus Scan Integration

FDsys integrate with the Symantec AntiVirus Scan engine for virus check before content gets ingest to the system. This is a one-way integration in that the service request is always issued by FDsys to the Symantec engine. The integration is based on the APIs provided by Symantec engine.

ILS Integration

This is a two-way and external integration for bibliographic information exchange between the two systems. As the ILS is an exiting and independent system, the integration must be able to support the loosely coupled model to accommodate changes at both ends. For two-way external system integration, FDsys adopts the ESB-based approach.

The ILS integration is planned for implementation post Release 1, as of October 2010.

As of July 2015, this has not been implemented.

10.2 External System Integration

The external systems in this document include systems within and outside GPO. Within GPO enterprise, FDsys interfaces with the ILS in the same CMS pillar for bibliographic information exchange. FDsys will also interface with applications in other two pillars of the GPO enterprise information systems.

For Release 1, no external system integration with FDsys had been implemented as of October 2010. The following high-level design, however, should stand as guidelines for later detailed design for implementation. See NextGen External Systems section below for current external system integration.

The design goal for external interfaces is to ensure the flexible adaptability of FDsys to changes in the external systems as well as FDsys itself. Following the industry best practices for enterprise integration, FDsys adopts the ESB-based integration approach to interface with external systems. By design, the ESB plays a broker role for the participating systems. The individual systems register and request services to and from the ESB without the need to be aware of the details of how the system on the other end receives or provides the service. This loosely coupled integration model makes the changes at one integration endpoint transparent to the other endpoint. The ESB becomes the central location to manage interface changes for the integration endpoints. Majority of ESB implementations use a messaging middleware for its backbone communication, and the messages are usually in XML. Figure 10.2-1 shows a schematic view of the integration approach that can be used as a guideline if required for FDsys.

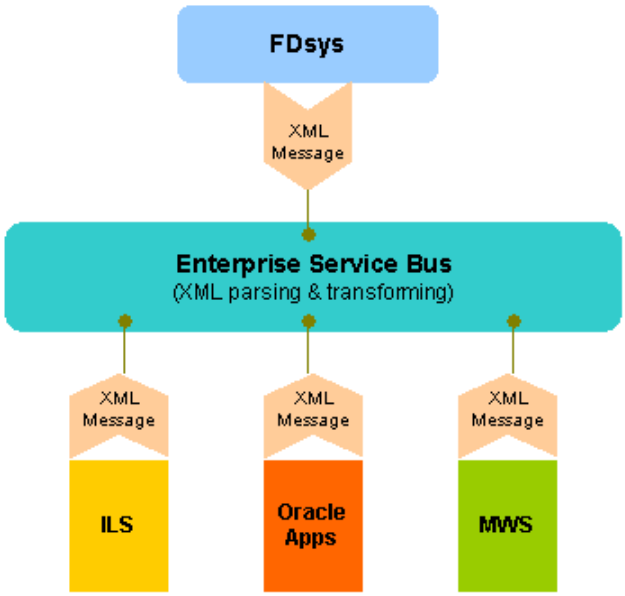

Figure 10.2-1 Application Integration through the ESB

The interfaces between the ESB and integrating systems are all in XML, and the ESB is responsible to parse the XML messages and transform the messages, along with the payload of application data if applicable, from the schema of the sender to the schema of the receiver. Notice that to illustrate the integration approach, Figure 10.2-1 showed a few examples for Oracle Applications and MWS as well.

The ESB-based integration strategy will be applied to all cases in which FDsys needs to interface with external systems within the GPO enterprise. For interfacing with systems outside GPO, extra security measures must be taken into account, and the appropriate approach will be designed in the system design documents for later releases when applicable.

ESB-Based Integration with ILS

The Integrated Library System (ILS) of GPO maintains the catalog of the federal government publications, including the content currently available from the GPO Access. Once the GPO Access content is migrated to FDsys, the catalog needs to be updated to reflect where the content can be accessed. For publications ingested to FDsys, FDsys needs to pass the bibliographic information about the publications to ILS to keep it up-to-date. Also any updates made to the ILS catalog for content in FDsys need to be propagated to FDsys as well to keep the two systems in sync. Synchronization between the two systems of updates that are made to the ILS bibliographic metadata is limited only to those records that have the unique FDsys ID.

FDsys initiates actions for both sending and pulling bibliographic data updates (including new records) to and from the ILS, to minimize the impact and development effort on the ILS side.

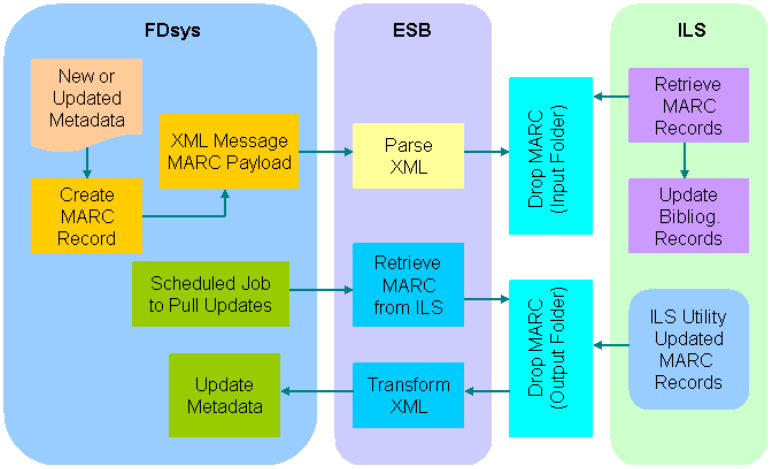

Figure 10.2-2 FDsys and ILS Integration

For sending new or updated metadata to ILS, FDsys initiates the action that completes when the MARC record is successfully dropped by the ESB to a configured location – an Input Folder for ILS. A scheduled job needs to be configured on the ILS side to periodically poll the MARC record and, if found, update the relevant bibliographic record. This scheduled job on the ILS side is completely independent of FDsys.

An Output Folder is also configured for ILS to drop MARC records that are updated by ILS. A scheduled job is configured on the FDsys side to periodically poll MARC records in the Output Folder. If found, the MARD records are transformed into XML format and the relevant FDsys metadata information is updated.

While the above discussion provides an overall architectural guideline for the integration, detailed design to support the architecture has to be flashed out before the implementation. Due to various reasons, the implementation of FDsys-ILS integration was deferred from Release 1 to a later release (as of July 2015).

NextGen External Systems

FDsys consumes a limited number of external services via Application Programming Interfaces (APIs). As illustrated in figure 10.2-2, services include the Library of Congress' Congress.gov API for Bill Summary and Status data in XML, Google Maps to support the DCPD mobile web application, and data from GPO's Member Guide site to support the Member Guide mobile web application. Note that the Library of Congress utilizes the services of api.data.gov for key management. The Mobile Member Guide and DCPD are further described in the Mobile Web Application Design Document. Authorized users utilize GPO's Tumbleweed software to transfer content via SFTP. Figure 10.2-3 depicts external interfaces for both the Production and COOP instances of FDsys.

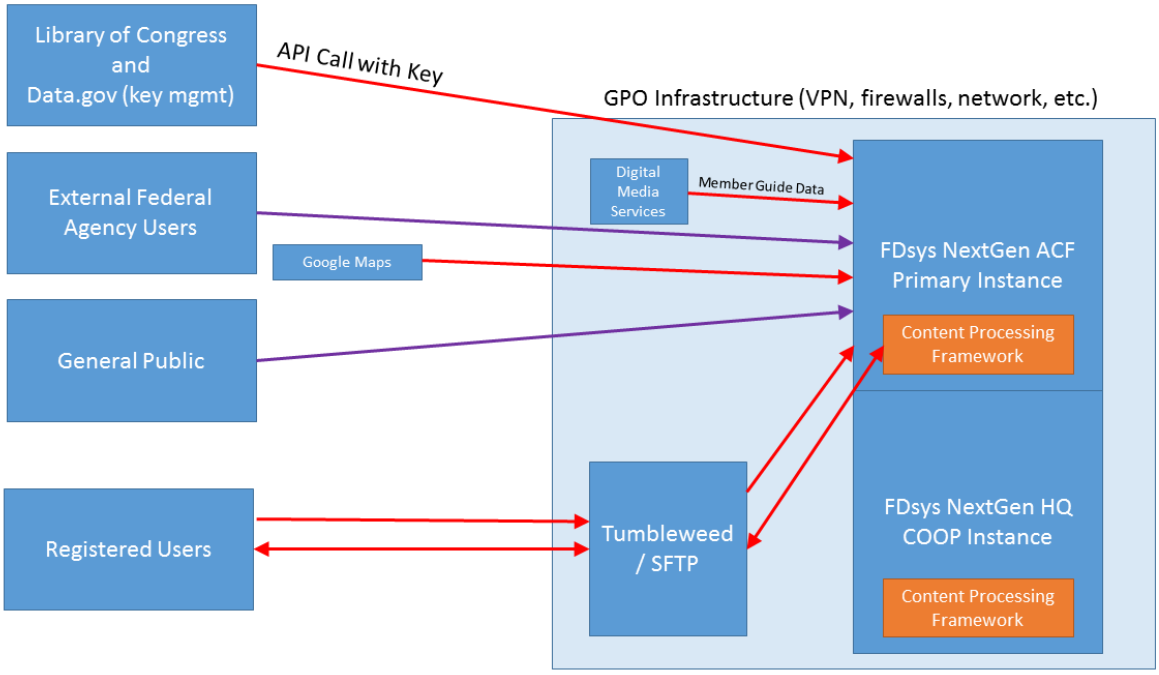

Figure 10.2-2 External Integration in NextGen

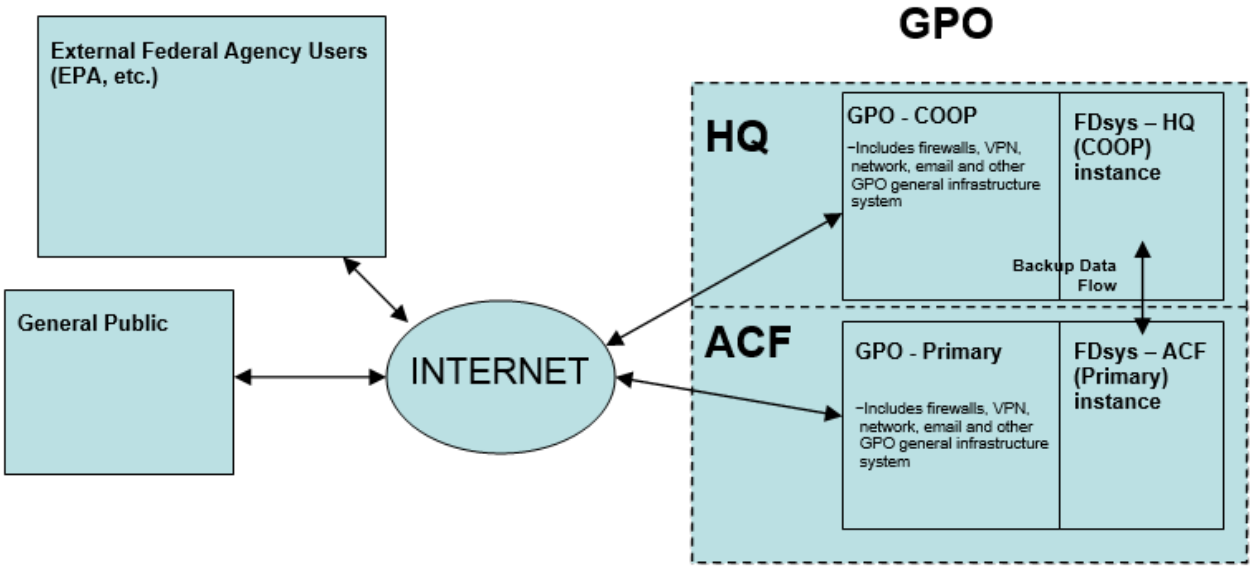

Figure 10.2-3 External Interfaces in NextGen

11. Hardware Architecture

The high-level storage and network architecture was addressed at the system architecture section. This section provides detailed architecture views for storage, computing and network connectivity.

11.1 Storage Architecture

The storage for FDsys includes:

- Storage for content submission (WIP area for SIP), including Hot Folder area
- Storage for daily content management operations (ACP)
- Storage for archival packages (AIP)
- Storage for publicly accessible contents (ACP Cache)
- Full-text indexes storage for the publicly accessible contents
- Storage for virtual machine images and content
- Databases for the Oracle application

The types of storage are used for FDsys: DAS (Direct attached Storage) for local operating systems, logs, and full-text indexes for publicly accessible content, Storage Network for virtual machine images and content, Fiber Channel storage for Oracle Databases, and the remainder on NAS (Network Attached Storage) for content files. Figure 11.1-1 shows the storage architecture. Notice that the accesses to the storage are through different Virtual LANs, so the security settings for the storages are configured to allow the access only from the specified (virtual) network, in addition to the server hosts and user access privileges.

Hot Folder

This is staging area with a small capacity (~600 GB) to enable the folder-based submission. It has to be accessible to the Documentum content server, and the designated user(s) from the GPO Plant Operations, LSCM, PST, and organizations with whom GPO has agreements in place to ingest content or process content. Files are added to or retrieved from the hot folder.

SIP / ACP Storage

This storage supports uploading and validating content files to finalize the SIP for submission. This is also where the ACP content files are stored to support daily content management operations, and is accessible to the content servers only. It requires large capacity of storage (~ multiple-TB). Access to this storage area should be limited to the content servers only.

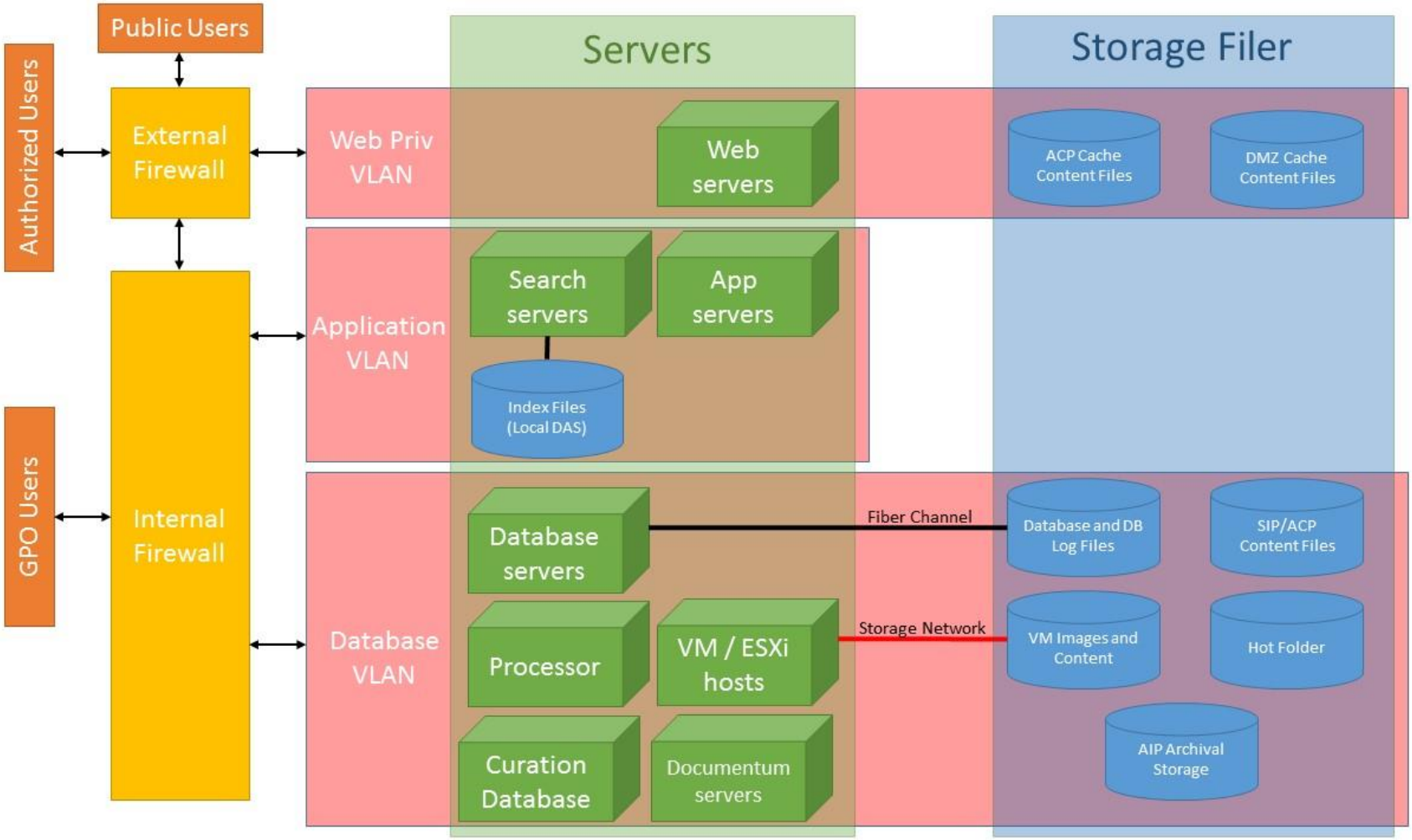

Figure 11.1-1 Storage Architecture

AIP Storage

This is the archival storage for FDsys, and is highly protected with the most restrictive security control.

RDF Storage

This is the storage for FDsys RDF, access to this storage area should be limited to the Documentum content servers only and publinkAPI.

ACP Cache Storage

This is to store the published content for public access and delivery, and is accessible to the \web servers for public access. It requires a large capacity of storage (~multiple-TB) and fast access speed.

DAS Storage

The DAS storage is used by all servers for local OS and logs, and by the search servers to store the full text indexes for the search service. By utilizing the DAS storage, the search application can continue to provide more scalable growth at optimal speeds and reduced costs when compared to enterprise NAS storage. The individual search nodes also need access to their portion of the full text index, strengthening the case for local DAS storage instead of enterprise NAS storage.

Storage Network

The Storage Network utilizes dedicated network connections to the VM ESXi hosts to provide enough bandwidth the store and server virtual images and content to the VMs running on the ESXi hosts.

Fiber Channel

The Storage Network utilizes dedicated network connections to the VM ESXi hosts to provide enough bandwidth the store and server virtual images and content to the VMs running on the ESXi hosts.

11.2 Hardware Architecture

The hardware architecture is shown in Figure 11.2-1. It includes major physical server allocations to the applications, and network connectivity layout. There are three virtual LANs, one for access to the database and content management system, one for access to the public-facing web servers, and one for search and application servers for FDsys. The server failover and high-availability are achieved by both techniques of redundant servers and the IP-based load balancer.

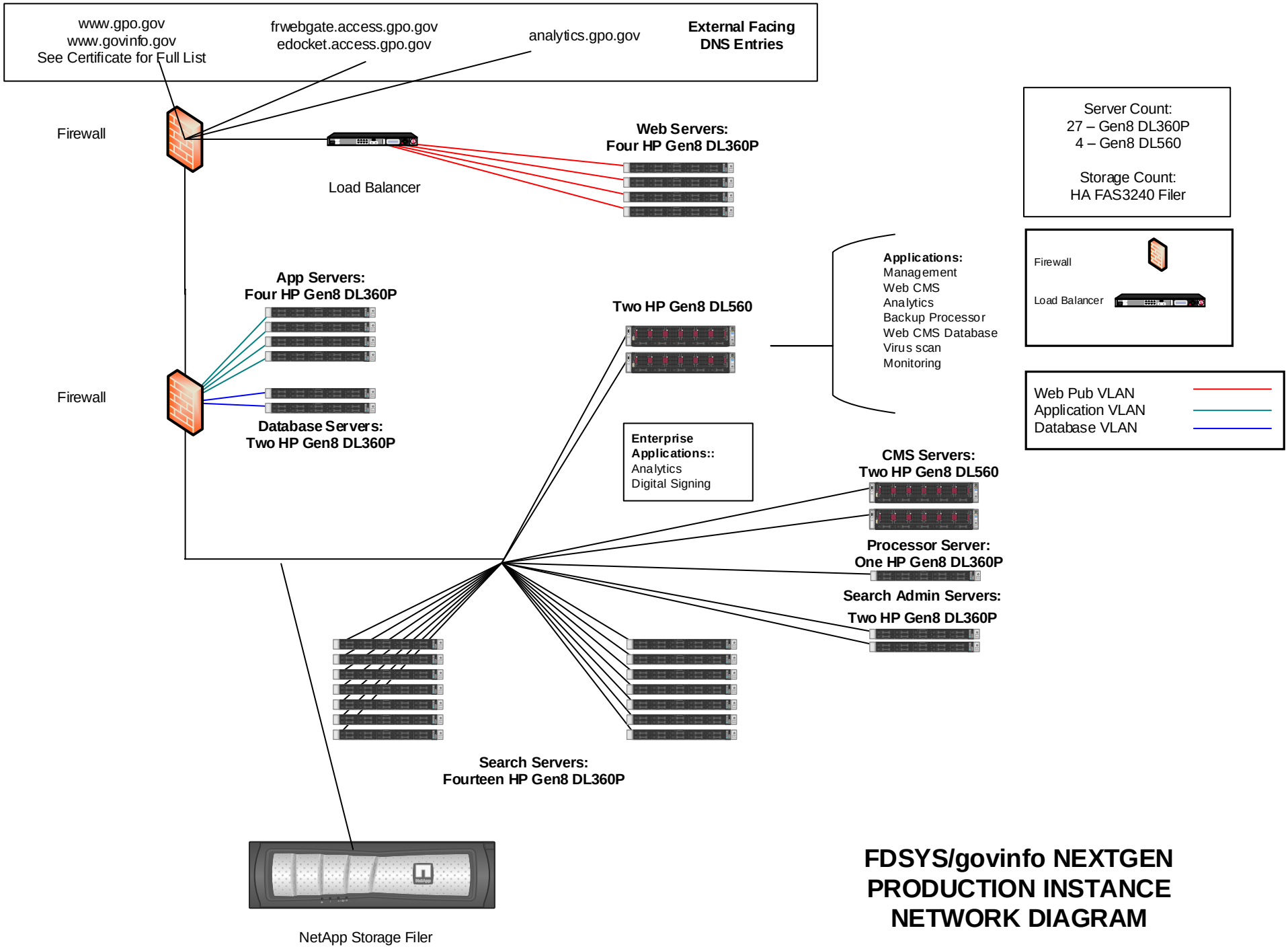

Figure 11.2-1 Hardware Architecture