FUJ00176545

Document Title: Audit Data Retrieval Low Level Design

Document Reference: DEV/APP/LLD/0071

Document Type: Low Level Design (LLD)

Release: N/A

Abstract: This document describes the Audit data extraction and filtering facilities within HNG-X.

Document Status: APPROVED

Author & Dept: Gerald Barnes, Andrew Mansfield

External Distribution:

Approval Authorities:

Name	Role	Signature	Date
Alan Holmes	Customer Solution Architect		

Note: See Royal Mail Group Account HNG-X Reviewers/Approvers Role Matrix (PGM/DCM/ION/0001) for guidance.

FUJ00176545

0 Document Control

0.1 Table of Contents

FUJ00176545

FUJ00176545

0.2 Figures and Tables

FUJ00176545

FUJ00176545

0.3 Document History

Version No.	Date	Summary of Changes and Reason for Issue	Associated Change - CP/PEAK/PPRR Reference
0.1	23/05/2007	Initial Draft	
0.2	26/10/2007	Correction of typographical errors.	
0.3	29/10/2007	Amendments highlighted from reviews	
0.4	09-Nov-2009	Revisions to 5.1.1, 5.1.7, 5.1.9, 6.3.	
0.5	13-May-2010	Server side changes for the Audit Strengthening CP.	CP0336
0.6	14-May-2010	Further server side changes for the Audit Strengthening CP.	CP0336
0.7	14-May-2010	Client side changes for the Audit Strengthening CP.	CP0336
1.0	08-June-2010	Approved version	CP0336

0.4 Review Details

Review Comments by :	28-MAY-2010
Review Comments to :	Gerald.Barnes GRO Andrew.Mansfield GRO RMGADocumentManagement GRO
Mandatory Review	
Role	Name
Solution Design	Alan Holmes
LST	John Rogers
SSC	Steve Parker (*)
SV&I Manager	Chris Maving (*)
Optional Review	
Role	Name
Chief Information Security Officer	Tom Lillywhite
Security and Risk Team	CSPOA.security GRO
System Qualities Architecture	Dave Chapman
Architect	Jason Clark
Business Continuity	Adam Parker (*)
Head of Service Operations	Tony Atkinson
Head of Service Introduction	Phillipa Verity Davies
Service Network	Ian Mills
Data Centre Migration	Vince Cochrane
Integration Team Manager	Peter Okely
LST Manager	Sheila Bamber
RV Manager	James Brett (POL, JTT)
VI & TE Manager	Mark Ascott
Integrity Testing	Michael Welch
Development Manager	Graham Allen

FUJ00176545

Networks Architect (Data Centre)	Mark Jarosz
Development Manager	Adam Spurgeon
Issued for Information – Please restrict this distribution list to a minimum	

(*) = Reviewers that returned comments

FUJ00176545

0.5 Associated Documents (Internal & External)

Reference	Version	Date	Title	Source
PGM/DCM/TEM/0001 (DO NOT REMOVE)	2.0	16-Apr-07	Fujitsu Services RMGA HNG-X Document Template	Dimensions
ARC/SVS/ARC/0001			HNG-X Architecture – Support Services	Dimensions
DES/APP/HLD/0030			Audit Data Collection & Storage High Level Design	Dimensions
TD/ION/011			Audit Server Configuration	PVCS
IA/PRO/004			Audit Data Extraction Process	PVCS
TD/LLD/013			Audit Enhancements for NWB-BI3 Low Level Design	PVCS
ARC/GEN/REP/0001			HNG-X Glossary	Dimensions
SVM/SDM/SD/0017			Security Management Service - Service Description	Dimensions
CR/FSP/006			Audit Trail Functional Specification	PVCS
SD/IFS/014			Audit to Tivoli Cluster Information Interface Specification	PVCS
DES/APP/HLD/0029			Audit Data Retrieval High Level Design	Dimensions

Unless a specific version is referred to above, reference should be made to the current approved versions of the documents.

0.6 Abbreviations

Abbreviation	Definition
ARQ	Audit Record Query
AS	Audit Server
ATR	Audit Track Retriever
ATS	Audit Track Sealer
BRDB	Branch Database
COTS	Commercial off the Shelf
CRUD	Create, Read, Update and Delete
CSV	Comma Separated Variables
DSS	Data Security Standards
EMC	Company that supplies resilient disk technology
FAD	Financial Accounts Department
FS	Fujitsu Services
FTMS	File Transfer Managed Service
HNG	Horizon Next Generation replacing current Baseline Horizon solution
HNG-X	Horizon Next Generation – Plan X
KEL	Known Errors Log
NBS	Network Banking Service
NPS	Network Banking Persistence Service
NBSC	National Business Support Centre
OBC	Operational Business Change
PCI	Payment Card Industry. A set of security controls defined by the Payment Card

FUJ00176545

	Industry organisation.
PO	Post Office
Post Office	Post Office Limited
RAG	Riposte Attribute Grammar
RDDS	Reference Data Distribution System
RDMC	Reference Data Management Centre
RDS	Post Office Reference Data System
RDT	Reference Data Team - the Post Office and Fujitsu Customer Services teams use the RDT environment to validate and verify the reference data associated with business changes.
RFI	Request For Information.
SLT	Service Level Target
SYSMAN	The systems management environment.
TES	Transaction Enquiry Service
TWS	Tivoli Workload Scheduler
XML	Extensible Markup Language

0.7 Glossary

Term	Definition
Baseline Horizon	The existing solution being re-architected
Branch	Post Office outlet identified by a unique Branch Code. Within the HNG model, a Branch is a logical entity that can be composed of several physical locations at which business is transacted.
Fast ARQ	This refers to a simpler interface and more streamlined workflow where branch transaction data is being retrieved. Introduced at release 02.04.
Operational Services	Those services that are needed to run the Horizon system that are not directly supporting the Post Office business. Examples include software distribution, audit, security management etc.
Sensitive Authentication Data	The full contents of any track from the magnetic stripe (on the back of a card, in a chip, etc.), Any data element extracted from the magnetic stripe other than Cardholder Name, PAN, expiry date and service code, and Encrypted PIN blocks.
Slow ARQ	This refers to the functionality of the HNG-X Audit Client application available prior to release 02.04. Accessing this functionality involves a complex interface and requires user interaction at each stage of the process. From release 02.04 onwards, this interface still exists but is supplemented by the 'fast' ARQ interface (see above).

0.8 Changes Expected

Changes
Incorporation of comments from review.

0.9 Accuracy

FUJ00176545

Fujitsu Services endeavours to ensure that the information contained in this document is correct but, whilst every effort is made to ensure the accuracy of such information, it accepts no liability for any loss (however caused) sustained as a result of any error or omission in the same.

FUJ00176545

0.10 Copyright

FUJ00176545

1 Introduction

This Low Level Design expands upon the design documented in the *Audit Data Retrieval High Level Design* (DES/APP/HLD/0029).

Within HNG-X, Fujitsu Services are required to provide facilities to archive, retrieve, filter and present to authorised POL staff, Audit Track data in support of the security policy and audit requirements laid down for the system.

Approximately 90% of the data currently held on the Audit Archive is in the Riposte Attribute Grammar (RAG) format used by the Escher Risposte server.

With the removal of the Riposte Service under HNG-X, the ability to filter, abstract and query this data will be lost, while the requirement to do so will continue to exist for seven years after Riposte is decommissioned.

HNG-X places an additional requirement upon the Audit Extraction toolset to provide filtering, abstraction and querying of the XML formatted data that will be the replacement for the Riposte messages.

Changes identified as arising from the Payment Card Industry (PCI) Data Security Standards (DSS) are also included in this LLD.

This Low Level Design (LLD) specifies the components required to provide the Audit Data Retrieval, filtering, abstraction and querying facilities together with their interfaces and functionality. The level of detail in this LLD is intended to be adequate to enable development, implementation, integration and test work packages to be specified.

The LLD includes details of changes for HNG-X CP0336. As part of the data retrieval process, the archived events generated by counters at the branch are also analysed to identify any possible occurrences of problems which might adversely affect the integrity of the transaction data. HNG-X CP0336 integrates this process into the mainstream Audit retrieval applications and also introduces the fast ARQ interface.

FUJ00176545

2 Scope

This Low Level Design Specification covers:

- Retrieval of data from the Audit Archive.
- Hash checking to ensure data has not been altered.
- Creation, maintenance & monitoring of Audit Record Queries (ARQs)
- Abstraction of messages from Audit Track data.
- Presentation to Auditor, tools to filter and present data in required format
- Interfaces between the Audit components involved in this process.

The data that will be stored in the Audit Archive, and hence the data that needs to be retrieved is defined in *Audit Server Configuration* (TD/ION/011).

The scope of this LLD does not cover:

- Specification of Information Requests, this is defined in *Audit Data Extraction Process* (IA/PRO/004)
- Online access to live data to support Internal Audit
- The analysis/interpretation of Audit Tracks to provide specific Audit Trails

No changes are expected to be made to the core Horizon Audit system components as part of the migration to HNG-X or the implementation of the Payment Card Industry (PCI) Data Security Standards (DSS).

Specific changes that are required for HNG-X which are covered in this document are:

- Replacement facility for the analysis of Horizon Riposte Audit Tracks
- New facility to analyse HNG-X Branch database Audit Tracks
- New facility to handle hashed or encrypted PAN data
- Changes to support server side filtration of Tivoli event data which is integrated in with HNG-X and Horizon branch data retrieval requests (CP0336).

FUJ00176545

3 Design

3.1 Principles

The primary design principle is to ensure that the new and amended HNG-X requirements should be implemented with minimal change to the remaining components of the Audit System.

This dictates that any new components must stand alone, and that synchronous integration with existing components must be kept to an absolute minimum.

Approaching the design in this manner will ensure that the Audit System as a whole continues to adhere to the Audit Architecture as defined in *HNG-X Architecture – Support Services* (ARC/SVS/ARC/0001), and utilises existing components, which are proven to provide only that data that is appropriate to the role of the requestor, wherever possible.

3.2 Legacy Design and Code

The Audit system was designed around two core technologies; Legato tape backup facilities and Riposte.

Legato was replaced as the preferred storage medium prior to HNG-X, although legacy code remains within a number of components. Additionally, the Audit retention policy, which is key to the design of the system, was based upon tape pooling. Removal of the legacy code, and amendments to the basic design are not considered to be cost beneficial.

Audit will continue to hold data in the Riposte Attribute Grammar format for seven years after Riposte has been retired. It is expected that Riposte server will initially be retained to enable both the Horizon and HNG-X solutions to perform data extractions, allowing for verification of results.

FUJ00176545

4 Requirements

The requirements covered in this LLD are:

- Create, maintain, execute and close Audit Record Queries (ARQ)
- Within the 'slow' ARQ framework:
  - Target a specific Audit Server
  - Specify files to be retrieved from Audit Archive based upon Audit Sub-Point and gathered date range.
  - Specify files to be retrieved from Audit Archive by FAD/Branch code.
  - Extract files from the Audit Archive based upon the Audit Sub-Point from which they were initially gathered for a specified date range.
  - Decompress and seal check the retrieved files to ensure data integrity.
  - For Branch Transactions
    - Specify whether Tivoli event data is to be retrieved.
    - For audit tracks delivered in Riposte Attribute Grammar format :
      - Extract messages for specified FAD codes
      - Sequence check messages based upon FAD/Counter sequence number for multiple counters within a FAD
      - Identify gaps in sequences
      - Verify the message based upon the internal CRC value.
      - Convert the RAG formatted message to XLM
    - For audit tracks delivered in XML format
      - Extract messages for specified FAD codes
      - Sequence check messages based upon Branch/Counter sequence number for multiple counters within a Branch
      - Identify gaps in sequences
      - Verify the message based upon the internal validation value.
    - For files in an XML format (either native or converted)
      - Abstract messages based upon a user specified selection criteria
      - Sort messages based upon a user specified sort order
      - Present the abstracted data to an agreed format in Excel, Access or plain text.
  - For all file types
    - Perform PAN number obfuscation and/or encryption to enable searches based upon PCI DSS compliant values
    - Perform string search based on single or multiple user specified strings.
- Within the 'fast' ARQ framework:

FUJ00176545

- Target a specific Audit Server
- Specify files to be retrieved from Audit Archive by FAD/Branch code and gathered date range.
- Specify whether Tivoli event data is to be retrieved.
- Specify data to be abstracted from messages.
- Retrieve, seal check, filter and query files as one continuous operation.

5 Audit Extractor Client Application

5.1 Overview

The Audit Extraction Client application resides upon the Audit workstation and provides a user interface to the extraction, filtering and querying components of the Audit servers.

Implemented as a Microsoft Windows Multiple Document Interface (MDI), the application allows the user to create, execute and monitor multiple ARQs, across multiple Audit servers, simultaneously.

Changes to the status of requests are displayed at the interface, and are periodically refreshed.

The application provides an interface to Audit server components to perform and monitor the following functions within the ARQ framework:

Retrieval

- Specify criteria for retrieval of Audit Track data from the Audit Archive.
- Specify whether event data is to be retrieved.
- Preview list of Audit Tracks matching the retrieval criteria.
- Remove Audit Tracks from the matching list prior to retrieval.
- Instigate the retrieval of selected Audit Tracks.
- Seal checking of retrieved Audit Tracks.

Filtering

For Audit Tracks containing Branch Transaction data

- Specify one or more FAD/Branch codes to be used in filtering.
- Identify Audit Tracks containing matching FAD/Branch codes.
- Abstract messages for the specified FAD/Branch codes.
- Verify integrity of all matching messages
- Verify messages are sequential with no gaps in the sequence

For Audit Tracks containing NBX Transaction journal data

- Specify one or more strings of alpha-numeric characters to match
- Provide PAN obfuscation/hashing and decryption
- Identify Audit Tracks containing matching strings

FUJ00176545

- Abstract lines from Audit Tracks containing matching strings

For Audit Tracks containing event data

- Abstract lines for the specified FAD/Branch code and date range
- Discard benign events.

For Audit Tracks containing any other data

- Specify one or more strings of alpha-numeric characters to match
- Identify Audit Tracks containing matching strings
- Abstract lines from Audit Tracks containing matching strings

Querying

- Only available for Audit Tracks containing Branch Transaction data
- Specify message level selection criteria
- Specify attributes to be abstracted from the message
- Specify sort order
- Apply query to filtered Audit Track data

Presentation

For Audit Tracks containing Branch Transaction data

- Convert filtered/queried data to Excel format
- Present data to a specified directory for writing to CD at the Audit workstation

For Audit Tracks containing any other data

- Present data to a specified directory for writing to CD at the Audit workstation

Not all filtering, and querying options are suitable for all Audit Track formats. It is the user's responsibility to ensure that the retrieved data is suitable for the type of filtering and querying specified.

The fast ARQ interface of the Audit Client Application allows the user to retrieve, filter and query branch data based on a FAD/Branch code and date range. Multiple queries may be selected and the directory to which the extracted data is written can be specified. Optionally, event data may be included.

Once started the fast ARQ process continues through to completion without further user interaction. The process is able to continue after any temporary loss of connection to the Audit Server. After any other errors or exceptions, it will not be possible to continue the ARQ. It will be necessary to close the ARQ and open a new one.

5.2 Security

The Audit Extraction Client application does not contain any explicit security challenges, instead Audit workstations are located in physically secure rooms and require both a valid Windows domain user account and a security ID tag in order to gain access. This is deemed to be sufficient to ensure unauthorised use of the system is prevented.

New users are added to the AWOperators table on the RFISQL database of each Audit server the first time they create an ARQ at the server. This information is used to track ownership of ARQs and any actions requested.

FUJ00176545

5.3 Connecting to Available Audit Servers

Connections to the Audit servers databases are attempted during initialisation of the application, based upon a list of known data centres maintained in the Audit workstation registry.

Upon successful connection a Server object is instantiated and retained for the duration of the user session. The server object maintains a collection of Connection objects for the databases available at that server.

If at any stage during the session, a database or server become inaccessible the appropriate object and ARQs will be marked as inaccessible. Periodic attempts will be made to reinstate the connection.

5.4 System Status Monitoring

The status of the available data centres, and active ARQ's is displayed in the Data Centres form.

The details are displayed in a tree view with data centres acting as root nodes. Active ARQ's are displayed beneath the data centre at which they were created. The ARQ maintenance forms may be accessed by double clicking on a selected ARQ.

The status of a data centre is determined by an attempt to connect to the RFISQL database on the associated Audit server and is indicated by icons representing either active or inactive.

ARQ status, derived from the Status field of the Requests table on the RFISQL database, is likewise represented in iconic form with the following states being uniquely identified.

ARQ Status	Meaning
New	The ARQ has been created, but not submitted.
Retrieving	A retrieval is in progress
Retrieval Failed	Errors encountered while retrieving one or more files
Retrieval Complete	All selected files have been retrieved successfully
Seal Checking	Seal Checking of retrieved files is in progress
Seal Check Failed	Errors encounters while Seal checking one or more files
Seal Check Complete	All files have been Seal checked successfully
Query Pending	Indicates that the ARQ has a valid query awaiting execution
Filtering	Applying filter criteria to extracted files
Filtering Failed	An error was encountered with applying filter criteria
Filtering completed	Extracted files filtered without errors
Querying	Applying query or queries to filtered data
Query Failed	An error was encountered while applying queries
Query Complete	The querying of filtered data completed without errors
Sorting	Sorting the output from the queries
Sorting Failed	An error was encountered while sorting the queried data
Sorting Complete	Queried data sorted without errors
Formatting	Sorted data is being formatted for presentation
Format Failed	An error was encountered while formatting the data
Completed	All requested action completed successfully

Table 1 – ARQ Status definitions

FUJ00176545

5.5 Slow ARQ Forms

5.5.1 Creation of a New ARQ

5.5.1.1 Description

The creation of a new ARQ requires user supplied data to be captured in two stages.

Initially the data centre from which the data is to be retrieved must be selected. The connection to the Audit server at the selected data centre is then validated by attempting a connection to the RFISQL database using the Connection object created during application initialisation.

Upon successful validation of the connection, the user is prompted via the New ARQ form to supply the following data:

Requester

Name of the requesting party. Selected from the Requesters table of the RFISQL database on the Audit server previously selected for the ARQ.

Date Received

Date on which the request was received by the Audit staff.

Date Required

Date by which a response is required

Catalogue Entry

Not used

Receipt Reference

A manually entered reference for the ARQ supplied by the requester

Access Reason

A description of the reason for the system access.

Once the form has been completed, the user has the option to save the ARQ or to specify the retrieval criteria.

Saving the ARQ results in the following actions:

- Validation of input data:

  Date fields are valid dates

  Date Required is greater than Date Received

- A row with a CmdUsedId value of 8 is created in the FileFoundConsole table on the RFISQL database of the Audit server at the data centre to which the ARQ has been assigned. The creation of this row triggers the spCreateFileFoundConsole stored procedure, which spawns the ExtractorLink executable at the Audit server. The ExtractorLink executable in turn spawns the

FUJ00176545

CreateDir executable which creates the ARQ directory structure in the Audit Servers USERAREA.

- A new row is created in the Requests table on the RFISQL database of the Audit server at the data centre to which the ARQ has been assigned.

- A unique RFIReference value is generated from the prefix and NextRFIRef field values of the Requesters table row relating to the ARQ requester, and a single character representing the data centre.
- The NextRFIRef field value for the selected requester is incremented.

Opting to specify the retrieval criteria will initiate an ARQ save, and upon completion will present the user with the Maintain ARQ form populated with the details of the new ARQ.

The user may cancel the request at any time. This will unload the form with no further actions.

5.5.1.2 Interface Definition

Type	Display Name	Source	I/O	Description
combo	Requester	Requesters (RFISQL)	I/O	List of requesters from the [RFI Requester ID] column.
text	Date Received	User	I	Date upon which the request was received
text	Date Required	User	I	Date by which the response is required
text	Catalogue Entry	User	I	Not used
text	Receipt Reference	User	I	Requester specified reference
text	Access Reason	User	I	Reason for the request

5.5.2 Opening an Existing ARQ

5.5.2.1 Description

The Open ARQ form presents a list of all active ARQs by the data centre to which they are assigned, allowing the user to select one for maintenance.

During the loading phase of the form a “Data Centre” drop down combo box is populated with all active data centres from the Servers collection created during the application initialisation.

Selecting a data centre from those available in the “Data Centre” combo will populate the “Query Reference” drop down combo with a list of ARQs active at the selected data centre.

The list of active ARQs is obtained from the Requests table of the RFISQL database at the selected data centre. Only active ARQs will be present on this table, so all rows should be selected.

Selection of an ARQ from the “Query Reference” combo results in the population of the “Access Reason” multi-line text box with the value of the [Access Reason] field of the corresponding ARQ row on the Requests table of the RFISQL database.

Options exist to browse or open the selected ARQ. These are only available when an ARQ is selected.

FUJ00176545

If the browse option is selected, a read-only "Query Details" form is displayed, populated with the ARQ details from the Requests table.

If the Open Query option is selected, a check is performed to ensure that an instance of the "Maintain ARQ" form for the selected ARQ does not already exist. If an existing instance is found it is moved to the front of the screen order. Otherwise a new "Maintain ARQ" form is opened, populated with the ARQ details.

5.5.2.2 Interface Definition

Type	Display Name	Source	I/O	Description
Combo	Data Centre	Servers Object		Displays list of all active data centres
Combo	Query Reference	Requests table		Displays a list of active ARQs from the Requests table on the RFISQL database at the selected data centre.
Text	Access Reason	Requests table		Read only display of the [Access Reason] field value from the Requests table on the RFISQL database at the selected data centre.
Button	Show Query Details			Displays a read only instance of the Query Details form populated with details from the Requests table on the RFISQL database at the selected data centre.
Button	Open Query			Checks for instances of the "Maintain ARQ" form for the selected ARQ. If none found creates a new instance of the "Maintain ARQ" form passing the ARQ reference to the instance.

5.5.3 Maintaining an ARQ

5.5.3.1 Description

The Maintain ARQ form is a multi-tabbed form which encapsulates the following functionality:

- Displays ARQ details and status
- Specification of Retrieval Criteria
- Facilitates:
    - Identification of matching Audit Tracks
    - Exclusion of selected Audit Tracks
    - Retrieval of Audit Tracks
    - Seal Checking of Audit Tracks
- Specification and application of Filtering criteria
- Specification and application of Queries
- Displays progress of filtering
- Display progress of querying
- Formatting data for presentation

FUJ00176545

Multiple instances of this form may exist for multiple ARQs across multiple data centres. However, only one instance may exist for a specific ARQ.

This is achieved by exposing the ARQ reference as a public property of the Maintain ARQ form which may be checked by requests to open an ARQ.

Upon loading, the form receives the Server object relating to data centre at which the ARQ was created and the ARQ reference, and all tabs are populated.

The following access restrictions are enforced:

ARQ Status	Available tabs.					
	ARQ Details	Retrieval Criteria	Audit Tracks	Filtering	Query	Presentation
New	View only	Updatable	View only	Updatable, No execute	Updatable, No execute	View only
Retrieving	View only	View only	View only	Updatable, No execute	Updatable, No execute	View only
Retrieval Failed	View only	Updatable	View only	Updatable, No execute	Updatable, No execute	View only
Retrieval Complete	View only	Updatable	Updatable	Updatable, No execute	Updatable, No execute	View only
Seal Checking	View only	View Only	View only	Updatable, No execute	Updatable, No execute	View only
Seal Check Failed	View only	Updatable	Updatable	Updatable, No execute	Updatable, No execute	View only
Seal Check Complete	View only	Updatable	Updatable	Updatable, Executable	Updatable, Executable	View only
Query Pending	View only	View only	View only	View only	View only	View only
Filtering	View only	View only	View only	View only	View only	View only
Filtering Failed	View only	View only	View only	Updatable, Executable	Updatable, Executable	View only
Filtering completed	View only	View only	View only	Updatable, Executable	View only	View only
Querying	View only	View only	View only	View only	View only	View only
Query Failed	View only	View only	View only	View only	Updatable, Executable	View only
Query Complete	View only	View only	View only	View only	View only	View only
Sorting	View only	View only	View only	View only	View only	View only
Sorting Failed	View only	View only	View only	View only	Updatable, Executable	View only
Sorting Complete	View only	View only	View only	View only	View only	View only
Formatting	View only	View only	View only	View only	View only	View only
Format Failed	View only	View only	View only	View only	View only	View only
Completed	View only	View only	View only	View only	View only	Executable

Table 2 – Client application Maintain ARQ execution restrictions

5.5.3.2 Interface Definition

Type	Display Name	Source	I/O	Description
Tab	ARQ Details			ARQ details and summary status information.
Tab	Retrieval Criteria			Maintain retrieval criteria
Tab	Audit Tracks			Matching Audit Track details and functions
Tab	Filtering			Specification of filtering criteria
Tab	Validation and Query			Results of validation. Query maintenance, validation and execution.
Tab	Presentation			Query output preparation

FUJ00176545

5.5.3.3 ARQ details

5.5.3.3.1 Description

This tab is essentially the same as the New ARQ form, with the exception that it also displays the ARQ current status and action list.

The current Status of the ARQ is from the status field of the Requests table on the RFISQL database.

The action list is retrieved from the FileFoundConsole table on the RFISQL database.

This tab is read only. No amendments may be made to the ARQ details by the user once the ARQ has been created.

5.5.3.3.2 Interface Definition

Type	Display Name	Source	I/O	Description
Text	ARQ Reference	Requests	O	Reference of the ARQ
Text	Requester	Requests	O	Requester reference
Text	Requested Date	Requests	O	Date ARQ requested
Text	Requester Ref.	Requests	O	Reference supplied by the requester
Text	Extract Status	Requests	O	Current status of ARQ
Text	Abstraction Status	QueryRequest	O	Abstraction status of ARQ
Grid	ARQ Actions	Requests	O	List of actions performed under the ARQ

5.5.3.4 Retrieval Criteria

5.5.3.4.1 Description

The criteria specified determines which Audit Tracks are to be retrieved from the Audit archive.

The list of retrieval criteria for an ARQ is maintained in the RepeatableSearchCriteria table, with each Audit Point/Sub-Point, FAD Code or filename template combination on a separate row.

The criteria consist of:

- A single data range during which the Audit tracks were gathered by the Audit system.
- One or more Audit point/sub-point combinations
- Optionally, a FAD code
- Optionally, a file name template.

There is no limit imposed on the number of days that may be included in an extract. However, if the date ranges specified exceeds 31 days, a warning to this effect will be displayed to the user.

Clicking the Select button displays a form that allows Audit Points/Sub Points to be selected from drop down lists populated from the RFISQL database.

FAD codes, when supplied, indicate that the user wishes to have the relevant Audit point/sub-points containing transaction data for the specified outlet, determined automatically. If a FAD code is specified at this stage it will be used for filtering the data should it be required. It will also prevent the introduction of any other filter values.

FUJ00176545

Audit Point/Sub-Point combinations may be determined by inserting a row into the FileFoundConsole table on the RFISQL database with the following values:

RFIReference - Current ARQ reference

CmdUsed - 7

StartAt - Retrieval start date

EndAt - Retrieval end date

GeneratedBy - User name

The returned list of Audit sub-points is returned to the ErrMessage field of the FileFoundConsole row which initiated the request.

File name templates are supplied as free format text. Wild card comparisons may be made using the "*" character.

When retrieving NBX Audit Tracks, and the PAN is in clear, the file template name can be generated via the NBX File Template Generator.

If the Search for files option is activated the retrieval criteria is used to generate an SQL query for execution against the [Initial Track table] table on the SEALERSQL database of the targeted Audit server. Rows matching the selection criteria are copied to the [Active Files] table of the RFISQL database, and referenced to the ARQ.

The selection criteria will be applied as follows:

[Date generated] >= Retrieval Start Date **and** [Date generated] <= Retrieval End Date

and

[Audit Track] like '%*_AuditPoint_AuditSubPoint_*%'

And optionally

[Audit Track] like File Name Template

5.5.3.4.2 Interface Definition

Type	Display Name	Source	I/O	Description
Date	From Date	RepeatableSelectCriteria	I	Start date for the retrieval
Date	To Date	RepeatableSelectCriteria	I	End date for the retrieval
Grid	Audit Point and FAD Details	RepeatableSelectCriteria	O	4 column grid containing Audit Point, Audit Sub-Point, FAD Code and FAD Hash
Button	Select			Allows Audit Points/Sub Points and FAD Codes to be added to the selection criteria
Text	Horizon Template	RepeatableSelectCriteria	I	Template against which Horizon Audit Track names are matched
Button	HNG-X Template			Template against which HNG-X Audit Track names are matched
Button	Search for files			Saves criteria to RepeatableSelectCriteria table, and populates ActiveFiles table with matching records from the [Initial Track table] table on the SEALERSQL database

FUJ00176545

5.5.3.5 Audit Tracks

5.5.3.5.1 Description

This tab displays the full list of Audit tracks for the ARQ contained in the ActiveFiles table, and applies the requested actions only to those Audit tracks marked as selected on the displayed list.

It should be noted that this list should only contain Audit Tracks that are required to fulfil the ARQ.

If the list contains Audit Tracks that are not retrieved or Seal checked it will not be possible to progress to the filtering tab.

This tab provides access to the following Audit Track related functionality:

Restore Files

Retrieves files from the EMC Centera.

A row for each selected file is inserted into the FileFoundArgument table, and a call to the ExtractorLink.exe server component instigated via the insertion of a FileFoundConsole row with the following values:

RFIReference = Current ARQ reference

CmdUsedID = 1

Files are retrieved to the RESTORED_AT directory of the ARQ. This exposes the files to the Retriever core component which will move the file to the ARQ's RETRIEVED_AT directory and create a Sealer marker file, signalling Sealer to begin Seal generation and comparison against the original seal value.

Upon completion of the retrieval the status, and error code of the FileFoundConsole row, and the status field of the Requests row will be updated accordingly.

Replace Files

Replaces the ActiveFiles rows related to the ARQ with a set or rows consisting of only those Audit Tracks selected in the Audit Track Grid.

Delete Files

Deletes rows relating to select Audit Tracks in the Audit Track Grid from the ActiveFiles table for the current ARQ.

Confirm Seal Status

Executes the stp_FileStatusAction stored procedure at the RFISQL database, which updates the ActiveFiles table with the current status of each file.

FUJ00176545

5.5.3.5.2 Interface Definition

Type	Display Name	Source	I/O	Description
Grid	Audit Track list	ActiveFiles	I/O	List of all currently selected Audit tracks for the ARQ
Checkbox	Select All		I	Selects all of the rows in the Audit Track List
Button	Restore Files			Retrieves selected files from the Centera
Button	Replace Files			Replaces files in the ActiveFiles table with selected Audit Track data
Button	Delete Files			Deletes selected rows from the ActiveFiles table
Button	Confirm Seal Status			Update current status of each file

5.5.3.6 Filtering

5.5.3.6.1 Description

This tab displays data relating to the filtering of retrieved Audit tracks, and includes filtering type, output type and the definition of filtering criteria.

Filtering criteria is linked to the ARQ via the RFIQueryRequest table, therefore the first action performed on this tab should check for the existence of a RFIQueryRequest row for the ARQ, and if not found create a new row for the ARQ with a status of “New”.

Two filtering methods exist, string based and message based.

The method which is displayed on the filtering tab will depend upon the combination of Audit Sub-Points selected in the retrieval criteria.

If the retrieval criteria contains only Audit Sub Points from which RAG or Branch Database XML messages are gathered, the message based filtering interface will be displayed.

All other combinations will result in the string based filtering interface being shown.

Filtration criteria, irrespective of the filter type are held in the RFIQueryRequestFilter table.

5.5.3.6.2 Message Filter Type

The Start and End date text boxes will be enabled, and the PAN Generator button hidden.

If a FAD/Branch code was specified during the retrieval of Audit Track data, this will be used as the filter criteria and no additional codes may be added. The FAD/Branch code(s) specified in the RepeatableSelectCriteria for the ARQ will be inserted in to the RFIQueryRequestFilter table.

Additional FADs may be included in the filtration by entering the code in the FAD/Branch code text box and selecting “Add”

Optionally a start and end date may be specified for the filtration. If none is present the entire date range contained within the extracted files will be utilised.

Available Output Type Options :

The output type option is fixed for message filtering.

Abstract - Messages with matching FAD/Branch codes will be abstracted.

FUJ00176545

Executing the filter, via the Apply Filter button will perform the following:

Clear rows for the ARQ from the RFIQuerySequence table

Clear rows for the ARQ from the RFIQuerySequenceGap table

Set the RFIQueryRequest status to “Pending”

The results from validating the extracted data will be displayed in summary on the ’Validation and Query’ tab.

5.5.3.6.3 String Filter Type

If a string filter type is selected the QueryType field of the RFIQueryRequestFilter row will be set to “STR”.

Search strings may be added to the RFIQueryRequestFilter table by entering the string in the “Search String” text box and selecting “Add”.

Alternatively the “PAN Generator” option may be selected. This will load the PAN Generator form, passing the ARQ Reference and the current form and tab references. PAN data entered or created at the PAN Generator form will be added to the RFIQueryRequestFilter table and the tab view refreshed.

Filtering by string is case sensitive. This is necessary to ensure correct matching of encrypted PAN values.

Available Output Type Options

File - Files containing one or more of the match strings will be extracted

Abstract - Lines containing one or more of the match strings will be abstracted.

Executing the filter, via the Apply Filter button will perform the following:

Clear rows for the ARQ from the RFIQuerySequence table

Clear rows for the ARQ from the RFIQuerySequenceGap table

Set the RFIQueryRequest status to “Pending”

String search results are not validated.

5.5.3.6.4 Special Considerations for NBX Data

It is anticipated that the implementation of PCI DSS the availability of “in clear” PAN data will be diminished. While the ability to filter NBX Transaction journals by the PAN hash value in the file name template is retained, it is anticipated that for obfuscated and encrypted PANs further filtration will be required.

In these circumstances all NBX data for the required period would be retrieved and a filter of Type “String” with an output type of “File” would be declared.

Filter search strings for Clear PAN, obfuscated PAN and Encrypted PAN could either be entered manually, or when a Clear PAN is available generated via the PAN Calculator form.

5.5.3.6.5 Interface Definition

It should be noted that the Filter Start and End Dates are not available on the String filter interface. They are however required on the RFIQueryRequest table.

For string filtering these values will be set to the date and time that the RFIQueryRequest row is created.

FUJ00176545

The server-side filtering process does not make use of these dates for string filtering.

Type	Display Name	Message	String	Source	I/O	Description
Combo	Output type	X	X	Internal	I	File or Abstract. The value is fixed as Abstract for Message filtering.
Date	Filter Start Date	X			I	Start date for message filtration
Date	Filter End Date	X			I	End date for message filtration
Text	Free Format String		X		I	User entry of free format string
List box	String List		X	RepeatableSelectCriteria	O	List of current string filtration criteria
Button	Add		X			Updates the RFIQueryRequestFilter with the string specified in the Free Format string text box. Rebuilds the String list from the RFIQueryRequestFilter table.
Button	Add PAN		X			Displays the PAN Generator
Button	Delete		X			Deletes entry selected in the String List from the RFIQueryRequestFilter. Rebuilds the String list from the RFIQueryRequestFilter table.
Text	FAD/Branch Code	X			I	User entry of FAD Code
List Box	FAD List	X		RepeatableSelectCriteria	O	List of current filtration criteria from the RFIQueryRequestFilter
Button	Add	X				Adds the FAD/Branch code or search string to the RFIQueryRequestFilter table. Rebuilds the FAD list from the RFIQueryRequestFilter table.
Button	Delete	X				Removes the entry selected in the FAD List from the RFIQueryRequestFilter table. Rebuilds the FAD list from the RFIQueryRequestFilter table.
Button	Save Filter					Updates the RFIQueryRequest.
Button	Apply Filter					Executes the filter

5.5.3.7 Validation and Query

5.5.3.7.1 Description

Displays the sequence validation details and allows the selection and execution of an XQuery query.

The number of queries required to satisfy the ARQ is determined from the Audit Sub Points included in the extract. A comparison with the RFIAuditSubPointExtended table will indicate all data sources covered by the extracted data.

Where both Horizon and HNG-X data has been extracted, two queries will be required. Where a single data source is involved only one query will be required.

If no RFIQuery rows exist for the ARQ, the first action performed in the specification of the query should determine the number of queries required by the ARQ, and create the RFIQuery rows linking them to the ARQ and appropriate data source.

Only the data source combo box is initially available on the Validation and Query Interface.

A data source must be selected to activate the Sequence Valiation, Select Query and Execute Query sub-tabs.

FUJ00176545

5.5.3.7.2 Sequence Validation

Displays the result of the message filtering sequence checking.

Results are displayed in FAD/Node/lowest sequence number order.

If gaps are identified the size of the gap is displayed in the Size of Gaps column.

5.5.3.7.3 Select Query

The selection of a query is a two stage process:

Retrieve

The Retrieve button displays a dialogue containing a list of queries from the //*Selected_Server*/USERAREA/CommonQueries directory, where "*Selected_Server*" is the server at which the ARQ was created.

Selecting the query does not load the query to the client interface.

Open

Opening the selected query copies the contents of the query file selected during retrieval to the user interface.

The query is displayed in a non-editable text box.

Double clicking on the text box displays the query in an editable textbox. Changes made can then be accepted or cancelled.

5.5.3.7.4 Save and Save AS

The query as displayed in the non-editable text box may be saved to both Audit servers using the name shown in the Query text box.

Alternatively the Save As option may be used to change the query name.

Selecting Save As displays a dialogue listing all of the queries currently held in the //*Selected_Server*/USERAREA/CommonQueries directory, where "*Selected_Server*" is the server at which the ARQ was created.

5.5.3.7.5 Execute Query

This interface exposes a single button and a list box.

The button instigates Execution of the selected and opened query.

The contents of the Query display text box is copied to the USERAREA*ARQ_DIR* of the server at which the ARQ was created.

The file containing the query will be named according to the data source against which it is to be executed. Horizon queries will be named Hz_*xxx*.xql, HNG-X queries, Hx_*xxx*.xql.

Once the file is created, the selected query and query request (RFIQuery, RFIQueryRequest) statuses are set to values of 1 (New).

FUJ00176545

5.5.3.7.6 Interface Definition

Type	Display Name	Source	I/O	Description
Combo	Data Source		O	Display all data sources for which data has been extracted
Text	Status			QueryRequest status

Sequence Validation

Type	Display Name	Source	I/O	Description
List	Node		O	FAD/Node
List	Lowest Message		O	Lowest message number in a sequence
List	Highest Message		O	Highest message number in a sequence
List	Size of Gaps		O	Size of Gap found in sequence.

Select Query

Type	Display Name	Source	I/O	Description
Text	Query			Name of the query to be executed
Button	Retrieve			Displays a list of available queries for selection
Button	Open			Opens and retrieves the selected query to the client interface
Button	Save			Save the query as displayed at the client interface to the file named in the Query Text box.
Button	Save As			Displays a dialogue to enable the query to be saved to a different file.

Execute Query

Type	Display Name	Source	I/O	Description
Button	Execute			Instigates execution of the query
List	Actions		O	Displays action performed at the server while executing this query.

5.5.3.8 Predefined Queries

A set of predefined queries will be maintained on both Audit servers. These queries implement the queries that were available under Horizon in the FLoWR query language.

H[xz]_AuthCds.xql	Query based upon the IOP query which also shows banking agent response information.
H[xz]_BQIOPPANBarcodes.xql	Query based upon the IOP query which also shows any banking agent C2 messages, PAN and client account references.
H[xz]_Bureau.xql	Query based upon the IOP query which also shows details of Bureau transactions.
H[xz]_Events.xql	Basic events query. Shows all counter events generated at a branch.
H[xz]_IOP.xql	Basic branch transactions query. Shows basic information on

FUJ00176545

	transactions recorded at a branch.
H[xz]_IOP_TITO.xql	Query based upon the IOP query which also shows additional information relating to transferring in and out.
H[xz]_IOPDVLA.xql	Query based upon the IOP query which also shows additional information relating to DVLA transactions.
H[xz]_IOPDVLAAdditional.xql	Query based upon the IOPDVLA query which also shows yet further additional information relating to DVLA transactions.
H[xz]_IOPMLabel.xql	Query based upon the IOP query which also shows additional information relating to Mails labels.
H[xz]_IOPPANBarcodes.xql	Query based upon the IOP query which also shows additional information relating to Pan and client account references.
H[xz]_IOPMailService.xql	Query based upon the IOP query which also shows additional information relating to Mails service transactions.
H[xz]_IOPMailServiceSDAddress.xql	Query based upon the IOP query which also shows additional information relating to the destination address of Mails transactions.
H[xz]_RemittancePouches.xql	Query which shows the barcodes of all remittance pouches delivered to, or collected from a branch.
H[xz]_SecurityEvents.xql	Query based upon the Events query which also shows additional security information.
H[xz]_Signons.xql	Query based upon the Events query which only shows details of sign-ons & signoffs.
H[xz]_StockDeclarations.xql	Query which shows details of all stock declarations made at a branch
H[xz]_IOPPouchId.xql	Query based upon the IOP query which returns an additional field containing a pouch id.

5.5.4 NBX File Template Generator

5.5.4.1 Description

This form is loaded from the Maintain ARQ – Retrieval Criteria tab.

At loading a reference to the calling form and the ARQ data centre name are passed to the NBX File template generator, and the read only Data Centre text box is populated with the name of the data centre.

The PAN, in clear, must be supplied.

There are two possible methods for generating the filename template, designated Type One and Type Two.

If the “Generate Template” option is selected a file template name is generated as follows:

FUJ00176545

*FN01_NBX_*_*_NPSJNL*Mod64PAN_*_*_V001.arc*

where

Mod64PAN is the result of (PAN % 64) +1 for a Type One template

Or

Mod64PAN is the result of (RIGHT(PAN, 3) % 64) +1 for a Type Two template.

Selecting the Update ARQ option copies the file name template to the file name template on the Retrieval Criteria tab of the calling form.

5.5.4.2 Interface Definition

Type	Display Name	Source	I/O	Description
Text	PAN		I	List of all currently selected Audit tracks for the ARQ
Text	Data Centre		I	Retrieves selected files from the Centera (Read Only)
Combo	Template Type		I	Values “Type One” or “Type Two”
Text	File name template		O	File name template. (Read only)
Button	Generate Template			Generates the file template name based upon the PAN and data centre.
Button	Update ARQ			Deletes selected rows from the ActiveFiles table

5.5.5 Pan Generator

5.5.5.1 Description

This form may be called from the Maintain ARQ Filter or Query tabs.

When loaded the ARQ reference, and references to the calling form and tab are passed to the PAN Generator form.

A PAN may be entered either in clear or encrypted form. Selecting the “Generate PANs” option will generate all possible PAN versions.

Entered PAN	Generated PAN		
	Clear	Obfuscated	Encrypted
Clear		Yes	No
Obfuscated	No		No
Encrypted	Yes	Yes	

Table 3 – PAN Generation matrix

PAN Activity is logged to the RFIPANDecryptLog table of the RFISQL database.

FUJ00176545

5.5.5.2 Interface Definition

Type	Display Name	Source	I/O	Description
Text	Clear PAN		I/O	PAN in clear form
Text	Obfuscated PAN		I/O	PAN in hashed and obfuscated form
Text	Encrypted PAN		I/O	PAN in Encrypted form
Button	Generate PANs			Generates possible PANs from the input version
Button	Update ARQ			

5.5.6 Presentation of data

5.5.6.1.1 Description

At the current release, query results can only be formatted as Microsoft Excel spreadsheets. After selecting the checkbox, an output file name can be supplied and the Create Output button clicked.

5.5.6.1.2 Exporting to Excel

Excel workbooks will contain at least two worksheets.

Starting at sheet one, the query results data will be exported to Excel, with attribute names used as column titles. If attribute aliases exist, these should be used in preference.

If the number of rows to be exported exceeds the number of rows permissible in an excel spread sheet, additional sheets are added to the workbook and populated.

When all data has been exported a final worksheet is added containing the following query summary information.

- FAD/Branch code

For each counter:

- Lowest Message
- Highest Message
- Size of Gaps
- Time Issues

5.5.6.1.3 Interface Definition

Type	Display Name	Source	I/O	Description
Check	Export to Excel		I	Export the query output to Excel
Text	Excel Output File	User input	I	Excel File name to which data is exported
Button	Change			Change Excel workbook path/filename
Check	Export to Access			Not available at this release
Text	Access Database			Not available at this release
Button	Change			Not available at this release
Check	Export to Text file			Not available at this release

FUJ00176545

Text	Text File			Not available at this release
Button	Change			Not available at this release
Button	Create Output			Perform the export to Excel format

5.6 Fast ARQ Form

5.6.1 Description

The creation of a fast ARQ requires user supplied data to be captured in two stages.

Initially the data centre from which the data is to be retrieved must be selected. The connection to the Audit server at the selected data centre is then validated by attempting a connection to the RFISQL database using the Connection object created during application initialisation.

Upon successful validation of the connection, the user is prompted via the Fast ARQ form to supply the following data:

Requester

Name of the requesting party. Selected from the Requesters table of the RFISQL database on the Audit server previously selected for the ARQ.

Receipt Ref.

A manually entered reference for the ARQ supplied by the requester

Date Received

Date on which the request was received by the Audit staff.

Date Required

Date by which a response is required

Filter Start Date

Start date for which messages are to be filtered.

Filter End Date

Last date for which messages are to be filtered.

Extra days for extraction

Number of extra days to be added to the Filter End Date to give the last extraction date.

FAD Code

Branch identifier of Post Office for which messages are to be retrieved.

Include Events

FUJ00176545

Checked if events data is to be extracted and filtered.

Drive and folder for output

Specifies a folder on the Audit Workstation to which the output files are copied.

Queries to be run

Selection of zero or more queries to be run against the filtered message data.

Once the form has been completed, the user clicks the 'Execute ARQ' button to start execution of the ARQ. This results in the following actions:

- Validation of input data:

  Receipt Reference entered

  Date Required is greater than Date Received

  Filter End Date is greater than Filter Start Date

  Six digit FAD Code entered

- A new row is created in the Requests table on the RFISQL database of the Audit server at the data centre to which the ARQ has been assigned.

- Selection criteria are saved in the RFIQueryRequest and RFIQuery tables on the RFISQL database of the Audit server at the data centre to which the ARQ has been assigned.

- A row with a CmdUsedId value of 8 is created in the FileFoundConsole table on the RFISQL database of the Audit server at the data centre to which the ARQ has been assigned. The creation of this row triggers the spCreateFileFoundConsole stored procedure, which spawns the ExtractorLink executable at the Audit server. The ExtractorLink executable in turn spawns the CreateDir executable which creates the ARQ directory structure in the Audit Servers USERAREA.

- The ARQ progresses through the retrieval and seal checking of audit files, filtering and querying of the extracted data, and generation of the final output files without further user interaction.

Progess is reported on the status bar of the Fast ARQ form.

If the connection to the RFISQL or SEALERSQL databases is lost, the fast ARQ process is able to continue automatically when the database connection is restored.

The fast ARQ process can not recover from any other types of error. When these are reported, via message boxes or the status bar, the user must close the ARQ and create a new ARQ after resolving the problem.

FUJ00176545

5.6.2 Interface Definition

Type	Display Name	Source	I/O	Description
combo	Requester	Requesters (RFISQL)	I/O	List of requesters from the [RFI Requester ID] column.
text	Receipt Ref.	User	I	Requester specified reference
text	Date Received	User	I	Date upon which the request was received
text	Date Required	User	I	Date by which the response is required
text	Filter Start Date	User	I	Start date for message filtration
text	Filter End Date	User	I	End date for message filtration
text	Extra days for extraction	User	I	Number of days to add to Filter End Date to calculate the end date for extraction
text	FAD Code	User	I	Branch identifier
checkbox	Include Events	User	I	Checked if event date to be included
listbox	Select drive and folder for output	User	I/O	Selects destination folder for output files
listbox	Queries to be run	User	I/O	Allows selection of multiple queries to be executed
button	Execute ARQ			Saves the request details and executes the ARQ
button	Exit ARQ			Closes the Fast ARQ form without saving the request details.

5.7 Closing an ARQ

5.7.1.1.1 Description

This form enables the closure of an ARQ.

Upon loading, a drop down combo is load with a complete list of available data centre from the Severs collection.

Selecting a data centre populates the ARQ Reference with a list of active ARQs from the Requests table of the RFISQL database at the selected data centre.

Selecting an ARQ populates the Access Reason text box with the contents of the [Access Reason] field of the matching Requests row, and the File List grid with a list of associated Audit Tracks from the ActiveFiles table.

Clicking on the Close Request button inserts a row into the FileFoundConsole table with the following values:

RFIReference = Selected ARQ Reference

CmdUsedID = 12

This will instigate an ExtractorLink execution calling the CloseLogAndTidy component.

5.7.1.1.2 Interface Definition

Type	Display Name	Source	I/O	Description
Combo	Data Centre	Servers global object	O	List of all available data centres

FUJ00176545

Combo	Query Reference	Requests	O	List of all active ARQs at the selected data centre
Text	Access Reason	Requests	O	Description of reason for the access
Grid	File List	ActiveFiles	O	List of files associated with the ARQ from the ActiveFiles table
Button	Close Query			

FUJ00176545

6 Server Components

Server components are divided into two major groupings, Core and Non-core components.

There are a number of considerations that determine whether a component will be considered a core or non-core component, but in general components that are controlled by the Tivoli Workload Scheduler, and are not constrained by the ARQ framework are considered to be core components.

Non-core components are sub-divided into two further categories, Extractor Link components and Filtering and Abstraction components. This division is based upon the method employed to access the components. Extractor Link , components are instantiated via the ExtractorLink executable (See Section 6.2) which resides on the Audit Server.

Many server components perform multiple roles within the audit system and may be utilised in one or more of the gathering, cross-campus and extraction processes. As such their role in the extraction process can not be considered in isolation, and where multiple role functionality is present the components are described in their entirety. Additionally, now redundant functionality, mainly associated with the use of the Legato tape backup system which preceded the introduction of the EMC Centera data repositories, is abundant. This is resultant from the historic requirements to migrate from the tape based system to a disk based system while retaining the ability to regress, and to utilise existing interfaces between components without requiring changes to their definition.

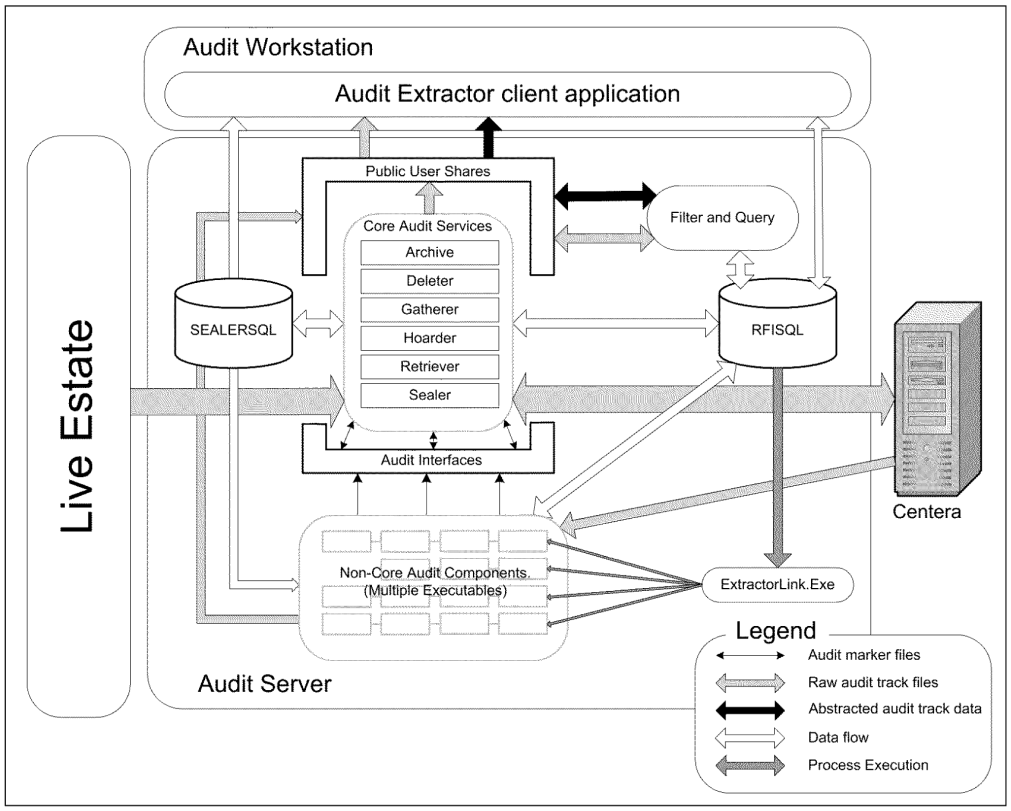

Figure 1 – Interaction of Audit Server Components

FUJ00176545

6.1 Core Components

This section details only those core components that are involved in the extraction process, but includes reference to all component functionality. This includes functionality which is not utilised in the extraction and filtering process, but is essential to the operation of the Audit system as a whole.

6.1.1 Retriever (ATR)

The purpose of the Retriever module is to move retrieved files to the Sealer module for further processing.

Configuration information is taken from the Audit server configuration file.

Only one instance of Retriever may be active on an Audit Server.

The Retriever communicates its progress by generating events and manipulating marker files.

Events are generated by calls to the Logger subsystem DLL. Marker files are manipulated by calls into the AuditInterfaces subsystem DLL.

6.1.1.1 Function

- The Retriever is started by the Tivoli Workload Scheduler and is capable of responding to system messages commanding it to stop, with immediate effect or after finishing the current retrieval activities, as required.
- While it is under the control of the Tivoli Workload Scheduler, it is run constantly as a background process so no scheduling is required.
- The purpose of the Retriever is to pass recovered files into the user's area for access by Sealer for seal checking.
- The Retriever records file activity by calling a sub-system interface function to create a Record File in the Activity Log - maintained by the Audit Track Deleter (ATD), for auditing purposes.
- The Retriever stores all progress indications to persistent storage, so it is capable of completing any operations interrupted by a system malfunction or "stop retrieval" command. This includes ensuring that the Sealer is notified of completion of the retrieval of the file.
- Processing exceptions are noted in the NT application event log via a sub-system interface function. "Normal" events are not logged, as this would quickly fill up the event log.

6.1.1.2 Inputs

All entities marked "*" are generated via sub-system interface calls

1. Audit Track File - for Retrieval from the recover directory
2. * From configuration file:
   - Pass to Sealer flag
3. * System date & time (UTC) for filename timestamps

6.1.1.3 Outputs

1. Audit Track File - for use by Sealer To Check Seal Request Directory

FUJ00176545

2. * Deleter activity log. One entry for each retrieved file, containing:
    - Archive component identifier (Retriever)
    - Time and Date of attempted retrieval
    - A Code indicating Success or Failure
    - The Audit Track Filename

6.1.1.4 Processing

The Retriever component scans the users file store area (F:\USERAREA\<ARQ>\RESTORED_AT) to identify any audit files that have been recovered from the Centera cube.

Retriever checks that it has exclusive access to the file and then creates a new marker file via another sub-system interface. Embedded in the marker file is the full path and file name of the retrieved file that is to be seal checked.

Retriever moves the audit files from the (F:\USERAREA\<ARQ>\RESTORED_AT) directory to the (F:\USERAREA\<ARQ>\RETRIEVED_AT) directory.

Retriever moves the marker file into the Sealer control directory so that the Sealer can do its job.

The Retriever then records file activity by calling a sub-system interface function to create a Record File in the Activity Log.

The process of file retrieval will continue until no audit tracks remain in the Centera recover directory, then Retriever will enter a short sleep before attempting to retrieve the next recovered file.

6.1.1.5 I/O Dependencies

Configuration File

A correctly installed Audit Server Application

6.1.1.6 Configuration File

The following flags are included in the Retriever section configuration file:

PassToSealer = yes/no (Obsolete)

PassToExtractor = yes/no (Obsolete)

These flags still exist in the configuration file but they are no longer used.

6.1.2 Sealer

The Sealer module accepts Audit Tracks from either the Gatherer (ATG) module or the Retriever (ATR) module, and passes them through a secure hashing algorithm known as MD5. The MD5 hashing algorithm produces a 128 bit hash value that is unique to each Audit Track. The 128 bit hash value, the audit track name, as well as other related data are saved in a relational database.

In addition to the hash values generated locally, the Sealer module imports, via FTMS, details of Audit Tracks that have been hashed on the alternate campus Audit Server.

6.1.2.1 Function

FUJ00176545

The Audit Track Sealer (ATS) runs on the Archive Server as a stand-alone process, of which there may be only 1 instance.

It is started by the Tivoli Workload Scheduler (normally) or from Tivoli (exceptionally).

An ATS instance deals with 1 or more Audit Tracks, whose identities are made available to the ATS by the use of an associated Audit Track marker files.

The main function of the ATS instance is to 'Seal' an Audit Track and place the results in a database.

When a file has been successfully sealed, ATS informs the Audit Track Hoarder (ATH) via a filestore interface of the availability of the sealed file. It does so by moving a marker file of the same name as the sealed file into the Hoarders files folder.

It should be noted that the Audit Track Hoarder's cross-campus replication functionality has been incorporated into the Sealer module, although Hoarder still appears to perform marker file manipulation.

When the ATH has been informed as above, ATS calls a function which places an entry in the activity log which is maintained by the Archive Deleter (ATD).

ATS is capable of responding to system messages commanding it to stop sealing, with immediate effect or after finishing current activities, as required.

Processing exceptions are noted in the NT application event log via a standard Archive Server function. "Normal" events are not logged, as this would quickly fill up the event log.

6.1.2.2 Inputs

Data file: Audit track from gatherer,

Data file: Audit track from retriever,

Data file: Seal file from alternate campus seal DB via FTMS,

Data files: locally generated FTMS control & audit files,

Data files: Seal database.

6.1.2.3 Outputs

Data record: Seal data to seal database,

Data file: Seal file to alternative campus seal DB via FTMS,

Data file: Target audit track to hoarder,

Data file: Sealer generated audit track data to deleter (activity log),

Data file: Seal database backup file,

Data files: locally generated FTMS control & audit files.

6.1.2.4 I/O Dependencies

The Sealer module has 4 primary functions. These are:

1. **Seal gathered files,**

   Read Gathered audit track,

   Write Generated seal data to the seal database,

FUJ00176545

	Write	Generated seal data to the alternate campus
	Move	Gathered audit track to Hoarder folder,
	Write	Sealer generated audit data to Deleter.

2. **Seal retrieved files,**

Read	Retrieved audit track
Write	Generated seal data to the seal database,
Write	Generated seal data to the alternate campus
Delete	Retrieved audit track,
Write	Sealer generated audit data to Deleter.

3. **Import seals from the alternate campus Audit Server,**

Read	Imported seal data from the alternate campus,
Write	Seal data to the seal database.

4. **Backup seal database and FTMS control files.**

Read	Seal database,
Write	Seal database,
Read	FTMS generated control & audit files,
Write	FTMS generated control & audit files.

6.1.2.5 Control

The sealer module can be invoked by a command line call of 'sealer.exe', or by using the 'archive.exe' program.

At start up, the configuration file is interrogated to obtain a set of runtime parameters specific to this invocation. The 'sealer' module will then be in a permanent run state that can only be terminated by detection of an error, or by receipt of a Windows message requesting an orderly shut down.

The run time parameters, obtained from the configuration file, determine which of the 4 sealer functions are to be operational, as well as the run time scheduling parameters for each of these functions. All functions can be invoked, controlled and managed independently. However, each configuration file change will require a re-start of the sealer module. Dynamically accessing configuration file changes is not supported.

The four functions supported by the sealer module are:

1. Seal gathered files,
2. Seal retrieved files,
3. Import seal data from the alternate campus Audit Server,
4. Backup the seal database and associated FTMS control and audit files.

Any of these functions, in any combination can be switched on by having the associated configuration file '… … Active' entry = YES. e.g. 'BackupActive=YES'. Should any of these '……Active' lines be set to

FUJ00176545

NO then that function will be inhibited and it can only be invoked by amending the configuration file and re-starting the sealer module to pick up the changed values.

6.1.1.6 Configuration File

The sealer requires the following entries in the configuration file. However, the values shown here are examples only, not definitive.

```
[SchedulingPoint 1]
Mode=CLOCK
StartTime 1=00:00
StopTime 1=23:59

[SchedulingPoint 2]
Mode=CLOCK
StartTime 1=00:00
StartTime 1=16:00
StopTime 1=12:00
StopTime 1=23:59

[SchedulingPoint 3]
Mode=CLOCK
StartTime 1=00:00
StartTime 2=06:00
StartTime 3=12:00
StartTime 4=18:00
StopTime 1=03:00
StopTime 2=09:00
StopTime 3=15:00
StopTime 4=21:00

[SEALER]
GenerationMode=MD5
MaxThreads=10
GatheredScheduling=1
RetrievedScheduling=1
FTMSInScheduling=2
FTMSOutScheduling=2
BackupScheduling=3
GathererActive=YES
RetrieverActive=YES
FTMSInActive=YES
FTMSOutActive=YES
BackupActive=YES
```

6.2 Non-core Components - Extractor Link

Extractor Link components, so named because they are instantiated by the ExtractorLink.exe application, reside upon the Audit Server, and provide asynchronous execution of server side processes instigated at an Audit workstation client.

FUJ00176545

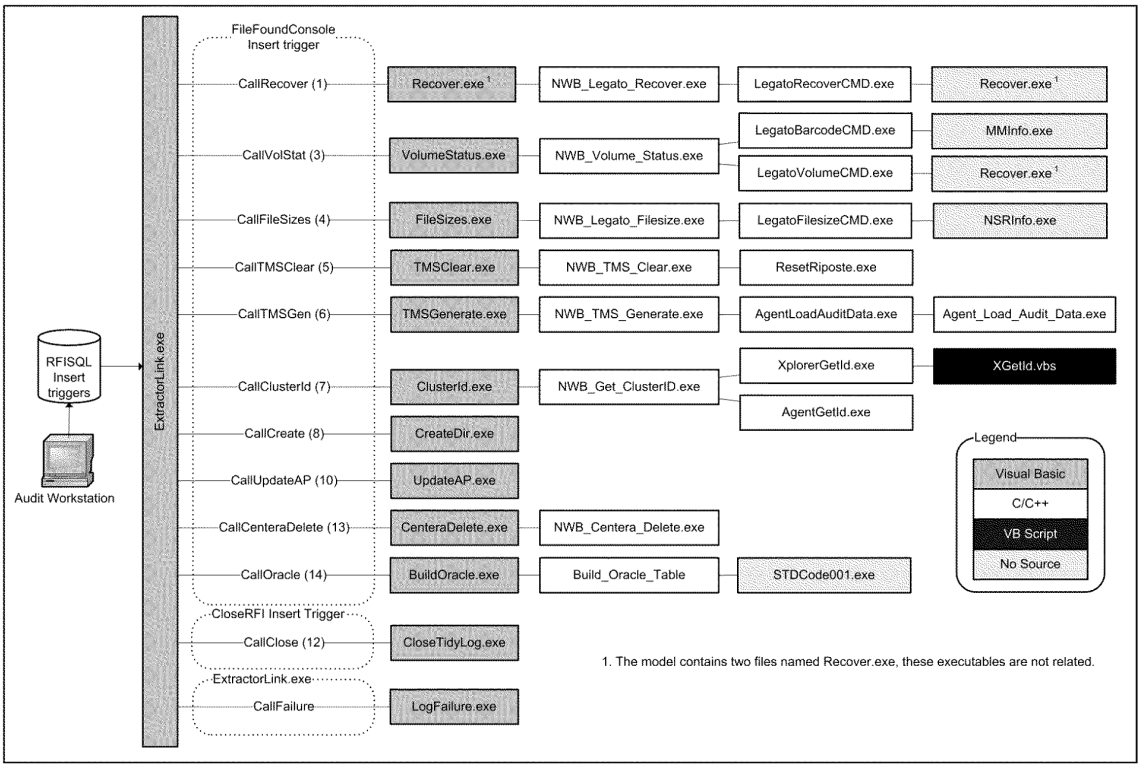

Figure 2 – Extractor Link – Non-Core Components

It should be noted that some of these execution paths, mainly those relating to the generation and management of Transaction Message Stores, will become redundant once the roll out of HNG-X has been completed. Until then they may continue to be used.

A number of these components also contain code relating to the Legato tape archive system, some with execution branches to facilitate the use of the EMC Centera as the data repository. The removal of this code is outside of the scope of this LLD, and is not desirable as it may have an adverse effect on the Audit system as a whole. Where applicable the presence of such code has been documented.

6.2.1 ExtractorLink Executable

Instantiation of the ExtractorLink application is achieved via insert triggers on the FileFoundConsole and ClosedRFI tables on the RFISQL database.

Inserting a row into either of these tables results in the generation of an argument list which is passed to the ExtractorLink executable, and which dictates the execution route to be followed, and where relevant, specific ARQ details.

Arguments are in the form:

ARQReference_1,Instance_2,CmdUsedID_3,ErrCode_4,UserDrive_5,StateID_6

In the event of a failure occurring the ExtractorLink executable will instantiate a LogFailure executable passing:

ARQReference, Instance, CmdUsedID, ErrorNumber, & ServerName

FUJ00176545

as arguments.

The value of the '*ErrorNumber*' parameter, relates directly to the called executable that ExtractorLink.exe failed to start.

Failed .exe	ErrorNumber
Recover	199
Volume	399
FileSizes	499
TMSClear	599
TMSGenerate	699
ClusterID	799
CreateDir	899
UpdatAP	1099
CloseTidyLog	1299
CenteraDelete	1399
BuildOracle	1499

Table 4 – Extractor Link Error codes

Additionally an Application Event log entry is created identifying the failing ARQ reference and Instance. The log entry is recorded as an error with a reference made to the "Support Guide for Audit Extractor". The fields 'StateId' and 'ErrCode' are updated with an error code and narration in the '***FilesFoundConsole***' or '***ClosedRFI***' as appropriate.

6.2.2 AgentGetID.exe

Called by the NWB_Get_ClusterId.exe, this component provides cluster information for specified Branch codes under HNG-X. Data will be accessed through a Branch Database view.

TBS

6.2.3 Build_Oracle_Table.exe

This facility allows the user to rebuild Oracle formatted data from Audit tracks containing Oracle backup files, and is called by BuildOracle.exe (see BuildOracle.exe), receives as arguments:

ARQReference, Drive, FilePathAndTextfilename, ServerName and *PathAndSelectedFilename*

The argument '*FilePathAndTextfilename'* identifies the location and name of the text file generated and being monitored by BuildOracle.exe, whereas *PathAndSelectedFilename* provides the location and name of the file selected by the user via frmFilesFound.

Two new text files are generated, based on '*FilePathAndTextfilename'* (also held in the same directory) having "_out.txt" and "_err.txt" appended to the textfile name. The first textfile is used to write the output from a called executable whilst the second captures any error output.

An application named STDC001.exe, residing in the 'D:\ArchiveServer directory, is started passing:

username/password -f PathAndSelectedFilename

FUJ00176545

as arguments. The process identifier for this application is determined and used to check at one second intervals whether this application has terminated. This checking continues until completion or a timeout value being exceeded. the timeout value is set to five minutes.

Upon termination the size of the '_err.txt' textfile is evaluated . If it is other than zero the header of the textfile identified by '*FilePathAndTextfilename'* is amended to 'ORACLEBUILD_NOK'. The header is likewise changed if a timeout occurs or it has not been possible to start STDC001.exe.

Successful completion results in the header of the textfile identified by '*FilePathAndTextfilename'* being amended to 'ORACLEBUILD_OK'

6.2.4 BuildOracle.exe

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole' with the value entered into the field 'CmdUsedID' being 14.

The arguments received by the SQLServer trigger 'ExtractorLink' (see Trigger Mechanism.) are validated and the StateID field in the 'FileFoundConsole' updated to 3. This indicates the action is 'Processing'.

A text file named:

x:\USERAREA\y\y_ORACLEBUILD_<Instance>_YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'ORACLEBUILD', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time.

The second contains the user selected filename followed by the normal end of line characters without white spaces or white space lines.

Having created the text file, a lower level executable named 'Build_Oracle_Table' is called, (See Build_Oracle_Table.exe) passing parameters*:*

ARQReference, Drive, FilePath, ServerName and *PathAndSelectedFilename*

The running of the lower level executable is monitored at 30-second intervals until it completes or a timeout value occurs. The timeout value is derived from the value of TimeoutClusterID in the RFISQL table ***CommonControls*** subject to a minimum value of five minutes.

Upon the lower level executable 'Build_Oracle_Table' completing, the text file in the userarea containing the file list will have been updated. Examination of the 13th and 14th characters of the keyword indicates success or failure. Only if the 13th and 14th characters are 'OK' has the process been successful.

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is, 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

FUJ00176545

6.2.5 CenteraDelete.exe

This is a system facility instigated via a daily scheduled job named 'PopulateRFICenteraDelateTables' running on SQLServer which adds a record to the FileFoundConsole table writing a 'CmdUsedID' value of 13 and the 'ARQ Reference' value 'DailyCenteraDeleteFiles'.

The arguments received by the SQLServer trigger 'ExtractorLink' are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'.

The executable calculates from two RFISQL tables namely: ***DailyDeleteCenteraInitialTrack*** and ***DailyDeleteCenteraDupInitialTrack*** whether any records need deletion. If none exist the 'StateID' value in the *FileFoundConsole* table is update to '4', indicating successful completion , and an event log entry is generated to indicate there were no records due for deletion.

If records exist requiring deletion, a textfile is generated named:

x:\USERAREA\y\z_ YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'CENTERADELETE', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = CENTERADELETE

z = CPurge

YYYYMMDD_HHMMSS = the current date and time

Second and subsequent lines list filenames. Each entry is written on a separate line, using the normal end of line characters without white spaces or white space lines.

Having created the textfile, this facility next calls a lower level executable 'NWB_Centera_Delete', which deletes the files from Centera. The running of the lower level executable 'NWB_Centera_Delete' is monitored at 30 second intervals until it completes or passes a timeout value of 30 minutes.

In the event the lower level executable 'NWB_Centera_Delete' returns an error, indicated by the first line of the textfile in the user area being 'CENTERADELETE_FAIL, the second and subsequent lines will contain the files unable to be deleted from Centera. Filenames able to be deleted will have been removed from the textfile.

In the event the lower level executable 'NWB_Centera_Delete' runs successfully, the contents of the textfile in the user will contain only the text 'CENTERADELETE_OK.

Upon success or failure of 'NWB_Centera_Delete' this executable will remove all files from the ***DailyDeleteCenteraInitialTrack*** and ***DailyDeleteCenteraDupInitialTrack*** tables other than any files named in the textfile. For these files which, 'NWB_Centera_Delete' has been unable to delete from Centera, the field 'DeleteFail' is updated with the value 'Fail'.

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'.

6.2.6 CloseTidyLog.exe

This facility implements the 'Close' action and is triggered by the user closing an ARQ via the AuditExtractorClient. The ' CloseTidyLog.exe' action, without user involvement, moves files within the RFISQL database from 'active' tables to tables used solely for holding permanent data, or 'archiving tables'.

FUJ00176545

The purpose of the ' CloseTidyLog.exe ' action is to:

1. Provide within the RFISQL database a log of all user operations on an ARQ
2. Undertake housekeeping:

   within the RFISQL database.

   on the server 'D:\USERAREA' directory structure
3. To generate a user report of the 'log' providing an Audit Trail.

This facility is called, by ExtractorLink.exe upon a new record being created in the RFISQL database table ClosedRFI with the value entered into the field ‘CmdUsedID’ being 12, via the SQLServer trigger ‘CenteraPh2AddTrigger'.

Upon starting the 'StateID' is updated to 3 indicating the action is ‘Processing’. Records for the ARQ being closed are copied from the FileFoundConsole table to the ClosedRFIActions table. Additionally the ClosedRFIFiles table is populated with values from the fields:

'Status' and 'AW Operator ID' in the Requests table, and

'Filename' and 'Status' in the ActiveFiles table (where the 'Status' is not 'Displayed').

As each record is written, an entry is added to newly created log file, held as a text file named:

x:\ARCHIVESERVER\USERAREA\y\RFIy.txt

where

x = Drive letter taken from RFISQL table CommonControl field ‘Drive’

y = ARQ Reference

Failure to write the records results in a rollback and the value of the field 'StateID' in the ClosedRFIActions table is updated to '5' indicating 'ProcessNOK'. An 'Event Log' entry is generated advising of failure and the textfile Log is deleted.

In addition the files QueryHandler.log, xquillaErr*.txt, xquilaOut*,txt, PerlErr.txt and PerlOut.txt are appended to this close log.

Upon successful completion, the data is deleted from the non-permanent tables and the log file moved from the directory 'x:\ARCHIVESERVER\USERAREA\y\', as above, to x:\ARCHIVESERVER\USERAREA'. The 'StateID' is updated to '4' indicating 'ProcessOK', and the record that caused the new trigger to fire is deleted from the ClosedRFI table.

The record in the ClosedRFIActions table remains, evidencing the closure of the ARQ.

6.2.7 ClusterID.exe

This facility implements a request to identify one or more ‘ClusterID’s from a specific FAD code. The ClusterID’s that are obtained can then be used to identify particular AuditPoints that are used in ‘frmUpdateAuditPoints’. This facility is an aid to the ‘user’ when refining the search criteria to obtain an active file list.

This facility is required to provide ClusterIDs for both Horizon and HNG-X data in a common manner, utilising two different data sources. Under Horizon this data is provided by the Tivoli Web Service. Under HNG-X this information will be provided by a view on the Branch Database (**TBS**).

FUJ00176545

The amalgamation of these two sources is achieved by sending the request to both the XplorerGetId.exe and AgentGetId.exe components, and merging the results.

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole' with the value entered into the field 'CmdUsedID' being 7.

The arguments received by the SQLServer trigger 'ExtractorLink' are validated and the StateID field in the 'FileFoundConsole' updated to 3. This indicates the action is 'Processing'.

In trying to satisfy the customer's request for audit trail information, relating to transactions undertaken through the Horizon system over a specific period of time from nominated post offices, it is necessary to identify the associated 'cluster' information that will contain the specific post office messages.

A text file named:

x:\USERAREA\y\y_CLUSTERID_<Instance>_YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'CLUSTERID', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time.

The second contains the FAD code. Each line contains the normal end of line characters without white spaces or white space lines.

Having created the text file, a lower level executable named 'NWB_GetClusterID is called, passing parameters:

RFIReference, Drive, FilePath, ServerName and *Timeout value*

The running of the lower level executable is monitored at 30-second intervals until it completes or a timeout value (calculated by flexing the value held in field 'TimeoutClusterID' in the RFISQL table CommonControl by 10%) occurs.

Upon the lower level executable 'NWB_ GetClusterID.exe' completing, the text file in the userarea containing the file list will have been updated. Examination of the 10th and 11th characters of the keyword indicates success or failure. Only if the 10th and 11th characters are 'OK' has the process has been successful.

Success will indicate that an information word or comma separated list of ClusterID's will have been written by 'NWB_ GetClusterID.exe' to the second line of the text file. The second line contents are used to populate the field 'ReturnStr' in the RFISQL table 'fileFoundArgument'.

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is, 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

In the event the lower level executable 'NWB_ GetClusterID.exe' returns an error, indicated by the first line of the text file in the user area being 'CLUSTERID _FAIL_nnn' where nnn represents an error code, the second and subsequent lines will contain the error message. This error message and the error code from the first line are written to the fields 'ErrCode' and 'ErrMessage' of the RFISQL table 'FileFoundConsole' and will suppress any future error codes and error message this executable creates.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

FUJ00176545

6.2.8 CreateDir.exe

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole' with the value entered into the field 'CmdUsedID' being 8.

The creation of a new record in the RFISQL database table 'FileFoundConsole' is generated by the user clicking either the 'Save' or 'Specify Selection Criteria' command buttons on frmNew.

The arguments received by the SQLServer trigger 'ExtractorLink', (see Trigger Mechanism), are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'.

A test is performed to determine if the directory '\USERAREA' exists on the server, if it does not then it is created. The same process is repeated for:

'\USERAREA*ARQ*'

'\USERAREA*ARQ*\RESTORED_AT'

'\USERAREA*ARQ*\RETRIEVED_AT'

'\USERAREA*ARQ*\EXTRACTED_AT'

'\USERAREA*ARQ*\EXTRACTED_AT'

'\USERAREA*ARQ*\QUERY_AT'

(where *ARQ* is the ARQ Reference number)

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause.

6.2.9 FileSizes.exe

This execution path has been disabled at the client application but still exists at the server.

It is retained and documented in this release, only to remove the necessity to amend the Generate Message Store functionality, which will ultimately be deprecated.

This facility implements a request to obtain details of the size of one or more files that are stored on tape.

This information is needed by the user to:

a) calculate an approximate time for transfer

b) for use with GenerateMessageStore

c) for use with the CD writer

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole' with the value entered into the field 'CmdUsedID' being 4.

The arguments received by the SQLServer trigger 'ExtractorLink', are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'.

At the Extractor Client (workstation) end the user will have selected one or more files from the RFISQL table 'ActiveFiles', applicable to a specific RFI, and clicked on the 'File SIZES' button. This list of 'selected' files are then added to the 'FileFoundArgument' table.

FUJ00176545

This executable then presents this list of files to the Legato Networker facility to obtain the size of the file (as reported to Legato Networker when written to tape) and return to the client. This is undertaken by writing a text file to the user area directory at x:\Userarea\y\ where:

A text file named:

x:\USERAREA\y\y_ FILESIZE_<Instance>_YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'FILESIZE', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time.

Second and subsequent lines take the format:

<PathOnly>\<TapePool>\<AuditPoint>\<FileName>

where <PathOnly> is extracted from the ***CommonControls*** table, <TapePool> and <AuditPoint> from the ***ActiveFiles*** table and <FileName> from the Object field of the ***FileFoundArgument*** table. All tables being in the RFISQL database. Each entry is written on a separate line, using the normal end of line characters without white spaces or white space lines.

This facility calls a lower level executable 'NWB_File_Size', which does the bulk of the work to instigate the file size extraction (for the nominated file, or files) from the Legato Networker system. The returned details are put into the field 'FileSize' of the RFISQL table ***ActiveFiles***.

The running of the lower level executable 'NWB_File_Size' is monitored at 30 second intervals until it completes or a timeout value (calculated by flexing the value held in field 'TimeoutFileSizes' in the RFISQL table CommonControl by 10%) occurs.

Upon the lower level executable 'NWB_File_Size' completing, the text file in the userarea containing the file list will have been updated. Examination of the 10th and 11th characters of the keyword indicates success or failure. Only if the 10th and 11th characters are 'OK' has the process has been successful.

Success will indicate that a comma and a number indicating (in bytes) the file size has been added to the second and subsequent lines of the text file. This executable then writes this file size to the field 'Size' of the ***ActiveFiles*** table for each corresponding record in the table If Legato has been unable to find the nominated file in it's index then it will have return the word 'Unknown' instead of a size. This executable interprets 'Unknown' as the value '0' (ZERO) and writes this value to the field 'Size' of the ***ActiveFiles*** table.

Upon successful completion the 'StateID' field in the ***FileFoundConsole*** table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

In the event the lower level executable 'NWB_File_Size' returns an error, indicated by the first line of the text file in the user area being 'FILESIZE _FAIL_nnn' where nnn represents an error code, the second and subsequent lines will contain the error message. This error message and the error code from the first line are written to the fields 'ErrCode' and 'ErrMessage' of the RFISQL table 'FileFoundConsole' and will suppress any future error codes and error message this executable creates.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

FUJ00176545

6.2.10 LogFailure.exe

As detailed in ExtractorLink.exe the linking mechanism 'ExtractorLink.exe' receives a parameter 'CmdUsedId', which is used to determine which executable it in turn calls.

If ExtracorLink.exe is unable to confirm that the required executable exists, for example it may not exist in D:\ARCHIVESERVER or may have been renamed, ExtractorLink.exe defaults to calling LogFailure.exe passing *RFIReference, Instance, CmdUsedID, ErrorNumber, & ServerName* as arguments.

This executable updates both the StateID field in the 'FileFoundConsole' table to 5 indicating a failed process ('ProcessNOK') and additionally populates the field 'ErrMessage' with an error message tailored with the name of the executable ExtractorLink.exe failed to call.

In the event the executable is unable to update the StateID field in the 'FileFoundConsole' table it generates an Application Event log entry showing an error message indicating failure to run the named executable.

6.2.11 Recover.exe

This facility implements a user's request to obtain a file or a number of files from tape via the Legato Networker system, or Centera disk array, to the user's file store area.

While the majority of code still relates to the retrieval of Audit Tracks from tape, the NWB_Legato_Recover.exe component can be configured, via the Audit server configuration file, to retrieve data from tape, disk or Centera. The process for signalling retrieval of Audit Tracks and reporting results is identical for all storage mediums, with the actual retrieval being the only difference.

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole' with the value entered into the field 'CmdUsedID' being 1.

The arguments received by the SQLServer trigger 'ExtractorLink' (see Trigger Mechanism) are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'.

At the Extractor Client (workstation) end the user will have selected one or more files from the 'ActiveFiles table, applicable to a specific ARQ, and clicked on the 'RESTORE files' button. This list of 'selected' files is added to the 'FileFoundArgument' table. Additionally the selected files in the 'ActiveFiles' table have their Status field set to: 'Requested'. The 'Recover.exe' action is invoked by creating a new record in the 'FileFoundConsole' table.

The extractor system (server end) then presents this list of files to the Legato Networker facility and requests the recovery of the nominated files. A text file named:

x:\USERAREA\y\y_ RECOVER _<Instance>_YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'RECOVER', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time

Second and subsequent lines take the format:

<PathOnly>\<TapePool>\<AuditPoint>\<FileName>

where <PathOnly> is extracted from the 'CommonControls' table, <TapePool> and <AuditPoint> from the 'ActiveFiles' table and <FileName> from the Object field of the FileFoundArgument table. All tables

FUJ00176545

being in the RFISQL database. Each entry is written on a separate line, using the normal end of line characters without white spaces or white space lines.

This facility calls a lower level executable 'NWB_Legato_Recover, which does the bulk of the work to instigate, file recovery and return the status of the individual file recovery requests. Execution branching, to either Legato, disk or Centera retrieval code takes place here. The status returned, is the status of the file request as acknowledged by the Audit Centera or Legato not the status of the action of recovering the file.

The running of the lower level executable 'NWB_Legato_Recover' is monitored at 30 second intervals until it completes or a timeout value (calculated by flexing the value held in field 'TimeoutRecover' in the RFISQL table CommonControl by 10%) occurs.

Upon the lower level executable 'NWB_Legato_Recover' completing, the text file in the userarea containing the file list will have been updated. Examination of the 10th and 11th characters of the keyword indicates success or failure. Only if the 10th and 11th characters are 'OK' has the process has been successful, meaning that the second and subsequent lines have had a comma followed by a descriptive status added. Permissible status values are detailed in the table 'ActiveFileStatusValues'.

This executable then reads the text file, amended by 'NWB_Legato_Recover', writing the status value to the field Status of the 'ActiveFiles' table for each corresponding record in the table. However prior to updating the status value the current value is read and only if the value is 'Requested' is it overwritten.

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

In the event the lower level executable 'NWB_Legato_Recover' returns an error, indicated by the first line of the text file in the user area being 'RECOVER_FAIL_nnn' where nnn represents an error code, the second and subsequent lines will contain the error message. This error message and the error code from the first line are written to the fields 'ErrCode' and 'ErrMessage' of the RFISQL table 'FileFoundConsole' and will suppress any future error codes and error message this executable creates.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

6.2.12 TMSClear.exe

This facility implements a user's request to reset or clear a MessageStore ready for importing or adding new TMS audit track data.

It will not be utilised by the Extractor client application, but should be retained to allow the execution of the RQueryUK application until Riposte is decommissioned.

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole', with the value entered into the field 'CmdUsedID' being 5.

In trying to satisfy the customer's request for audit trail information, relating to transactions undertaken through the Horizon system over a specific period of time from nominated post offices, it is necessary to reconstitute the MessageStore with the messages that were generated over a set period of time. However, at times, typically when a 'New' RFI is started it is necessary to clear out the MessageStore area. Thus there is a need for this facility.

The arguments received by the SQLServer trigger 'ExtractorLink' (see Trigger Mechanism) are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'

FUJ00176545

The executable next examines the value of the field MS_BuildOnServer in the 'CommonControl table of the RFISQL database. If the value of the field is 'FREE' or any value other than the current RFI Reference then the Message Store on the users workstation is to be cleared. A field value matching the current RFI Reference indicates the Message Store on the server is to be cleared.

Two timeout values are calculated, the first a local timeout value is calculated by flexing the value held in field 'TimeoutTMSClear' in the RFISQL table CommonControl by 10%. The second, a 'pass on' timeout value, depends on the location of the Message Store to be cleared. For a Message Store based on the server the value of 'TimeoutTMSClear' can be used whereas for a workstation based Message Store the value of 'TimeoutTMSClear' is flexed by a factor of two.

A text file named:

x:\USERAREA\y\y_TMSCLEAR _<Instance>_YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'TMSCLEAR', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time.

Having created the text file a lower level executable named 'NWB_TMS_Clear.exe' is called passing parameters*: RFIReference, Drive, FilePath, ServerName* and *Timeout value.* The running of the lower level executable 'NWB_TMS_Clear' is monitored at 30-second intervals until it completes or the local timeout value occurs.

Upon the lower level executable 'NWB_TMS_Clear' completing, the text file in the userarea containing the file list will have been updated. Examination of the 10th and 11th characters of the keyword indicates success or failure. Only if the 10th and 11th characters are 'OK' has the process has been successful.

This executable then reads the text file, amended by 'NWB_TMS_Clear'. If the header indicates failure, indicated by the text file header resembling:

TMSCLEAR_FAIL_nnn

where 'nnn' relate to a numeric error code, the second and subsequent lines of the text file are read and form an error message. Successful completion does not result in a second line of text, instead 'nnn' will hold the value'0' (Zero).

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

In the event the lower level executable 'NWB_TMS_Clear' returns an error, the returned error message and the error code are written to the fields 'ErrCode' and 'ErrMessage' of the RFISQL table 'FileFoundConsole' and will suppress any future error codes and error message this executable creates.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

6.2.13 TMSGenerate.exe

This facility implements a user's request to populate a MessageStore with data from files that have been restored and exist in this ARQ's user area and have been seal checked.

FUJ00176545

It will not be utilised by the Extractor client application, but should be retained to allow the execution of the RQueryUK application until Riposte is decommissioned.

In trying to satisfy the customer's request for audit trail information, relating to transactions undertaken through the Horizon system over a specific period of time from nominated post offices, it is necessary to reconstitute the MessageStore with the messages that were generated over a set period of time. This facility implements the 'User Agent' utility that will scan a user area for a specified format filename, open that file and extract data that conforms to a number of criteria. These being a date range and a FAD code.

This facility is called by 'ExtractorLink.exe' upon a new record being created in the RFISQL database table 'FileFoundConsole', with the value entered into the field 'CmdUsedID' being 6.

The arguments received by the SQLServer trigger 'ExtractorLink' (see Trigger Mechanism) are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'

The executable next examines the value of the field MS_BuildOnServer in the 'CommonControl' table of the RFISQL database. If the value of the field is 'FREE' or any value other than the current RFI Reference, then the Message Store is to be created on the users workstation. A field value matching the current RFI Reference indicates a Message Store on the server is to be created.

Two timeout values are calculated, the first a local timeout value is calculated by flexing the value held in field 'TimeoutTMSGenerate' in the RFISQL table CommonControl by 10%. The second, a 'pass on' timeout value, depends on the location of the Message Store to be cleared. For a Message Store based on the server the value of 'TimeoutTMSGenerate' can be used whereas for a workstation based Message Store the value of 'TimeoutTMSGenerate' is flexed by a factor of two.

A text file named:

x:\USERAREA\y\y_TMSGENERATE _<Instance>_YYYYMMDD_HHMMSS.txt

is created with the first line keyword as 'TMSGENERATE', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time.

Having created the text file a lower level executable named 'NWB_TMS_Generate.exe' is called passing parameters*: RFIReference, Drive, FilePath, ServerName* and *Timeout value.* The running of the lower level executable 'NWB_TMS_Generate' is monitored at 30 second intervals until it completes or the local timeout value occurs.

Upon the lower level executable 'NWB_TMS_ Generate' completing, the text file in the userarea, containing the file list, will have been updated. Examination of the 10th and 11th characters of the keyword indicates success or failure. Only if the 10th and 11th characters are 'OK' has the process has been successful.

This executable then reads the text file, amended by 'NWB_TMS_ Generate'. If the header indicates failure, indicated by the text file header resembling:

TMSGENERATE_FAIL_nnn

where 'nnn' relate to a numeric error code, the second and subsequent lines of the text file are read and form an error message. Successful completion does not result in a second line of text, instead 'nnn' will hold the value'0' (Zero).

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error

FUJ00176545

code ‘0’ indicates no error. Other error codes returned relate to the errors cause and in addition the field ‘ErrMessage’ is populated with a descriptive text of the error.

In the event the lower level executable ‘NWB_TMS_ Generate’ returns an error, the returned error message and the error code are written to the fields ‘ErrCode’ and ‘ErrMessage’ of the RFISQL table ‘FileFoundConsole’ and will suppress any future error codes and error message this executable creates.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

6.2.14 UpdatAP.exe

This facility implements a system command to re-populate the tables ‘AuditPoints’, ‘AuditSubPoints’ and ‘TapePools’ within the RFISQL database, taking data from the file ‘ConfigurationFile.txt’ located at ‘D:\ARCHIVESERVER’.

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table ‘FileFoundConsole’, with the value entered into the field ‘CmdUsedID’ being 10.

The arguments received by the SQLServer trigger ‘ExtractorLink’ (see Trigger Mechanism) are validated and the StateID field in the ‘FileFoundConsole’ updated to 3 indicating the action is ‘Processing’

The configuration file is first read and the number of TapePools determined from the ‘NoOfTapePools ‘ value in the ‘[General]’ section of the configuration file. A set of all ‘TapePool’ values held in the ‘TapePool’ table in the RFISQL database are extracted and each TapePool value from the configuration file is evaluated against every value in the set extracted from the database.

If the values match, no action is taken other than to proceed to the next item from the configuration file. If there is no matching record in the set extracted from the database then a new record is created in the table ‘TapePool’.

A similar process is undertaken to update the ‘AuditPoints’ table, which contains the fields ‘AuditPoints’ and ‘TapePool’. The configuration file is read and the number of AuditPoints determined from the ‘NoOfAuditPoints’ value in the ‘[General]’ section of the configuration file. A set of all 'AuditPoints' and ‘TapePool’ values, that is a set of pairs of values, held in the ‘AuditPoints’ table in the RFISQL database is likewise extracted.

One by one the AuditPoint value and TapePool values, as a pair, are taken from the configuration file until all have been evaluated. The AuditPoint value of the pairing is compared with every AuditPoint value within the set taken from the ‘AuditPoints’ table in the RFISQL database.

If no match is found a new record consisting of both AuditPoint and TapePool values are added to the ‘AuditPoints’ table in the database. If the AuditPoint is found then a further comparison is undertaken for the pairing, that is, the TapePool value is compared with the corresponding TapePool value from the database set.

If no record in the database is found with both the AuditPoint value and TapePool values matching those of the pairing from the configuation file a new record is added. If both values are already held in the database the process moves on to the next pairing.

Having updated both the ‘TapePool’ and ‘AuditPoints’ tables the executable next updates the ‘AuditSubPoints’ table. A set of all 'AuditPoints' and ‘AuditSubPoints’ values, that is a set of pairs of values, held in the ‘AuditSubPoints’ table in the RFISQL database is likewise extracted.

One by one the AuditPoint value and AuditSubPoint values, as a pair, are taken from the configuration file until all have been evaluated. Each pairing is compared with every pairing from the set extracted from the database. If a match is not found a new record is created.

FUJ00176545

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

6.2.15 Volume.exe

This execution path has been disabled at the client application but still exists at the server.

It is retained and documented in this release as the existence of dependencies could not be determined within the scope of this document.

This facility implements a user's request to obtain details of the 'tape' media that contains the files that the user wishes to restore. The user has to supply this information to operators at the data centres as a request (by telephone) for specific tapes to be retrieved from the secure storage and put 'on-line'.

This facility is called by ExtractorLink.exe upon a new record being created in the RFISQL database table 'FileFoundConsole' with the value entered into the field 'CmdUsedID' being 3.

The arguments received by the SQLServer trigger 'ExtractorLink' (see Trigger Mechanism) are validated and the StateID field in the 'FileFoundConsole' updated to 3 indicating the action is 'Processing'

At the Extractor Client (workstation) end the user will have selected one or more files from the ActiveFiles table, applicable to a specific ARQ, and clicked on the 'VOLUME Status' button. This 'selected' files list will have been added to the FileFoundArgument table and the 'Volume Status' action invoked

The extractor system (server end) presents this list of files to the Legato Networker facility (and other server systems) to obtain the following media information for the selected file:

Volume Name, (Name identifier, as written to tape (tape label) of the tape volume that holds this file)

Volume Status, ('on line', 'off line' or expired, or ….)

Barcode, (The barcode value that is on the tape case. This is read by Legato when the tape is put 'on-line'. Note it may or may not exist)

This is achieved by the executable creating a text file named:

x:\USERAREA\y\y_VOLUME _<Instance>_YYYYMMDD_HHMMSS.txt

with the first line keyword as 'VOLUME', where

x = Drive letter taken from RFISQL table CommonControl field 'Drive'

y = ARQ reference

<Instance> = the instance value of the current ARQ

YYYYMMDD_HHMMSS = the current date and time.

Second and subsequent lines take the format:

<PathOnly>\<TapePool>\<AuditPoint>\<FileName>

where <PathOnly> is extracted from the 'CommonControls' table, <TapePool> and <AuditPoint> from the 'ActiveFiles' table and <FileName> from the Object field of the FileFoundArgument table. All tables being in the RFISQL database. Each entry is written on a separate line, using the normal end of line characters without white spaces or white space lines.

FUJ00176545

This facility calls a lower level executable 'NWB_Volume_Status', which does the bulk of the work to instigate the volume status extraction (for the nominated file(s)) from the Legato Networker system.

The running of the lower level executable 'NWB_Volume_Status' is monitored at 30 second intervals until it completes or a timeout value (calculated by flexing the value held in field 'TimeoutRecover' in the RFISQL table CommonControl by 10%) occurs.

Upon the lower level executable 'NWB_Volume_Status' completing, the text file in the userarea containing the file list will have been updated. Examination of the 10th and 11th characters of the keyword indicates success or failure.

Only if the 10th and 11th characters are 'OK' has the process has been successful. If the process has been successful, the first line of the text file will resemble 'VOLUME_OK_nnn' where 'nnn' indicates a count of the number of files for which data is required.

Upon success, the second and subsequent lines will have added a comma and will take the form of either:

A) If Legato is unable to identify the tape holding this file, for whatever reason, then a keyword (NODETAILS) will be returned. In this instance the executable populates the field 'VolumeName' in the 'ActiveFiles' table with the narrative:

'***No Legato volume(s) identified'*** .

B) If Legato obtains the details then the files will be returned with appropriate data, namely three comma separated entries:

Volume,OnLine,Barcode

However note that whilst the tape volume name and its status will always be present, the barcode value may not be present. In which case the keyword (NULL) will be used to indicate no bar code. In this instance the 'BarCode' field in the 'Volumes' table in the RFISQL database will be left blank.

C) To cater for the situation where a large file spans two (or more) tape volumes, the format of the data in the file in this instance will form a seven comma separated entry, e.g.

*,<VolumeName>,< VolumeStatus>,<BarCode>,

< VolumeName>,< VolumeStatus>,,<BarCode>

Where the '*' acts as a flag to indicate the file spans two (or more) tape volumes. However note that whilst the tape volume name and its status will always be present, the barcode value may not be present, in any combination. The database is updated as per B) above.

Upon successful completion the StateID field in the 'FileFoundConsole' table is updated to 4 indicating 'ProcessOK'. For failure the StateID field is updated to 5, that is 'ProcessNOK'. Irrespective of success or failure an appropriate error code is written to the ErrCode field in the 'FileFoundConsole'. An error code '0' indicates no error. Other error codes returned relate to the errors cause and in addition the field 'ErrMessage' is populated with a descriptive text of the error.

In the event the lower level executable 'NWB_Volume_Status' returns an error, indicated by the first line of the text file in the user area being 'VOLUME_FAIL_nnn' where nnn represents an error code, the second and subsequent lines will contain the error message. This error message and the error code from the first line are written to the fields 'ErrCode' and 'ErrMessage' of the RFISQL table 'FileFoundConsole' and will suppress any future error codes and error message this executable creates.

Upon success or failure of the lower level executable all records written to the FileFoundArgument table of the RFISQL database are deleted.

6.3 Non-core Components – Filtering and Querying

FUJ00176545

Data filtration and abstraction is performed within the ARQ framework, and may only commence once all files for the ARQ have been extracted and seal checked.

Where processing of Audit Track data is dependant upon the source platform, the source of the data may be determined from the RFIDataSource and RFIAuditSubPointExtended tables on the RFISQL database. Permitted processing may also be determined from these tables.

The filtering and querying process involves two distinct stages, filtering and querying. The extent to which the process may be applied is dependant upon the format of the Audit Track data. Counter Transaction records, originating under either Horizon or HNG-X, may be filtered and queried, while other data formats may only be filtered.

During the filtering stage files are read one message, or line, at a time and the filtering criteria applied. If the data retrieval contains both Horizon and HNG-X data, the files will be processed by platform. This may involve the application of two sets of criteria.

If a match is found the file may be marked as matching, or the line/message written to an intermediate file for concatenation at the end of the filtration. Data written to intermediate files must only be concatenated with data from the same platform.

If the Audit Track data is suitable, the concatenated intermediate files may be queried. Queries will be platform specific due to differences in attribute naming and data structure.

Querying the concatenated files will allow selected attributes to be included in the final output, as well as allowing for finer a granularity of message selection to be applied. The result of the query will be held in an intermediate file relating to the data source from which the data originated.

Throughout this process files manipulation will be contained beneath the QUERY_AT directory structure of the relevant ARQ directory in the UserShare area of the appropriate Audit Server.

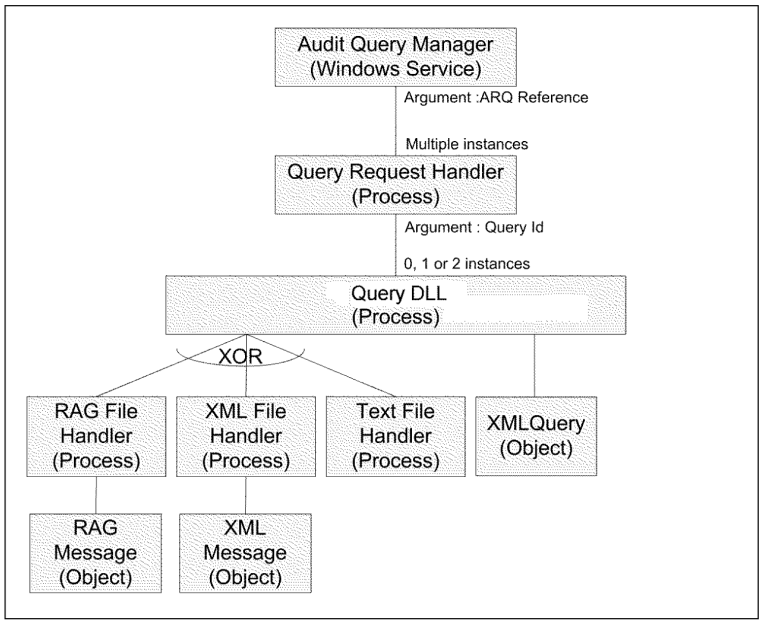

Figure 3 – Filtering and Query Component Structure

FUJ00176545

6.3.1 Audit Query Manager (AQM)

6.3.1.1 Interfaces

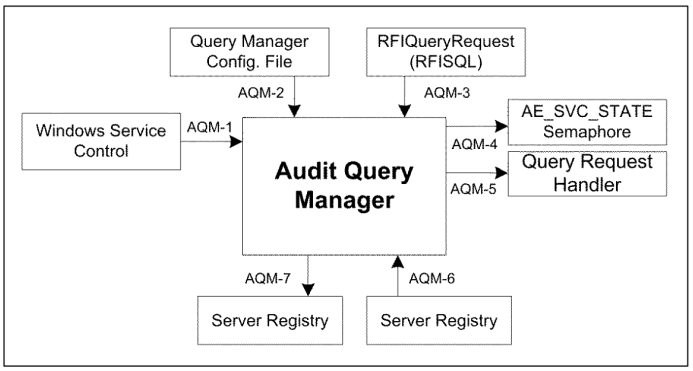

Figure 4 - Audit Query Manager Interfaces

AQM-1 Windows Service Control

Accepts Start, Stop Pause and Restart signals from the operating systems Service Control interface.

AQM-2 Query Manager Configuration File

Reads the D:\ArchiveServer\QMComponents\QueryManager.ini configuration file.

Expects the values specified in Section 6.3.1.2.8 Query Manager Initialisation File:

AQM-3 RFISQL Database – RFIQueryRequest Table (Read)

Read pending query requests from the RFIQueryRequest table of the RFISQL database.

AQM-4 AE_SVC_STATE Semaphore

Instantiates the AE_SVC_STATE Semaphore which is used to co-ordinate termination across all sub-components. The semaphore is set to signal termination to sub-components.

AQM-5 Query Request Handler (Calls to)

Constructor

Creates an instance of the Query Request Handler for each query request to be processed, up to the maximum number of concurrent query requests specified in the configuration file.

Arguments : QueryRequestId (Integer) from RFIQueryRequest row

Process

Instigates query request processing. No arguments passed.

FUJ00176545

AQM-6 Server Registry (Read)

Reads the following keys from the server registry;

- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentFiles
- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentQueries
- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\DBConnectString

AQM-7 Server Registry (Write)

Creates the following keys in the server registry is not present:

- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentFiles
- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentQueries
- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\DBConnectString

Updates the values of the following keys if discrepancies are found with the configuration files values:

- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentFiles
- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentQueries
- HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\DBConnectString

6.3.1.2 Function

Query management at the server side is implemented as a Windows Service to ensure that any interruption, either planned or otherwise, to the normal processing of queries results in a recoverable state being maintained. The service runs under the local system account with automatic start up.

The Audit Query Manager is responsible for initiating, on a one per ARQ basis, Query Request Handler objects, and signalling commanded termination to all sub-components

Errors encountered during the execution of the Audit Query Manager are recorded at the Audit server event logs.

Synchronisation of the services functions is via the AE_SVC_MODE variable which indicates one of four possible states:

- Initialising (AE_SVC_MODE = 0)
- Active (AE_SVC_MODE = 1)
- Paused (AE_SVC_MODE = 2)
- Terminating (AE_SVC_MODE = 99)

Control of sub-components instantiated by the service is via the AE_SVC_STATE semaphore. This may only be used to signal immediate termination.

The service instantiates two threads, and maintains synchronisation across the threads by means of the AE_ INTERNAL_CTRL semaphore which is maintained by the Service Control Handler.

The possible combinations of the three control mechanisms and service states are shown in Table 3.

Service State	AE_SVC_MODE	AE_ INTERNAL_CTRL	AE_SVC_STATE	Description
Start	Initialising	NOT SET	NOT SET	The service is starting.
Running	Active	NOT SET	NOT SET	The service is running. The system time is

FUJ00176545

				not within a configured period of inactivity.
Running	Paused	NOT SET	SET	The service is running. The system time is within a configured period of inactivity.
Pause	Paused	SET	SET	The service is paused. Configured periods of activity/inactivity are ignored.
Resume	Active	NOT SET	NOT SET	The service is resuming within a configured period of activity.
Resume	Paused	NOT SET	SET	The service is resuming within a configured period of inactivity.
Stop	Terminating	SET	SET	The service is terminating. Configured periods of activity/inactivity are ignored.

Table 5 – Query Manager State Control Combinations

6.3.1.2.1 Service Control Handler

Registered by the service at loading, the Service Control Handler manages service start, pause, resume and stop commands received from the operating system service management console, setting the AE_ INTERNAL_CTRL and AE_SVC_STATE semaphores, and the AE_SVC_MODE variable accordingly (*See Table 3*).

6.3.1.2.2 Run Time Availability Thread

While the AE_ INTERNAL_CTRL semaphore is not set the Run Time Availability Thread performs periodic checks of the system time against the time bands extracted from the configuration file and sets the AE_SVC_STATE semaphore and AE_SVC_MODE variable accordingly.

If the AE_ INTERNAL_CTRL semaphore is set the thread will set the AE_SVC_MODE to Terminating, and set the AE_SVC_STATE semaphore before exiting and terminating.

This thread must complete it's first pass during the initialisation phase, prior to the primary worker thread instantiation to ensure a valid state is recorded in the AE_SVC_MODE variable.

6.3.1.2.3 Primary Worker Thread

The primary worker thread's function is dependent upon the AE_SVC_MODE value and is covered in detail in sections 6.3.1.5 Active State, 6.3.1.6 Pause State and 6.3.1.7 Terminating State

6.3.1.2.4 Initialisation State

This state is implemented by the Service Control Handler when the service receives a start command, and indicates that the service is performing pre-operational checks. Upon initialisation the following control states will be set:

AE_SVC_MODE	Initialising
AE_ INTERNAL_CTRL	NOT SET
AE_SVC_STATE	NO SET

This enables the starting conditions for the service to be set by the appropriate threads as they are initialised.

During the initialisation phase the Audit Query Manager Service reads and validates the QueryManager.ini configuration file (*Section 6.3.1.7*) held in the *D:\ArchiveServer\QMComponents* directory. If the file is missing, or contains errors the service records an appropriate error in the Audit server event log before terminating.

FUJ00176545

Concurrency and database settings extracted from the configuration file are compared to those in the server registry and the registry values updated if discrepancies are found.

Local concurrency and run time availability data is retained in local volatile memory for use by the service.

Upon successful completion of the configuration file processing, the Run Time Availability thread is instantiated to handle configured periods of inactivity, and upon completion of it's first pass the primary worker thread will be instantiated.

6.3.1.2.5 Active State

During active running the primary worker thread will, at intervals specified in the configuration file, periodically poll the RFIQueryRequest table on the RFISQL database.

Rows with a QueryRequestStatusId value equal to or greater than that of the Pending status (*Section 7.3.32*) and a ProcessHalt value of zero, will be returned in descending QueryRequestStatusId, and ascending submission date/time order.

This enforces the following ARQ prioritisation:

Nearest to completion (irrespective of submission date/time)

Earliest to last un-actioned submissions.

In the absence of any returned rows, the primary worker thread will enter a sleep state for the period specified in the configuration file, before polling the RFIQueryRequest table for un-actioned queries.

For each row returned (up to the maximum concurrent Query Requests value set in the configuration file) an out of process Query Request Handler component is instantiated with the QueryRequestId as an argument. Once instantiated a call is made to The Query Request Handler Process method.

The primary worker thread monitors each of the Query Request Handler processes. As processes terminate the RFIQueryRequest table is re-queried, and Query Request Handler processes instantiated to ensure the configured maximum number or query requests are processes concurrently.

6.3.1.2.6 Paused State

Upon entering a paused state as signalled by the Run Time Availability Monitor thread, the primary worker thread:

- Signal sub-components via the AE_SVC_STATE semaphore to terminate
- Waits for the termination of all active sub-components
- Enters a low resource wait state.
- Periodically checks the service state and respond accordingly

No other activities may be performed while the service is in a paused state.

If a commanded termination is received while in the paused state, the primary worker thread will exit and terminate.

6.3.1.2.7 Terminating State

Upon receiving a termination event all active sub-components will be signalled via the system semaphore that they should terminate.

The Query manager service will then terminate the Run Time Availability thread.

FUJ00176545

The primary worker thread will wait for a predefined time for the sub-components to indicate that termination has been performed, before forcing termination and exiting.

The service will then shut down.

6.3.1.2.8 Query Manager Initialisation File

The Query Manager Initialisation file is implemented as a Windows .ini file and holds all of the settings that are required to control the execution of the query manager and sub-components. The file will be read and validated when the service is started, requiring a re-start to implement any configuration changes.

Concurrency and database connection details will be compared to those held in the server registry for use by sub-components, and the registry updated where discrepancies exist.

The registry keys used are:

- **HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentFiles**
  A DWORD value containing the number of concurrent file handling threads per query that may be instantiated. Taken from the Concurrency – MaxFilesPerQuery ini file setting.

- **HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentQueries**
  A DWORD value containing the number of concurrent query threads per request that may be instantiated. Taken from the Concurrency – MaxQueries ini file setting.

- **HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\DBConnectString**
  A REG_SZ value containing a valid connection string for use by all query sub-components. This value will be derived through the combination of all ini file settings in the DBConnect category.

The configuration file, detailed below, can only contain single instances of concurrency, DBConnect and Execution Control and Pause settings, but may contain multiple Pause*X* entries (where X is an integer value).

Validation of PauseX settings is based upon the following rules;

- For each value of X, a StartTime and End Time value must be present.
- The EndTime must be greater than the StartTime, a Pause may not span a date change.
- Pause Periods must not overlap.
- X values must start at 1 and rise sequentially.
- No chronological ordering is implied by the X value. Eg. Pause1 may be later than Pause2

Section	Setting	DataType	Values/Format	Description
Concurrency	MaxARQs	Integer	1-100	Maximum number of concurrent query requests.
Concurrency	MaxQueries	Initeger	1-100	Maximum number of concurrent queries that can be processed.
Concurrency	MaxFilesPerQuery	Integer	1-100	Maximum number of files that can be concurrently processed per query.
DBConnect	Provider	String		Values required to build a valid connection string.
DBConnect	Integrated Security	String		
DBConnect	Persist Security Info	String		
DBConnect	Data Source	String		
DBConnect	Initial Catalog	String		

FUJ00176545

DBConnect	Procedure Prepare	String		
DBConnect	Auto Translate	String		
DBConnect	Packet Size	String		
DBConnect	Use Encryption	String		
DBConnect	Column Collation	String		
ExecutionControl	PollDBSeconds	Integer	30-600	Seconds between database polls.
ExecutionControl	QueryHandler	String		Valid path to executable.
Pause	PauseWaitTime	Integer	60-600	Seconds between checks to determine the Paused status
Pause*X*	StartTime	String	HH:MM (24 hour)	Start and end time during which all processing should be halted. Multiple pauses may be implemented under multiple Pause sections. Eg. Pause1, Pause2, etc.
Pause*X*	EndTime	String	HH:MM (24 hour)	

Table 6 – Query Manager Service Configuration File

6.3.2 Query Request Handler (AQR)

6.3.2.1 Interface

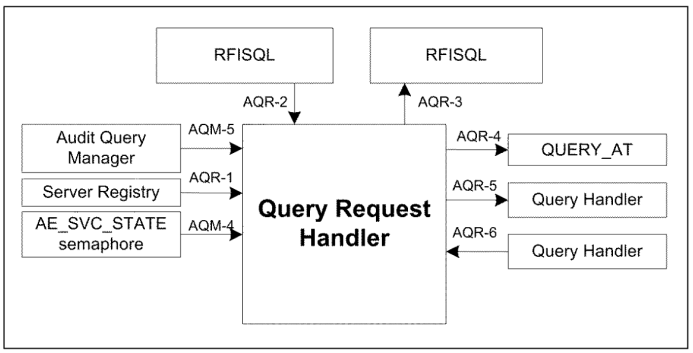

Figure 5 – Query Request Handler Interfaces

AQM-4 AE_SVC_STATE Semaphore

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQR-1 Server Registry

Reads the following:

HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentFiles
Type Integer
Value >= 1
Desc. Maximum number of concurrent file handling threads per query that may be instantiated.

HKEY_LOCAL_MACHINE\SOFTWARE\Fujitsu\POA\AuditServer\ConcurrentQueries
Type Integer

FUJ00176545

Value 0, 1 or 2
Desc. Determines the number of concurrent queries that may be processed

AQM-5 Audit Query Manager (Calls from)

Constructor

Receives an integer containing the QueryRequestId of the request to be processed.

Process

No arguments passed.

AQR-2 RFISQL Database (Reads)

Reads the RFIQueryRequest row for the ID supplied at construction.

Reads the RFIQuery rows for the current RFIQueryRequest

AQR-3 RFISQL Database (Writes)

Updates the Status of the RFIQueryRequest row for the current query request.

AQR-4 UserArea QUERY_AT (Create/Delete)

Creates and Deletes the ARQ Query Directory Structure:

D:\UserArea\ARQReference\QUERY_AT

D:\UserArea\ARQReference\QUERY_AT\ABSTRACTED

D:\UserArea\ARQReference\QUERY_AT\MERGED

D:\UserArea\ARQReference\QUERY_AT\XQUERY

D:\UserArea\ARQReference\QUERY_AT\FINAL

AQR-5 Query Handler (Calls to)

Constructor

Instantiates the Query Handler component passing the following arguments:

pRequest	Pointer to the Query Request Handler object.
pQueryId	Integer containing the QueryId
pMaxThreads	Integer containing the maximum number of concurrent file handler threads

Execute method

No arguments required.

AQR-6 Query Handler (Returns from)

Returns a Boolean value from the Query Handler object's Execute method.

True indicates successful completion.

FUJ00176545

False indicates an error was encountered.

6.3.2.2 Function

Instantiated by the Query Manager on a one per ARQ basis, the Query Request Handler controls all actions to be performed on an ARQ.

Upon instantiation an initial check is performed to determine if a query exists for the ARQ. In the absence of a valid query the Query Request Handler will update the RFIQueryRequest row Status to indicate failure before terminating.

The QueryRequestStatusId of the RFIQueryRequest table on the RFISQL database is read to determine if the query has been run to completion, or partially executed on a previous occasion. This is ascertained from the QueryRequestStatusId of the RFIQueryRequest table on the RFISQL database.

A status value of “Pending” indicated the first attempt to process the query. While a status value of “Complete” would indicate the requirement for a complete rerun. Any status value in between these two values indicate that a partially completed run has been performed.

If the QueryRequestStatusID indicates that this is the first run of the filtering and querying process, the RFIQueryFile table will be populated with a row for each entry in the [Active Files] table for the current ARQ.

Each sub-component called by the Query Request handler is responsible for determining the last successful action, and for restarting from the appropriate processing point. Where intermediate files are involved incompletely processed intermediate files will be deleted, and the process restarted from that point.

Control of the Query Request is via the AE_SVC_STATE semaphore generated by the QueryManager.

During the processing of a query request, the status of the ARQ record on the RFIQueryRequest table should be maintained at the appropriate processing event points. This involves recording the start and conclusion, successful or otherwise, of each action in the appropriate RFISQL table. Upon encountering an error, the ARQ status is updated and the process terminated, awaiting user intervention.

The Query Request retrieves the details of the query and output type from the RFIQueryRequest table, and instantiates the Query Processor with the appropriate arguments.

Dependant upon the configuration setting, query type and data sources multiple instances of the Query processor may be instantiated.

If an existing query has previously been executed to completion for the ARQ , the Query Request Handler will perform a tidy up before continuing.

The tidy up will perform the following actions:

Delete all files from the ARQ QUERY_AT directory structure.

Remove rows relating to the ARQ from the following RFISQL database tables:

- RFIQueryFile
- RFIQueryFileSequence
- RFIQueryFileSequenceGap
- RFIQueryFileErrorMessage

Set the statuses of the rows relating to the ARQ to “Pending” on the following tables:

- RFIQueryRequest
- RFIQuery

FUJ00176545

For pending or incomplete queries the Query Request Handler will:

- If no processing of the ARQ has previous occurred, copy files from the ActiveFiles table to the RFIQueryFile table allocating them to the correct query, where applicable using the AuditSubPointExtended table to verify the datasource, based upon the file name.
- Determine the type of query(s) to be performed based upon the QueryType and OutputType field values in the RFIQueryRequest table.
- Instantiate the appropriate number or Query Processor threads as dictated by the Query Manager configuration file, and the query type.
- Record the termination code of each Query Processor upon completion.

6.3.3 Query Handler (AQH)

6.3.3.1 Interfaces

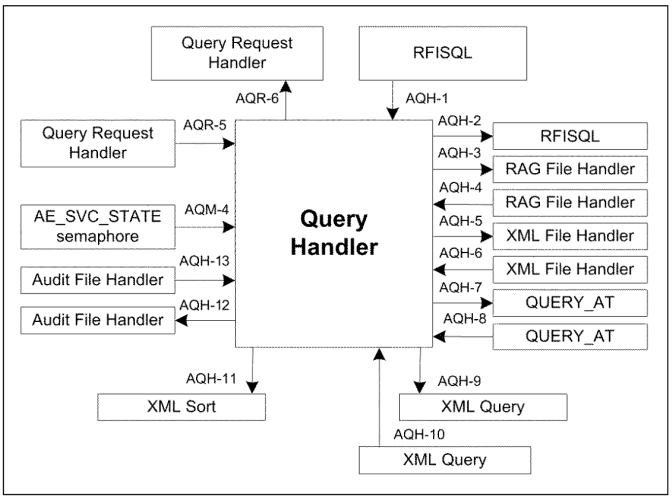

Figure 6 – Query Handler Interfaces

AQM-4 AE_SVC_STATE Semaphore

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

ARQ-5 Query Request Handler (Calls From)

Constructor

Receives the following arguments:

pRequest	Pointer to the Query Request Handler object.
pQueryId	Integer containing the QueryId
pMaxThreads	Integer containing the maximum number of concurrent file handler threads

FUJ00176545

Execute method

No arguments required.

AQR-6 Query Request Handler (Returns to)

Returns a Boolean value from the Query Handler object's Execute method.

True indicates successful completion.

False indicates an error was encountered.

AQH-1 RFISQL Database (Read)

RFIQueryFile	Retrieves list of files to be processed for the query
RFIQueryRequestFilter	Retrieves the query request filter to be used in abstraction
RFIAuditSubPointExtended	Retrieves Audit Sub Point permissible analysis options

AQH-2 RFISQL Database (Write)

RFIQuery	Updates query status at functional checkpoints
RFIQueryError	Inserts rows for each error encountered when applying query.
RFIQueryFile	Updates query file status at functional checkpoints

AQH-3 RAG File Handler (Calls to)

Constructor

pRequestId	Query Request Id. Integer.
pQueryId	Query Id. Integer
pQueryFile	Pointer to RFIQueryFile row for the current file.
pRFIReference	ARQ Reference. Char string

AQH-4 RAG File Handler (Returns from)

All returns are via public properties of the current RAG File Handler object.

NumberOfMessages	Number of messages processed
NumberOfBadMessages	Number of problem messages encountered
FileStatus	Status at completion of file processing

AQH-5 XML File Handler (Calls to)

Constructor

pRequestId	Query Request Id. Integer.
pQueryId	Query Id. Integer
pQueryFile	Pointer to RFIQueryFile row for the current file.
pRFIReference	ARQ Reference. Char string

AQH-6 XML File Handler (Returns from)

All returns are via public properties of the current XML File Handler object.

FUJ00176545

NumberOfMessages	Number of messages processed
NumberOfBadMessages	Number of problem messages encountered
FileStatus	Status at completion of file processing

AQH-7 QUERY_AT (Write)

QUERY_AT\MERGED	Writes merged abstracted files
QUERY_AT\XQUERY	Writes query results from merged files
QUERY_AT\FINAL	Writes sorted queried output.

AQH-8 QUERY_AT (Read)

QUERY_AT\ABSTACTED	Reads files abstracted for merging
QUERY_AT\MERGED	Reads merged files for querying
QUERY_AT\XQUERY	Reads queried files for sorting

AQH-9 XML Query (Calls to)

Constructor The XMLQuery constructor accepts two arguments from the QueryHandler component.

pQueryId	Integer QueryId value
pQuery	Character string containing the Query

ProcessAccepts one argument from the QueryHandler component:

IInputXML	String containing XML to be queried

AQH-10 XML Query (Returns from)

Returns a character string from the XMLQuery Process method containing the XML resulting from the Query. If a null value is returned, an error has been encountered during the application of the query

AQH-11 XML Sort (Calls to)

Constructor XML Sort constructor accepts two arguments from the Query Handler:

QueryId	Integer Query Id
pFilename	Pointer to file name of the intermediate file to sort.

Add Accepts the following arguments:

pFileOffset	Long value indicating the position of the message in the source file.
pMessage	The message to be sorted
pLength	Length of the message

BuildAndSave Accepts a pointer to the output file.

AQH-12 Audit File Handler (Calls to)

Constructor

pRequestId	Query Request Id. Integer.
pQueryId	Query Id. Integer
pQueryFile	Pointer to RFIQueryFile row for the current file.

FUJ00176545

pRFIReference	ARQ Reference. Char string

AQH-13 Audit File Handler (Returns from)

All returns are via public properties of the current Audit File Handler object.

NumberOfLines	Number of lines processed
FileStatus	Status at completion of file processing

6.3.3.2 Function

The Query Handler is responsible for managing progress through the filtering, abstraction, querying and sorting processes for a single query applied to a single ARQ.

Two query types are possible:

Message

This query type involves the analysis of Riposte Attribute Grammar query (Horizon data only) or XML query (HNG-X data only), and will relate to data from only one data source.

String Search

String searches may be applied to any data format

A single query request may not contain multiple query types.

At construction the Query Handler receives a pointer to the parent Query Request Handler object, The QueryId and the maximum number of concurrent threads that may be utilised in processing the files.

The ARQ Reference is retrieved from the RFIQuery table, and the type of query and output determined from the RFIQueryRequest table.

Upon successful retrieval of this data, the Query Request Handler will call the Query Handler's *Execute* method.

The Query Handler will determine the last completed stage in the filtering, abstraction and query process, and will begin execution at the next logical process.

All processes result in the generation of intermediate files which allow a part complete process to be restarted at the last accessed file within the process. Where this occurs, database table rows for the part processed file are removed and the intermediate file deleted. Processing then recommences at the start of the file.

6.3.3.2.1 Filtering and Abstraction

Filtering and abstracting is available for all data sources. If the query type is "String" this is the only analysis that may be performed on the data.

Rows containing details of the files to be processed by the Query Handler are retrieved from the RFIQueryFile table where the queryid matches the value provided at construction.

If the query type is "String"

For each file to be processed, an Audit File Handler is instantiated.

FUJ00176545

If the query type is "Message"

The data source of the query identified from the RFIQuery DatasourceId.

For queries against a Horizon data source:

- For each row retrieved from the RFIQueryFile table a RAG File Handler is instantiated passing the Query Request Id, QueryId, File name and ARQ Reference.

For Queries against a HNG-X data source.

- For each row retrieved from the RFIQueryFile table a XML File Handler is instantiated passing the Query Request Id, QueryId, File name and ARQ Reference.

If an error s encountered whilst processing a file, the RFIQuery row Status id updated accordingly and the query terminated.

Upon successful completion of file processing a final sequence check is performed.

This check identifies sequence gaps between files, and is based upon the sequences identified for files related to the query, ordered by FAD Code, CounterId and StartSequence number.

Rows are retrieved from the RFIQueryFileSequence table for the current query, and gaps identified where the SequenceEndNumber of the preceding row is not equal to the SequenceStartNumber of the current row -1. Identified gaps are written to the RFIQueryFileSequenceGap table, and the RFIQueryFile and RFIQuery statuses updated accordingly.

6.3.3.2.2 Merge

Upon successful completion of Filtering and abstraction, the intermediate files created in the *ARQ_Reference*\QUERY_AT\ABSTRACTED directory are merged.

For "String" query types files will be merged in file name order. As the date and time of creation are included in the file name, this will ensure chronological ordering of the data.

For "Message" query types files must be merged based upon their data source.

Horizon files are identified by the Hz extension, and HNG-X files by the Hx extension.

Within these groupings chronological ordering is once again maintained by file name ordering.

Merged files are written to the *ARQ_Reference*\QUERY_AT\MERGED directory

6.3.3.2.3 Query

This process is only applicable to queries of type "Message"

The XQuery to be applied is held in a .qry file in the ARQ's USERAREA directory.

The file will be named according to the data source it is to be applied to.

The paths and file names for the query file, the output file and the data stream merged file from the QUERY_AT\FINAL directory, are passed as arguments to the xqilla executable, which executes the query returning a code, as detailed below, indicating success or failure.

Exit code	Description
0	XQuery processing completed

FUJ00176545

1	Failed to process XQuery - Malformed command line
2	Failed to process XQuery - XQuery exception.
3	Failed to process XQuery - Memory Allocation Failure. The input file may be too large to parse.
4	Failed to process XQuery - XQilla Exception detected.
5	Failed to process XQuery - Malformed URL
6	Failed to process XQuery - Unknown XQilla exception
>6	Failed to process XQuery - Unknown Error

6.3.3.2.4 Events Filtering

If the IncludeEvents field in the Requests table (see section 7.3.19 “Requests” on page 102) is set to 1 (which is achieved by checking “Include Events” in the AuditClient) then Event Filtering is done.

The Perl script EventFilter.pl in the F:\UserArea\CommonScripts folder is called at the client filtering stage.

This script is passed the start date, end date and FAD code.

The script runs though all event files in the F:\UserArea\RFIREF\EXTRACTED_AT folder (where RFIREF is the variable RFI reference) and selects all events within the supplied date range and with the relevant FAD (or corresponding in addition to BAL events for the SYSMAN 3 queries) and passes any selected to the additional filters Sysman2Filter.pl for SYSMAN 2 events and Sysman3Filter.pm for SYSMAN 3 events. These scripts perform additional filtering based on advice from the SSC as to what events are relevant to financial transactions.

The output from this filtering is placed in the F:\UserArea\RFIREF\QUERY_AT\FINAL\ folder in 4 files as described in the table below –

File Name	Description
Sysman2Events.txt	Those events within the date range and with the correct FAD which pass the Sysman2Filter.pm rules.
RejectedSysman2Events.txt	As above but rejected by the Sysman2.pm rules.
Sysman3Events.txt	Those events within the date range and with the correct FAD (or are BAL events) which pass the Sysman3Filter.pm rules.
RejectedSysman3Events.txt	As above but rejected by the Sysman3.pm rules.

In addition EventFilter.pl stores the latest and earliest message date found. If it finds that the supplied earliest and latest dates are outside of this range it flags a fatal error.

Error information from the Perl script is stored in F:\UserArea\RFIREF\PerlErr.txt and run information in F:\UserArea\RFIREF\PerlOut.txt. Certain classes of error (for example the range check alluded to above) pass in error code back to the Client which alerts the user directly. Less critical errors are just logged in the files without directly alerting the user.

These Perl output files are then stored in the close log by CloseTidyLog.exe (see section 6.2.6 “CloseTidyLog.exe 2 on page 43) when the query is closed.

6.3.4 RAG File Handler (AQG)

6.3.4.1 Interfaces

FUJ00176545

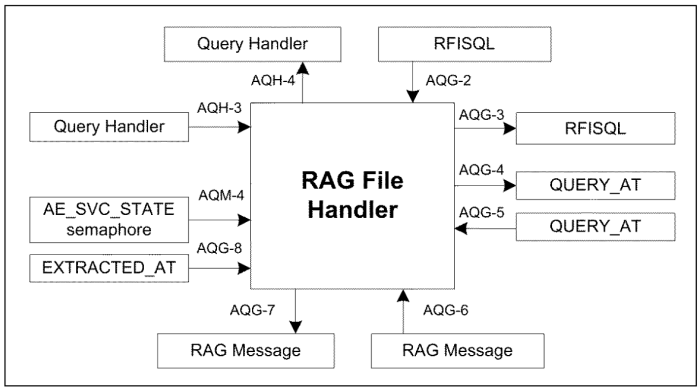

Figure 7 – RAG File Handler Interfaces

AQM-4 AE_SVC_STATE Semaphore

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQH-3 Query Handler (Calls from)

Constructor

pRequestId	Query Request Id. Integer.
pQueryId	Query Id. Integer
pQueryFile	Pointer to RFIQueryFile row for the current file.
pRFIReference	ARQ Reference. Char string

AQH-4 Query Handler (Returns)

NumberOfMessages	Number of messages processed
NumberOfBadMessages	Number of problem messages encountered
FileStatus	Status at completion of file processing

AQG-2 RFISQL (Read)

RFIQueryFile	Reads the row for the file being processed.

AQG-3 RFISQL (Write)

RFIQueryFile	Updates Status at relevant functional checkpoints
RFIQueryFileSequence	Adds row for each FAD/Counter sequence in the file
RFIQueryFileSequenceGap	Adds row for each gap in the FAD/Counter sequence encountered.

AQG-4 QUERY_AT (Write)

Deletes intermediate file for the current file from the ARQ's QUERY_AT\ABSTRACTED directory if found at construction.

Creates intermediate file in the ARQ's QUERY_AT\ABSTRACTED directory at construction.

FUJ00176545

Writes matching messages to the intermediate file in the ARQ's QUERY_AT\ABSTRACTED directory.

AQG-4 QUERY_AT (Read)

Checks for existence of intermediate file for the current file in the ARQ's QUERY_AT\ABSTRACTED directory.

AQG-6 RAG Message (Calls to)

Constructor

RAG Message object's constructor accepts a string containing the complete RAG message.

CheckCRC

No arguments required.

AQG-7 RAG Message (Returns from)

All returns are via the RAG Message object's public properties.

FAD Code	String	Originating Branch Code.
CounterID	Int	Originating counter within the branch
SequenceNo	Long	Message counter sequence number
Date	Date	Date of message generation
Time	Time	Time of message generation
CRC	String	Cyclic Redundancy Count for the message
Malformed	Bool	Indicates the message contains duplicated identifiers
Truncated	Bool	Indicates that the message has been truncated
CRCFail	Bool	Indicates the CRC does not match the message

AQG-8 EXTRACTED_AT (Read)

Specified file read from the appropriate ARQ's EXTRACTED_AT directory.

6.3.4.2 Function

The file handler will open the file specified in the constructor from the associate ARQ's EXTRACTED_AT directory.

Data source matching strings are generated from the ARQ's RepeatableSelectCriteria table row, where the matching Audit Sub-Point row in the RFIAuditSubPointExtended table is identified as originating on Horizon.

For each file the handler will:

- Check that the row for the file currently exist with the ARQ reference on the RFIQueryFile table, and process only where the status is not "Completed". If the row indicates partial completion intermediate files and database references should be removed, and processing restarted for the file.
- Open the file

FUJ00176545

- Update the appropriate a row in the RFIQueryFile table as Processing.
- Read data from the file one message at a time. A message is based upon a count of opening and closing braces.
- For each message
  - Instantiate a RAG Message handler
  - Retrieve Branch Code and Sequence details from the RAG Message handler
  - Compare Branch Code from the RAG Message Handler with filtration criteria.
  - If the message matches the primary filtration values
    - Check sequence continuity and record complete sequences to the RFIQueryFileSequence table on the RFISQL database.
    - Update the RFIQueryFile row FilterMatchFound to a value of 1
    - If the Query Output Type is Message
      - Write the message to the appropriate intermediate file.
- Upon reaching the end of the file
  - Close the file
  - Record the handler response values dependant upon the handler type
  - For message based filtration:
    - Record final sequence data to the RFIQueryFileSequence table
    - Record errors to the RFIQueryFileError table
  - Update the RFIQueryFile status with the appropriate status.

6.3.5 XML File Handler (AQX)

6.3.5.1 Interfaces

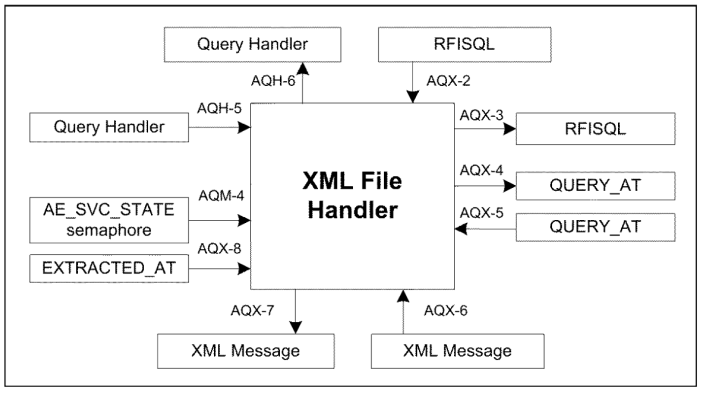

Figure 8 – XML File Handler Interfaces

AQM-4 AE_SVC_STATE Semaphore

FUJ00176545

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQH-5 Query Handler (Calls from)

Constructor

pRequestId	Query Request Id. Integer.
pQueryId	Query Id. Integer
pQueryFile	Pointer to RFIQueryFile row for the current file.
pRFIReference	ARQ Reference. Char string

AQH-6 Query Handler (Returns to)

NumberOfMessages	Number of messages processed
NumberOfBadMessages	Number of problem messages encountered
FileStatus	Status at completion of file processing

AQX-2 RFISQL (Read)

RFIQueryFile	Reads the row for the file being processed.

AQX-3 RFISQL (Write)

RFIQueryFile	Updates Status at relevant functional checkpoints
RFIQueryFileSequence	Adds row for each FAD/Counter sequence in the file
RFIQueryFileSequenceGap	Adds row for each gap in the FAD/Counter sequence encountered.

AQX-4 QUERY_AT (Write)

Deletes intermediate file for the current file from the ARQ's QUERY_AT\ABSTRACTED directory if found at construction.

Creates intermediate file in the ARQ's QUERY_AT\ABSTRACTED directory at construction.

Writes matching messages to the intermediate file in the ARQ's QUERY_AT\ABSTRACTED directory.

AQX-5 QUERY_AT (Read)

Checks for existence of intermediate file for the current file in the ARQ's QUERY_AT\ABSTRACTED directory.

AQX-6 XML Message (Calls to)

Constructor

XML Message object's constructor accepts a string containing the complete XML message.

CheckDigitalSignature

Accepts a string containing the public key required for digital signature checking.

FUJ00176545

AQX-7 XML File Handler (Returns from)

All returns are via XML Message object's public properties.

FAD Code	String	Originating Branch Code.
CounterID	Int	Originating counter within the branch
SequenceNo	Long	Message counter sequence number
Date	Date	Date of message generation
Time	Time	Time of message generation
DigitalSignature	String	Digital Signature for the message
Malformed	Bool	Indicates the message contains duplicated identifiers
Truncated	Bool	Indicates that the message has been truncated
DSFail	Bool	Indicates the Digital Signature does not match the message

AQX-8 EXTRACTED_AT (Read)

Specified file read from the appropriate ARQ's EXTRACTED_AT directory.

6.3.5.2 Function

The handling of native XML message is similar to that of RAG messages. There is however one significant difference; In order to validate a message from it's digital signature, the public key must be available. The public key will be generated at the counter at the start of each user session and passed to Audit in a distinct message.

It is therefore necessary to capture these messages and maintain a list of active public keys for each counter at each included branch code. The absence of the public key within the XML message to be validated forces a two pass approach to validating XML messages, and dictates a slightly different processing sequence.

The XML File Handler will retrieve the file specified at construction from the associated EXTRACTED_AT directory. The source of the file is determined from the file name's embedded Audit Sub Point.

The file source is matched against the ARQ's RepeatableSelectCriteria rows, and is confirmed as originating from HNG-X by matching the Audit Sub-Point with the matching Audit Sub-Point row in the RFIAuditSubPointExtended table.

For each file the handler will:

- Check that the row for the file currently exist with the ARQ reference on the RFIQueryFile table, and process only where the status is not "Completed". If the row indicates partial completion intermediate files and database references should be removed, and processing restarted for the file.
- Open the file
- Update the appropriate row in the RFIQueryFile table as Processing.
- Read data from the file one message at a time. A message is based upon a count of opening and closing braces.
- For each message

FUJ00176545

- Instantiate an XML Message handler
- Validate the XML message
- Retrieve branch, counter, session and Sequence details from the XML Message handler
- Compare branch details from the XML Message Handler with filtration criteria.

If the message matches the filtration criteria

- Check for the presence of the Public Key Attribute.

If the Message contains the Public Key Attribute:

- Compare the Branch/Counter details against the list of active public keys

If the Branch/counter combination does not exist, add it, and the public key to the list. If the combination already exists, replace the public key with the new value.

If the Message does not contains the Public Key Attribute:

- Call the "CheckDigitalSignature" method of the XML Message handler, passing the current active public key from the list. If there is no current active public key, terminate the query recording the error in the RFIQueryError table.

- If the digital signature confirms the state of the message
  - Check sequence continuity and record complete sequences to the RFIQueryFileSequence table on the RFISQL database.
  - Update the RFIQueryFile FilterMatchFound field with a value of 1

  If the Query Output Type is "Message"
  - Write the message to the appropriate intermediate file.

  If the Query Output Type is "File"
  - Exit from the XML File Handler

- Upon reaching the end of the file
  - Close the file
  - Record the handler response values dependant upon the handler type
  - Record final sequence data to the RFIQueryFileSequence table
  - Record errors to the RFIQueryFileError table
  - Update the RFIQueryFile status with the appropriate status.

6.3.6 Audit File Handler (AQA)

6.3.6.1 Interfaces

FUJ00176545

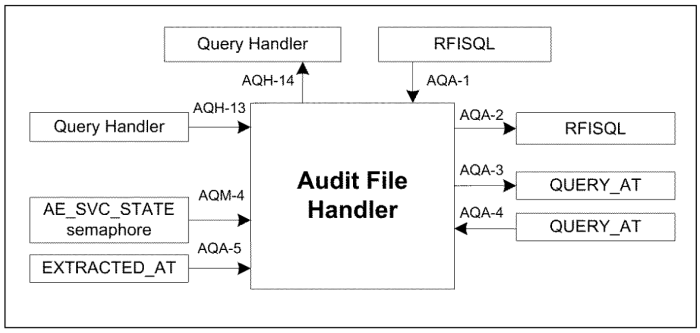

Figure 9 – Audit File Handler Interfaces

AQM-4 AE_SVC_STATE Semaphore

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQH-13 Query Handler (Calls from)

Constructor

pRequestId	Query Request Id. Integer.
pQueryId	Query Id. Integer
pQueryFile	Pointer to RFIQueryFile row for the current file.
pRFIReference	ARQ Reference. Char string

AQH-14 Query Handler (Returns to)

NumberOfLines	Number of lines processed
FileStatus	Status at completion of file processing

AQA-1 RFISQL (Read)

RFIQueryFile	Reads the row for the file being processed.
RFIQueryRequestFilter	Reads the filters to be applied to each line.

AQA-2 RFISQL (Write)

RFIQueryFile	Updates Status and FilterFound fields at relevant functional checkpoints

AQA-3 QUERY_AT (Write)

Deletes intermediate file for the current file from the ARQ's QUERY_AT\ABSTRACTED directory if found at construction.

Creates intermediate file in the ARQ's QUERY_AT\ABSTRACTED directory at construction.

Writes matching messages to the intermediate file in the ARQ's QUERY_AT\ABSTRACTED directory.

AQA-4 QUERY_AT (Read)

FUJ00176545

Checks for existence of intermediate file for the current file in the ARQ's QUERY_AT\ABSTRACTED directory.

AQA-5 EXTRACTED_AT (Read)

Specified file read from the appropriate ARQ's EXTRACTED_AT directory.

6.3.6.2 Function

The Audit File Handler implements basic string searching of a file on a line by line basis.

While this form of matching may be applied to any file type, it is assumed that no further querying of the data is possible.

The Audit File Handler retrieves the file to be processed, as provided to the constructor, for the ARQ from the associated EXTRACTED_AT directory. No separation of data by data source is required.

For each file the handler will:

- Check that a row exists for the file exists for the ARQ on the RFIQueryFile table
- Open the file
- Update the appropriate row in the RFIQueryFile table as Processing.
- Read data from the file one line at a time. A line is considered to be complete when either a line feed or end of file is encountered
- For each line
    - Perform a String Search for each of the criteria specified in the RFIQueryRequestFilter for the QueryRequestId supplied at contruction.
    - If a match is found and the Query Output type is "Files"

      Mark the file as matching in the RFIQueryFile table

    - If a match is found and the Output type is "Lines"

      Extract the line to an intermediate file

6.3.7 RAG Message Handler

6.3.7.1 Interfaces

FUJ00176545

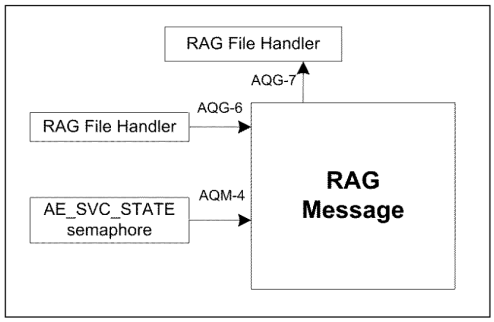

Figure 10 – RAG Message Interfaces

AQM-4 AE_SVC_STATE Semaphore

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQG-6 RAG File Handler (Calls)

Constructor

RAG Message object's constructor accepts a string containing the complete RAG message.

CheckCRC

No arguments required.

AQG-7 RAG File Handler (Returns)

All returns are via RAG Message object's public properties.

FAD Code	String	Originating Branch Code.
CounterID	Int	Originating counter within the branch
SequenceNo	Long	Message counter sequence number
Date	Date	Date of message generation
Time	Time	Time of message generation
CRC	String	Cyclic Redundancy Count for the message
Malformed	Bool	Indicates the message contains duplicated identifiers
Truncated	Bool	Indicates that the message has been truncated
CRCFail	Bool	Indicates the CRC does not match the message

6.3.7.2 Function

The RAG Message handler receives the ARQ reference and the message to be processed from the RAG File Handler.

The RAG Message handler performs the following processing:

1. Validation of the RAG message structure
2. Validation against invalid XML character specifications

   The following rules are applied:

FUJ00176545

RAG must belong to one of the following groupings:

":" or "_" or "–"

0-9

A-Z

a-z

0xC0 - 0xD6

0xD8 - 0xF6

> 0xF8

0xB7

Any character falling outside of these ranges are replaced by "_x%2.2x_" where "x" is the integer representation of the character to be replaced.

3. Extracting Branch and sequence details
4. Checking that the message CRC is valid
5. Converting the message to XML

The first 3 steps may be performed sequentially

6.3.7.2.1 Validating the RAG message

The RAG message is validated against all of the following rules:

- Contain an equal number of open and close braces ("<" ">")
- Contain a single GroupID attribute
- Contain a single ID attribute
- Contain a single NUM attribute
- Contain at least one CRC attribute
- Contain a numeric values in GroupID, ID and NUM attributes
- Have a CRC attribute as the last attribute in the message

The result of the validation process should be held in a state that may be accessed by the RAG File Handler.

Upon failing validation the RAG message handler should return control to the RAG File Handler without performing any further processing.

6.3.7.2.2 Extracting Sequence details

The following sequence details are extracted from the RAG message and maintained in a format accessible to the RAG File Handler:

FAD code	- From the GroupID attribute
Counter number	- From the ID attribute
Sequence number	- From the NUM attribute

FUJ00176545

6.3.7.2.3 CRC Checking

A RAG message may have multiple CRC attributes. Only the last CRC attribute value should be used to check the message integrity.

The CRC should be generated against the body of the message, and should match the value of the messages CRC attribute.

The result of the CRC check process should be held in a state that may be accessed by the File Handler.

Upon failing the check the RAG message handler should return control to the File Handler without performing any further processing.

6.3.7.2.4 Converting RAG to XML

RAG formatted data is converted to simplest form XML. This is achieved by the counting of the message braces and the maintenance of an attribute array which is used for determining which attribute is next to be closed.

Additionally RAG attribute names are checked to ensure that they conform to XML rules. This chiefly involves checking that the attribute name does not begin with a whitespace or period and prefixing with "AUDIT$" where encountered.

Riposte Attribute Grammar also allows the use of non-printable control characters, and characters outside the ASCII range 0-127, in attribute data. These cannot be represented directly as XML content, and are replaced with &#xnn;

6.3.8 XML Message Handler (AQL)

6.3.8.1 Interfaces

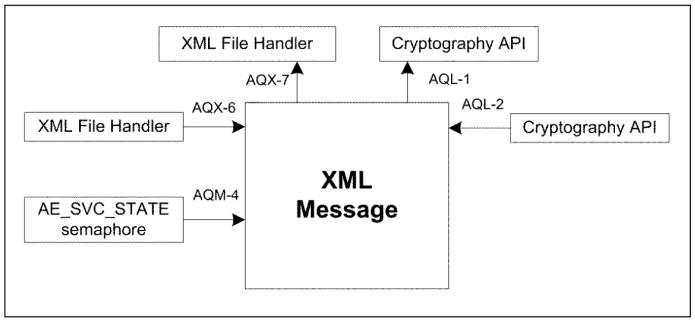

Figure 11 – XML Message Interfaces

AQM-4 AE_SVC_STATE Semaphore

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQX-6 XML File Handler (Calls)

Constructor

XML Message object's constructor accepts a string containing the complete XML message.

FUJ00176545

CheckDigitalSignature

Accepts a string containing the public key required for digital signature checking.

AQX-7 XML File Handler (Returns)

All returns are via XML Message object's public properties.

FAD Code	String	Originating Branch Code.
CounterID	Int	Originating counter within the branch
SequenceNo	Long	Message counter sequence number
Date	Date	Date of message generation
Time	Time	Time of message generation
DigitalSignature	String	Digital Signature for the message
Malformed	Bool	Indicates the message contains duplicated identifiers
Truncated	Bool	Indicates that the message has been truncated
DSFail	Bool	Indicates the Digital Signature does not match the message

AQL-1 Cryptography API (Calls)

AQL-2 Cryptography API (Returns)

6.3.8.2 Function

The XML Message handler receives the ARQ reference and the message to be processed from the XML File Handler.

The XML Message handler performs the following processing:

1. Parse the message to separate the HTTP header from the XML message.
2. Read required data from the HTP header
3. Abstract the XML message, or embedded message
4. Validation of the XML message structure
5. Extracting Branch, counter, sequence and session details
6. Signal the existence of the PubliKey attribute within the message for second logon events.
7. Checking that the messages Digital Signature is valid

The first 3 steps may be performed sequentially

6.3.8.2.1 Validating an XML Message

The XML message is validated against all of the following rules:

- Contain an equal number of open and close braces ("<" ">")

FUJ00176545

- Contain a single BranchID attribute
- Contain a single CounterID attribute
- Contains a single SessionID attribute
- Contain a single Sequence attribute
- Contain at least one Digital Signature attribute
- Contain a numeric values in BranchID, CounterID, Sequence and Session attributes
- Check for the existence of a Public Key attribute within the message

The results of the validation process should be held in a state that may be accessed by the XML File Handler.

6.3.8.2.2 Extracting Attribute details

The following sequence details are extracted from the XML message and maintained in a format accessible to the XML File Handler:

Branch code	- From the GroupID attribute
Counter number	- From the ID attribute
Session number	- From the SessionId attribute
Sequence number	- From the NUM attribute
Public Key	- From the Public Key attribute

6.3.8.2.3 Digital Signature checking

Digital signature checking is instigated from the XML File Handler, and requires the public key for the active Branch/Counter/Session.

This process should be accessible via a “CheckDigitalSignature” public function.

6.3.9 XML Query (AQQ)

6.3.9.1 Interfaces

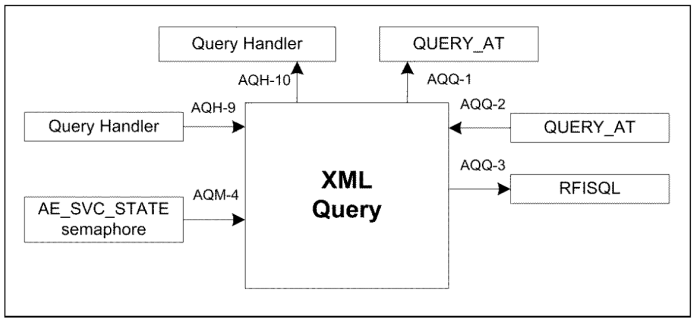

Figure 12 – XML Query Interfaces

AQM-4 AE_SVC_STATE

FUJ00176545

Semaphore created by the Audit Query Manager. Used to signal immediate termination to components.

AQH-9 Query Handler (Calls)

Constructor The XMLQuery constructor accepts two arguments from the QueryHandler component.
pQueryId Integer QueryId value
pQuery Character string containing the Query

ProcessAccepts one argument from the QueryHandler component:
IInputXML String containing XML to be queried

AQH-10 Query Handler (Returns)

Returns a character string from the XMLQuery Process method containing the XML resulting from the Query.

AQQ-1 QUERY_AT (Write)

Writes intermediate file to D:\UserArea\ARQReference\QUERY_AT\XQUERY

AQQ-2 QUERY_AT (Read)

Reads intermediate files from D:\UserArea\ARQReference\QUERY_AT\MERGED

AQQ-3 RFISQL database (Write)

Writes query errors to the RFIQueryError table of the RFISQL database

6.3.9.2 Function

The XML Query component utilises COTS XQuery tools to compare and abstract data from the concatenated intermediate files.

Where the ARQ contains data originating from multiple data sources, an XML query is required for each data source, and will be instantiated from the Query Handler.

- The XQuery standard allows for the analysis and sorting of data within a single query.

The query is executed by calling the xqilla command line interface passing the source data file, the query file and the location of the output file as arguments.

The result code from the xqilla executable is handled, and where applicable errors recorded in the RFIQueryError table.

Exit code	Description
0	XQuery processing completed
1	Failed to process XQuery - Malformed command line
2	Failed to process XQuery - XQuery exception.
3	Failed to process XQuery - Memory Allocation Failure. The input file may be too large to parse.
4	Failed to process XQuery - XQilla Exception detected.
5	Failed to process XQuery - Malformed URL

FUJ00176545

6	Failed to process XQuery - Unknown XQilla exception
>6	Failed to process XQuery - Unknown Error

6.1.1.1.1 Generation and Application of XQuery

The set of common queries are held in the F:\USERAREA\CommonQueries directory.

For an individual ARQ the query or queries to be executed will be located in the root of the ARQs directory in the F:\USERAREA structure.

The query is identified by the name which will either be Horizon.qry or HNG-X.qry, dependant upon the data source against which the query is to be executed.

Queries are executed using the xqilla XQuery tools which utilise the xerces XML handling components.

FUJ00176545

7 Database Design

7.1 Overview

Each Audit server runs a single instance of Microsoft SQL Server 2000, providing access to the two Audit databases:

SEALERSQL Contains data generated during the gathering, sealing, seal checking and cross campus distribution processes. Only server based components write to or delete from this database.

RFISQL Contains data generated and maintained throughout the day to day operation of the Audit system. Both server (core and non-core) and client components perform CRUD operations on this database.

.

As both databases are accessed by a mixture of core Audit services, extraction components and Audit work station applications, alterations to the existing database schema are difficult and time consuming to impact accurately. Additionally, data in a number of tables can be periodically refreshed from external sources, rendering any changes to the table contents temporary in nature.

To minimise the impact of changes to the database, a policy of addition only has been adopted. In practice this will be implemented by adding new tables, stored procedures and triggers as opposed to altering existing entities. This approach has the advantage of having no impact upon those parts of the Audit system that do not change for HNG-X.

It is worth noting that the database schemas for both SEALERSQL and RFISQL, upon which the new schemas are based, have changed little since their initial implementation. Subsequently terminology in use at the time of their creation is still present in the current version. The most notable example of which is the prevalence of the acronym RFI (Request For Information) which was superseded by the current ARQ (Audit Request Query). To avoid confusion all additions to the database will conform to the existing database nomenclature, therefore RFI will be used in preference to ARQ.

Both schemas also contain what would appear upon first inspection to be functionally redundant columns, the majority of which relate to the original Legato tape archive system. In many instances however these columns have been retained to support audit track retention period management, around which the original tape pools were organised. Determining which of these columns could be removed from the schema would require a disproportional amount of effort, with no discernable benefits.

7.2 SEALERSQL Database

The SealerSQL database is utilised to maintain details of all files currently held on the local EMC Centera, along with any associated information gathered at the time of the file being added to the archive.

Once added, the data is retained in it's original form until it is deleted at the expiry of its retention period.

FUJ00176545

During the life time of this data it will be regarded as the *de facto* reference by both core and non-core server processes.

7.2.1 Check Seal table

Detail	
Status	Existing
Description	Contains a row for each seal check action against an Audit Track.

Table Structure		
Column	Type	Description
Audit Track	varchar(255)	Audit track data file on which the seal algorithm has been applied.
Seal Value	varchar(40)	Seal value as generated by algorithm named in 'Seal algorithm name'
Seal algorithm name	varchar(30)	Name of the seal algorithm that has been used to seal this audit track.
Generation Mode	int	Increment number for the seal algorithm name. (ie. MD5 = 1, MD5V2 = 2, SEALXYX = 3 etc.)
Seal generation result code	float	Result of running the seal algorithm. (0 = OK, +n = failure result code)
Local generation	bit	Was this seal value generated by the local Archive Server or imported from the remote Archive Server (Yes = Local, No = Remote)
Date generated	smalldatetime	Date on which the calculation of the seal value was completed.
Time Generated	datetime	Time on which the calculation of the seal value was completed.
Comparison request ID	varchar(50)	Identifier of the comparison requester.
Seals match	bit	Result of comparing the 'Seal value' from the Initial Seal table with the 'Seal value' from this Check Seal table

Trigger		
On	**Name**	**Function**
Insert	trgCheckSealTblInsert	This trigger is required to update the Seal check status of the RFISQL.ActiveFiles row to which the inserted CheckSeal row relates.

FUJ00176545

7.2.2 Duplicate Initial Track table

Detail	
Status	Existing
Description	Contains a row for each attempt to add an audit track to the [Initial Track table] which fails due to the pre-existence of the audit track.

Table Structure

Column	Type	Description
Audit Track	varchar(255)	Audit track data file name. (Duplicate of main indexed field in the 'Initial Track table'.
Seal Value	varchar(40)	Seal value as generated by 'Seal algorithm name' on 'Audit track'
Seal algorithm name	varchar(30)	Name of the seal algorithm that has been used to seal this audit track.
Generation Mode	int	Increment number for the seal algorithm name. (ie. MD5 = 1, MD5V2 = 2, SEALXYX = 3 etc.)
Seal generation result code	float	Result of running the 'Seal algorithm name' on 'Audit track'. (0 = OK, +n = failure result code)
Local generation	bit	Was this seal value generated by the local Archive Server or imported from the remote Archive Server (Yes = Local, No = Remote)
Date generated	smalldatetime	Date on which the calculation of the seal value was completed.
Time generated	datetime	Time on which the calculation of the seal value was completed.
From date	smalldatetime	Date on which this unique audit track was first passed to the Archive Server 'sealer' component .
From time	datetime	Time on which this unique audit track was first passed to the Archive Server 'sealer' component .
Until date	smalldatetime	Calculated date after which ALL records (in all tables) relating to this audit track can be deleted.
Until time	datetime	Calculated time after which ALL records (in all tables) relating to this audit track can be deleted.
Retention period	smallint	Number of days after the 'From' date when ALL records (in all tables) relating to this audit track can be deleted.
FileSize	int	Size of file in bytes
StorageType	varchar(4)	Type of storage to which the file has been archived. D – Tape, C - Centera
CenteraID	varchar(100)	EMC Centera Clip ID for this file

FUJ00176545

7.2.3 Initial Track table

Detail	
Status	Existing
Description	Contains an entry for each audit track added to the EMC Centera

Table Structure

Column	Type	Description
Audit Track	varchar(255)	Audit track data file name. (Main indexed field)
Seal Value	varchar(40)	Seal value as generated by 'Seal algorithm name' on 'Audit track'
Seal algorithm name	varchar(30)	Name of the seal algorithm that has been used to seal this audit track.
Generation Mode	int	Increment number for the seal algorithm name. (ie. MD5 = 1, MD5V2 = 2, SEALXYX = 3 etc.)
Seal generation result code	float	Result of running the 'Seal algorithm name' on 'Audit track'. (0 = OK, +n = failure result code)
Local generation	bit	Was this seal value generated by the local Archive Server or imported from the remote Archive Server (Yes = Local, No = Remote)
Date generated	smalldatetime	Date on which the calculation of the seal value was completed.
Time generated	datetime	Time at which the calculation of the seal value was completed.
From date	smalldatetime	Date on which this unique audit track was first passed to the Archive Server 'sealer' component .
From time	datetime	Time at which this unique audit track was first passed to the Archive Server 'sealer' component .
Until date	smalldatetime	Calculated date after which ALL records (in all tables) relating to this audit track can be deleted.
Until time	datetime	Calculated time after which ALL records (in all tables) relating to this audit track can be deleted.
Retention period	smallint	Number of days after the 'From' date when ALL records (in all tables) relating to this audit track can be deleted.
FileSize	int	Size of file in bytes
StorageType	varchar(4)	Type of storage to which the file has been archived. D – Tape, C - Centera
CenteraID	varchar(100)	EMC Centera Clip ID for this file

7.2.4 No Initial Track table

FUJ00176545

Detail	
Status	Existing
Description	The purpose of this table is not known. It is thought to relate the migration exercise from Legato tapes to the EMC Centeras. No data appears to have been added to this table for several years.

Table Structure		
Column	Type	Description
Audit Track	varchar(255)	Audit track data file name.
Seal Value	varchar(40)	Seal value as generated by 'Seal algorithm name' on 'Audit track'
Seal algorithm name	varchar(30)	Name of the seal algorithm that has been used to seal this audit track.
Generation Mode	int	Increment number for the seal algorithm name. (ie. MD5 = 1, MD5V2 = 2, SEALXYX = 3 etc.)
Seal generation result code	float	Result of running the 'Seal algorithm name' on 'Audit track'. (0 = OK, +n = failure result code)
Local generation	bit	Was this seal value generated by the local Archive Server or imported from the remote Archive Server (Yes = Local, No = Remote)
Date generated	smalldatetime	Date on which the calculation of the seal value was completed.
Time generated	datetime	Time at which the calculation of the seal value was completed.
Comparison request ID	varchar(50)	Identifier of the comparison requester.
Seals Match	bit	Result of comparing the 'Seal value' from the Initial Seal table with the 'Seal value' from this Check Seal table

7.3 RFISQL Database

The RFI database is utilised as the general purpose data repository in the execution of all daily activities including the creation, management, execution and archival of requests for information.

Additionally, asynchronous execution of server-side components is achieved through insert triggers on the various RFISQL tables.

The full RFISQL database schema is attached at Appendix A.

FUJ00176545

7.3.1 ActiveFiles

Detail	
Status	Existing
Description	Files associated with a Request row. Populated from the [Initial Track Table] table on the SEALERSQL database with rows matching the selection criteria held in the associated RepeatableSelectCriteria entry. Rows may be manually removed by the user prior to execution of the extraction request, to reduce the number of files retrieved. Upon execution the remaining rows associated with the ARQ are used to determine the extraction requirement. The Status column will be updated at each stage of the extraction and filtering process.

Table Structure

Column	Type	Description
FileName	varchar(255)	Audit Track from [Initial Track table] on the SEALERSQL database
RFI Reference	varchar(50)	ARQ reference to which the row relates.
VolumeName	varchar(50)	Tape to which audit track was archived
TapePool	varchar(50)	Tape pool of tape. Used to reconstruct the full path of the archived audit track.
AuditPoint	varchar(50)	Audit point associated with the audit track. Used to reconstruct the full path of the archived audit track.
Status	varchar(50)	Last known status of the file. Allowed values from ActiveFileStatusValues table.
FileSize	int	Size of file in bytes
Volume2	varchar(50)	
CenteralD	varchar(100)	Audit track clip ID from the EMC Centera
StorageType	varchar(4)	How the audit track is stored. Currently all audit tracks will have a value of 'C'

Trigger

On	Name	Function
Insert and Update	trgActiveFilesInsUpd	Updates the Status attribute of the RFISQL.Requests table entry related to the inserted/updated row based upon the lowest Status across all ActiveFiles rows for the relevant ARQ. This was amended at CP0336 to detect when all Fast ARQ files have gone to SEALOK status and then update the associated QueryRequest entry to start the Filtering and Query processing.

7.3.2 ActiveFileStatusValues

Detail	
Status	Existing

FUJ00176545

Description	Status values that may be applied to an ActiveFile row.

Table Structure

Column	Type	Description
FileStatus	varchar(50)	File status. (See Values table below)
Comment	varchar(100)	Supporting comments.

Values

FileStatus	Meaning
CenteraFail	The Centera has accepted the request but has failed to invoke recovery
Displayed	File has been identified in the SEALERSQL database as matching the extraction criteria.
LegatoFail	No longer used
Requested	Retrieval of the file has been initiated.
Restored	Restored to Archive server but not yet processed by the Retriever core component.
RestoreFailure	File is not on archive server and an error message has been reported by the Centera.
Restoring	File not found on archive server and no error message has been reported. It is assumed that the file is being moved as part of the restore process.
SealFailure	Seal value generated on this file does not match the value generated at the time of it's initial seal creation
SealOk	Seal value generated on this file matches the value generated at the time of it's initial seal creation.
TapeOffline	No longer used
Unknown	Centera SDK reported an unknown or unhandled error code.
WaitingForTape	No longer used
WaitingSealCheck	File has been processed by the Retriever component but not by the Sealer component.

7.3.3 AuditPoint

Detail

Status	Existing
Description	Contains all active Audit Points. This table may be periodically refreshed from an external source.

Table Structure

Column	Type	Description
AuditPoint	varchar(50)	Audit Point name
TapePool	varchar(50)	Tape pool to which the Audit Point belongs. This is a legacy column relating to the now redundant Legato tape system.

FUJ00176545

FUJ00176545

7.3.4 AuditSubPoint

Detail	
Status	Existing
Description	Contains all active Audit Sub Points. This table may be periodically refreshed from an external source.

Table Structure		
Column	Type	Description
Audit Sub Point	varchar(50)	Audit Sub Point name which maps to a physical network location via the Audit server configuration file.
Audit Point	varchar(50)	Audit Point to which the Audit Sub Point belongs

7.3.5 AWOperators

Detail	
Status	Existing
Description	Authorised Audit Extractor users. Contains details of network logon and access status.

Table Structure		
Column	Type	Description
AW Operator ID	varchar(50)	Name of the operator, derived from the domain user
Machine ID	varchar(50)	Machine from which the system was last accessed
Logged On	bit	Current logon status. 0 – Logged out, 1 – Logged on
Last Logon	datetime	Date and time at the start of the last access.
Last Logoff	datetime	Date and time at the end of the last access
Concurrent Logon Count	smallint	Number of concurrent sessions currently active

7.3.6 ClosedRFI

Detail	
Status	Existing
Description	Long term record of closed requests.

Table Structure		
Column	Type	Description
RFI Reference	varchar(50)	RFI Reference of the request.
Instance	int	This is the identifier for the instance of the application which closed the ARQ.

FUJ00176545

7.3.7 ClosedRFIActions

Detail	
Status	Existing
Description	Long term record of actions performed during the life of a request. Taken from the FileFoundConsole table upon closure of an active request.

Table Structure		
Column	Type	Description
RFI Reference	varchar(50)	ARQ Reference of the request.
Instance	int	This is the identifier for the instance of the application which created the original action.
CmdUsedID	varchar(50)	Either : The ID of the command from the CmdUsedIndex table which resulted in the action. or The command description.
ErrCode	int	Internal error code resulting from this action.
StateID	int	State ID of the action from the StateIndex table
StartAt	datetime	Date and time at which the action was executed.
EndAt	datetime	Date and time at which the action completed.
ErrMessage	varchar(255)	Either: The error message generated by an action failure. or Information relating to the action, but not resulting from an error. This column can not be relied upon as an indication that an error has occurred.
GeneratedBy	varchar(255)	Operator name from AWOperators

7.3.8 ClosedRFIFiles

Detail	
Status	Existing
Description	Long term record of files associated with a closed request. Rows are copied from the ActiveFiles table at closure of an active request.

Table Structure		
Column	Type	Description
RFI Reference	varchar(50)	Reference from Requests

FUJ00176545

RFICreatedBy	varchar(50)	Operator name from AWOperators
FileName	varchar(255)	Name of the file from ActiveFiles
Status	varchar(50)	Textual description of the file status at the time of closure.

7.3.9 CmdUsedIndex

Detail

Status	Existing
Description	Details of recordable request commands (referred to as actions elsewhere in the database).

Table Structure

Column	Type	Description
CmdUsedID	int	Unique command identifier.
CmdUsed	varchar(50)	Description of command

Values

CmdUsedId	CmdUsed
1	Recover files
2	File Status
3	Volume status
4	File sizes
5	Clear message store
6	Generate message store
7	Get cluster ID
8	Create directory structure
9	Tidy filestore
10	Update Audit points
11	Request analysis
12	Close
13	Centera daily purge
14	Build Oracle Table

7.3.10 CommonControl

Detail

Status	Existing
Description	Contains a single row. Audit extraction system default values.

FUJ00176545

FUJ00176545

Table Structure		
Column	Type	Description
UserDrive	varchar(1)	Identifies the drive where message stores are to be created.
MS_BuildOnServer	varchar(50)	A flag to record where a message-store is to be created. 'FREE' or an ARQ reference other than the users current ARQ reference forces the message-store to be created on the users workstation. If the value is equal to the users current ARQ reference, it indicates creation of the message-store is to be on the server.
Version Control	varchar(10)	Referenced by AuditExtractor client application to check that latest release is being used.
PlatformID	varchar(30)	Name of the Audit server
PathOnly	varchar(255)	Default location for files to be hoarded.
TimeoutRecover	smallint	Recovery timeout value in minutes
TimeoutVolume	smallint	Volume timeout value in minutes
TimeoutFileSizes	smallint	FileSizes timeout value in minutes
TimeoutTMSClear	smallint	TMSClear timeout value in minutes
TimeoutTMSGenerate	smallint	TMSGenerate timeout value in minutes
TimeoutClusterID	smallint	ClusterId timeout value in minutes
TimeoutTidyFileStore	smallint	TidyFileStore timeout value in minutes
SealDBTimeout	smallint	Default timeout in minutes for accesses to the SEALERSQL database.

7.3.11 DailyDeleteCenteraDupInitialTrack

Detail	
Status	Existing
Description	Contains a row for each audit track in the SEALERSQL [Duplicate Initial Track table] that has reached or passed it's expiry date. Populated daily by the PopulateRFICenteraDelete scheduled job (see section 6.1.4.1).

FUJ00176545

Table Structure		
Column	Type	Description
Audit track	varchar(255)	Audit Track from [Duplicate Initial Track table] on the SEALERSQL database
Seal value	varchar(40)	Seal value as generated by 'Seal algorithm name' on 'Audit track'
Seal algorithm name	varchar(30)	Name of the seal algorithm that has been used to seal this audit track.
Generation mode	int	Increment number for the seal algorithm name. (ie. MD5 = 1, MD5V2 = 2, SEALXYX = 3 etc.)
Seal generation result code	float	Result of running the 'Seal algorithm name' on 'Audit track'. (0 = OK, +n = failure result code)
Local generation	bit	Was this seal value generated by the local Archive Server or imported from the remote Archive Server (Yes = Local, No = Remote)
Date generated	smalldatetime	Date on which the calculation of the seal value was completed.
Time generated	datetime	Time at which the calculation of the seal value was completed.
From date	smalldatetime	Date on which this unique audit track was first passed to the Archive Server 'sealer' component .
From time	datetime	Time at which this unique audit track was first passed to the Archive Server 'sealer' component .
Until date	smalldatetime	Calculated date after which ALL records (in all tables) relating to this audit track can be deleted.
Until time	datetime	Calculated time after which ALL records (in all tables) relating to this audit track can be deleted.
Retention period	smallint	Number of days after the 'From' date when ALL records (in all tables) relating to this audit track can be deleted.
FileSize	int	Size of file in bytes
StorageType	varchar(4)	Type of storage to which the file has been archived. T – Tape, C - Centera
CenteraID	varchar(100)	EMC Centera Clip ID for this file
DeleteFail	varchar(10)	

7.3.12 DailyDeleteCenteraInitialTrack

Detail	
Status	Existing
Description	Contains a row for each audit track in the SEALERSQL [Initial Track table] that has reached or passed it's expiry date. Populated daily by the PopulateRFICenteraDelete scheduled job (see Section 6.1.4.1),

FUJ00176545

	and subsequently used by the CenteraDelete non-core server component as basis for the removal of audit tracks from the.

Table Structure		
Column	Type	Description
Audit track	varchar(255)	Audit Track from [Initial Track table] on the SEALERSQL database
Seal value	varchar(40)	Seal value as generated by 'Seal algorithm name' on 'Audit track'
Seal algorithm name	varchar(30)	Name of the seal algorithm that has been used to seal this audit track.
Generation mode	int	Increment number for the seal algorithm name. (ie. MD5 = 1, MD5V2 = 2, SEALXYX = 3 etc.)
Seal generation result code	float	Result of running the 'Seal algorithm name' on 'Audit track'. (0 = OK, +n = failure result code)
Local generation	bit	Was this seal value generated by the local Archive Server or imported from the remote Archive Server (Yes = Local, No = Remote)
Date generated	smalldatetime	Date on which the calculation of the seal value was completed.
Time generated	datetime	Time at which the calculation of the seal value was completed.
From date	smalldatetime	Date on which this unique audit track was first passed to the Archive Server 'sealer' component .
From time	datetime	Time at which this unique audit track was first passed to the Archive Server 'sealer' component .
Until date	smalldatetime	Calculated date after which ALL records (in all tables) relating to this audit track can be deleted.
Until time	datetime	Calculated time after which ALL records (in all tables) relating to this audit track can be deleted.
Retention period	smallint	Number of days after the 'From' date when ALL records (in all tables) relating to this audit track can be deleted.
FileSize	int	Size of the audit track in bytes
StorageType	varchar(4)	Type of storage to which the file has been archived. T – Tape, C - Centera
CenteraID	varchar(100)	EMC Centera Clip ID for this file
DeleteFail	varchar(10)	

7.3.13 DeliveredFiles

Detail	
Status	Existing

FUJ00176545

Description	This table is not currently used as the Extractor core component does not record the delivery of files.

Table Structure		
Column	Type	Description
FileName	varchar(50)	Audit track name
RFI Reference	varchar(50)	ARQ reference.
Delivery Date	datetime	Date of file delivery

7.3.14 FileFoundArgument

Detail	
Status	Existing
Description	Contains all actions performed against an ARQ.

Table Structure		
Column	Type	Description
RFI Reference	varchar(50)	Request reference
Instance	int	Identity of the instance of the application the user is running
CmdUsedId	int	Index for the action being performed by the user or system upon an ARQ. Foreign key linking to table CmdUsedIndex.
Object	varchar(255)	Arguments required by code. May include filenames or comma-separated lists
ReturnStr	varchar(255)	Not used
returnInt	int	Not used

7.3.15 FileFoundConsole

Detail	
Status	Existing
Description	Contains all actions performed against a request which result in the initiation of a server side process. Achieved via an insert trigger which shells to the ExtractorLink.exe application.

Table Structure		
Column	Type	Description
RFI Reference	varchar(50)	Request reference
Instance	int	Identity of the instance of the application the user is running
CmdUsedID	int	Index for the action being performed by the user or

FUJ00176545

		system upon an ARQ. Foreign key linking to table CmdUsedIndex.
ErrCode	int	Holds the error state for an ARQ. '0' indicates no error.
StateID	int	Index for the state of an action being performed by the user or system upon an ARQ. Foreign key linking to table StateIndex.
StartAt	datetime	Used to record the starting time of an action being performed by the user or system upon an ARQ. For user actions the field is completed by the client. Format used is: 'dd/mm/ccyy hh:mm:ss'.
EndAt	datetime	Used to record the finishing time of an action being upon an ARQ. The field is completed by exe's running on the server. Format used is: 'dd/mm/ccyy hh:mm:ss'.
ErrMessage	varchar(255)	A text field completed by exe's running on the server indicating and narrating the unsuccessful completion of a user or system action on an ARQ.
GeneratedBy	varchar(255)	Holds the user name of the user that instigated the action causing this record to be generated

7.3.16 RecoverBatches

Detail	
Status	Existing
Description	

FUJ00176545

Table Structure		
Column	Type	Description
RecoverFileName	varchar(255)	
RFI Reference	varchar(50)	

7.3.17 RepeatableSelectCriteria

Detail	
Status	Existing
Description	Contains criteria used in the extraction of files from the EMC Centera for the specified ARQ. Multiple rows may be applied to a single ARQ, and in combination with the [Select Start Date] and [Select End Date] columns of the Requests table constitute the complete extraction criteria.

Table Structure		
Column	Type	Description
CriteriaID	int	Unique identifier
Audit Point	varchar(50)	Audit Point to be used in the extraction criteria.
Audit Sub Point	varchar(50)	Audit Sub Point to be used in the extraction criteria.
FAD Code	int	FAD Code used to generate an Audit Point/Audit Sub Point combination
RFI Reference	varchar(50)	Reference to the ARQ to which the criteria is to be applied.

7.3.18 Requesters

Detail	
Status	Existing
Description	Details of the divisions from which requests originate

Table Structure		
Column	Type	Description
RFI Requester ID	varchar(255)	Name of the Requester
Phone	varchar(50)	Requester contact phone number
Prefix	varchar(50)	Prefix used in the generation of ARQ references
NextRFIRefNo	int	Sequentially incremented number used in the generation of the ARQ reference
Organisation	varchar(50)	Organisation to which the requester belongs

FUJ00176545

7.3.19 Requests

Detail	
Status	Existing
Description	Contains a row for each active ARQ.

Table Structure

Column	Type	Description
RFI Reference	varchar(50)	ARQ reference. Generated from data held in the Requesters table relating to the value of the [RFI Requester ID] field for the row.
AW Operator ID	varchar(50)	Operator name from AWOperators
RFI Requester ID	varchar(255)	Identifies the request originator. Selected by the operator from the values available in the Requesters table.
Access Reason	varchar(255)	Reason for the ARQ as input by the operator.
Catalogue Entry	varchar(50)	
Date RFi Received	datetime	Date upon which the ARQ was received. Input by operator.
Required By Date	datetime	Date by which the ARQ response is required. Input by the operator.
Receipt Reference	varchar(50)	Reference details. Input by the operator
Status	varchar(50)	Current status of the ARQ. Available values held in the RequestStatusValues table.
Select Start Date	datetime	Date from which files should be extracted. This value is used to select matching files from the SEALERSQL [Initial Track table].
Select End date	datetime	Date before which files should be extracted. This value is used to select matching files from the SEALERSQL [Initial Track table].
Data Centre	varchar(50)	Not currently used
Action Count	int	A counter to hold the next instance value for a particular ARQ. The value increments whenever an action, as per table CmdUsedId, is instigated.
File Name Template	varchar(255)	A string which may optionally be supplied by the operator
Assigned Marker	varchar(255)	Used to indicate an ARQ's availability status. Can be 1 of 3 possible values: • (Blank, empty or NULL). This has been generated prior to this release and, if it is not permanently 'Closed', it is available for use by any 'Audit' operator. • 'FREE' This ARQ is NOT permanently 'Closed' and it is available for use by any 'Audit' operator.

FUJ00176545

		• <User Name>-<WorkstationID>-<ProcessID> This ARQ is in use by this user on this workstation, and is not available for selection. Note: If this ARQ is permanently 'Closed' it is used to show which 'Audit' operator closed the ARQ. (See ***AWOperators*** table)
Include Events	int	Whether or not it has been requested by the Audit Client that Event data corresponding to transaction data should be retunred.
FastARQ	int	Whether or not in the client it has been requested that a Fast ARQ be done.

7.3.20 RequestStatusValues

Detail	
Status	Existing
Description	Status values for ARQs

Table Structure

Column	Type	Description
Request Status	varchar(50)	Status description
Comment	varchar(100)	Explanatory note

Values

Request Status	Comment
Closed	This RFi is no longer available to the user
Copying:Active	File copies to CDROM/DLT in progress
Copying:Complete	Files copied to CDROM/DLT
Filtering:Active	A filter is being applied to the extracted data
Filtering:Complete	Filtering of all active files has completed
Filtering:Failed	Filtering of one or more active files has failed
New:CriteriaOnly	User has supplied criteria. No file list generated
New:FileList	File list generated & saved. No retrieval request
New:FileList&Vol	File list generated with vol info. No retrieval.
New:RequestOnly	User has only submitted RequestInfo details
Query:Active	ARQ Query is active
Query:Complete	ARQ Query has completed
Query:Failed	ARQ Query has failed
Retrieving:Active	There is a retrieval request in progress
Retrieving:Passive	Some files have been retrieved
Seal:Complete	All active files have passed the seal check

FUJ00176545

Seal:Failed	One or more active files have failed the seal check

7.3.21 RFIAuditSubPointExtended

Detail	
Status	Introduce in HNG-X
Description	Contains data relating to format, filtering and transformation requirements of audit tracks originating from the Audit sub point which is not available externally and would be lost during a refresh of the data held in the AuditSubPoint table.

Table Structure

Column	Type	Description
AuditPoint	varchar(50)	Audit point name
AuditSubPoint	varchar(50)	Audit Sub Point Name
DataSourceId	varchar(10)	Name of data source. Either “Horizon” or “HNG-X”
AllowStringSearch	bit	Indicates whether a string search may be performed on audit tracks originating at this sub point. 0 = No 1 = Yes
AllowXMLQuery	bit	Indicates whether an XML query may be performed on audit tracks originating at this sub point. 0 = No 1 = Yes
RAGConversion	bit	0 - No conversion required. 1 - RAG to XML conversion required.

7.3.22 RFIDataSource

Detail	
Status	Introduce in HNG-X
Description	Details relating to the source of the data. This table should only contain two rows; One for Horizon and one for HNG-X.

Table Structure

Column	Type	Description
DataSourceId	int	Unique id for the data source
DataSource	archar(50)	Data source name.
RAGConversion	bit	Indicates whether audit tracks originating at this data

FUJ00176545

		source require conversion from Riposte Attribute Grammar (RAG) to XML prior to performing and XML query. 0 - No conversion required. 1 - RAG to XML conversion required.

7.3.23 RFIPublicKeys

Detail	
Status	Introduce in HNG-X
Description	Query to be executed against extracted data

Table Structure		
Column	Type	Description
PublicKeyId	int	Unique id for the key
QueryId	int	QueryId of the query which resulted in this key request
QueryFileId	int	FileID of the file to which this key applies
FADCode	Varchar(7)	FAD Code
CounterNo	Int	Counter identifier
SessionId	Varchar(40)	Counter session Id
PublicKey	Varchar(50)	Public key

7.3.24 RFIQuery

Detail	
Status	Introduce in HNG-X
Description	Query to be executed against extracted data

Table Structure		
Column	Type	Description
QueryId	int	Unique id for the query
QueryRequestId	int	ID of the Query Request to which the query is linked.
QueryStatusId	int	Status ID of the Query. Foreign key from RFIQueryStatus table.
DataSourceId	int	Data source to which the query applies.

7.3.25 RFIQueryColumn

Detail	
Status	Introduce in HNG-X
Description	Attributes to be included in the query as selected from the message and/or existing within the message. Multiple rows may exist for a single query.

FUJ00176545

Table Structure		
Column	Type	Description
QueryColumnId	int	Unique query column ID
QueryId	int	ID of the query from RFIQuery
OrderBy	int	Use column in “order by”. 0 – Don’t use 1 – Order Ascending 2 – Order Descending
OrderBySequence	int	Sequence in which sort order columns should be applied
ColumnName	varchar(100)	Name of the column in either RAG or XML format
Selected	int	Use column in selected columns section of the query
Existing	int	Use column in “exists” section of the query.
OrderType	Varchar(20)	Type of ordering to be applied (Text, numeric, date)

7.3.26 RFIQueryError

Detail	
Status	Introduce in HNG-X
Description	Holds details of errors encountered during the processing of a query.

Table Structure		
Column	Type	Description
QueryErrorId	int	Unique ID of the query error
QueryFileId	int	ID of the file from RFIQueryFile in which the error was encountered.
QueryId	int	Identity of the query to which the error relates
MessageNumber	int	Error message number
ErrorDescription	varchar(255)	Error description
ErrorData	text	Supporting data generated during error

7.3.27 RFIQueryFile

Detail	
Status	Introduce in HNG-X
Description	Holds the list of files from the extract which are to be included in the filtering. Populated at the start of the query run and based upon the contents of the ActiveFiles table for the selected RFIReference.

Table Structure		
Column	Type	Description

FUJ00176545

QueryFileId	int	Unique identity of the query file
FileName	varchar(255)	Name of the audit track file
QueryRequestId	int	ARQ reference
QueryId	int	Identity of the query from RFIQuery that will be used to process the file.
DataSourceId	int	Identity of the data source from which the file originated
QueryFileStatusId	int	Status of the query file from RFIQueryFileStatus
FilterMatchFound	int	Indicates whether matching message have been found. 0 = No matches 1 = Matches found
GapsFound	int	Number of sequence gaps found
ErrorsFound	int	Number of errors found

7.3.28 RFIQueryFileSequence

Detail	
Status	Introduce in HNG-X
Description	Sequence details generated during the parsing of query files. A row will be generated for each FADCode/CounterNo consequtive sequence per file.

Table Structure		
Column	Type	Description
SequenceId	int	Sequence identifier
QueryRequestID	int	Identifier of the query request to which the row relates
QueryId	int	Identifier of the query to which the row relates
QueryFileId	int	ID of the Query file from RFIQuery file from which the sequence was generated
FADCode	int	FADCode to which the sequence applies
CounterNo	int	Counter number within the FAD to which the sequence applies
SeqType	varchar(1)	Sequence Type 'N' = Numeric, 'T' = Time
SeqStartNo	int	First number in the sequence
SeqEndNo	int	Last number in the sequence
SeqStartDate	datetime	Earliest date/time in the sequence
SeqEndDate	datetime	Latest date/time in the sequence

7.3.29 RFIQueryFileSequenceGap

FUJ00176545

Detail	
Status	Introduce in HNG-X
Description	Details of sequence gaps encountered when checking query files

Table Structure		
Column	Type	Description
SequenceGapId	int	Unique sequence gap ID
QueryFileId	int	ID of the query file in which the gap occurred. This may be '0' if the gap occurred between files, indicating that a file is missing from the sequence.
FADCode	varchar(8)	FAD code to which the gap relates
CounterNo	int	Counter number within the FAD to which the gap relates
GapStartNo	int	First sequence number which is missing
GapEndNo	int	Last sequence number which is missing

7.3.30 RFIQueryFileStatus

Detail	
Status	Introduce in HNG-X
Description	Status' that may be applied to an RFIQueryFile

Table Structure		
Column	Type	Description
QueryFileStatusId	int	Unique Query File Status
QueryFileStatus	varchar(50)	Description of the status
ProcessHalt	int	Should a change to this status prevent further processing of the query file? 1=Yes 0=No

Values		
QueryStatusId	QueryStatus	ProcessHalt
0	Created	1
1	New	0
2	Abstracting Files	0
3	Abstract Failed	1
4	Abstract completed Ok	0
5	Signature checking	0
6	Signature check failed	1
7	Signature check completed ok	0

FUJ00176545

7.3.31 RFIQueryRequest

Detail	
Status	Introduce in HNG-X
Description	Holds details of a query request for an ARQ. The presence of a row in this table for an ARQ indicates that filtering and/or querying are required.

FUJ00176545

Table Structure		
Column	Type	Description
QueryRequestId	int	Unique ID of the Query Request
RFIReference	varchar(255)	ARQ Reference
QueryRequestStatusId	int	Current status IDof the query request from RFIQueryRequestStatus.
Submitted	datetime	Date and time at which the request was submitted
QueryType	varchar(3)	The type of query to be performed. STR = string search QRY = XML Query
OutputType	varchar(4)	The level of abstraction to be applied. FILE = Complete audit track file containing match. ABST = Only matching lines or messages
FilterStartDate	datetime	Start date for message abstraction
FilterEndDate	datetime	End date for message abstraction

7.3.32 RFIQueryRequestFilter

Detail	
Status	Introduce in HNG-X
Description	Holds details of a query request for an ARQ. The presence of a row in this table for an ARQ indicates that filtering and/or querying are required.

Table Structure		
Column	Type	Description
QueryRequestFilterId	int	Unique ID of the Query Request Filter
QueryRequestId	int	Identifier of the query request to which the filter relates
SearchString	varchar(100)	FAD code to be included in the abstract or Search string
Locked	int	Flag to prevent the amendment of the query request filter FAD. 0 - Not locked, may be amended 1 – Locked, previously specified at extract.

7.3.33 RFIQueryRequestStatus

Detail	
Status	Introduce in HNG-X
Description	Status values for Query Requests

FUJ00176545

Table Structure		
Column	Type	Description
QueryReqStatusId	int	Unique ID of the query request status
QueryReqStatus	varchar(50)	Description of the status
ProcessHalt	int	Should a change to this status prevent further processing of the query request? 1=Yes 0=No

Values		
QueryStatusId	QueryStatus	ProcessHalt
0	Created	1
1	New	0
2	Query Invalid	1
3	Creating Directory Structure	0
4	Directory structure creation failed	1
5	Directory structure created	0
6	Abstracting files	0
7	Abstraction failed	1
8	Abstraction completed ok	0
9	Concatenating files	0
10	Concatenation failed	1
11	Concatenation completed	0
12	Query execution started	0
13	Query execution failed	1
14	Query execution completed	0
15	Closed	0

7.3.34 RFIQuerySelect

Detail	
Status	Introduce in HNG-X
Description	Selection criteria for a query

FUJ00176545

Table Structure		
Column	Type	Description
QueryselectId	int	Unique query selection criteria ID
QueryId	int	ID of the query from RFIQuery to which the criteria applies
SelectCriteria	Text(16)	Selection criteria

7.3.35 RFIQueryStatus

Detail	
Status	Introduce in HNG-X
Description	Status of a query

Table Structure		
Column	Type	Description
QueryStatusId	int	Unique ID of the Query Status
QueryStatus	varchar(50)	Query status description
ProcessHalt	int	Should a change to this status prevent further processing of the query? 1=Yes 0=No

Values		
QueryStatusId	QueryStatus	ProcessHalt
0	Created	1
1	New	0
2	File copy started	0
3	File copy failed	1
4	File copy completed	0
5	Sequence checking files	0
6	Sequence check completed with gaps	1
7	Sequence check completed without gaps	0
8	Concatenating files	0
9	Concatenation completed with errors	1
10	Concatenation completed without errors	0
11	Query execution started	0
12	Query execution failed	1
13	Query execution completed	0

FUJ00176545

14	Sort started	0
15	Sort failed	1
16	Sort completed	0

7.3.36 RFISQueryFName

Detail	
Status	Introduce in HNG-X
Description	Details of groupings of known attributes for use in query generation.

Table Structure		
Column	Type	Description
id	int	Unique attribute identifier
FType	Varchar(3)	Attribute Usage
FName	Nvarchar(1000)	
FNameHx	Nvarchar(1000)	Attribute name
Comments	ntext	Descriptive comment
STANDARD	bit	Indicates inclusion in the STANDARD grouping 0 – Not Included 1 - Included
BANKING	bit	Indicates inclusion in the BANKING grouping 0 – Not Included 1 - Included
EPOSS	bit	Indicates inclusion in the EPOSS grouping 0 – Not Included 1 - Included
OBCS	bit	Indicates inclusion in the OBCS grouping 0 – Not Included 1 - Included
APS	bit	Indicates inclusion in the APS grouping 0 – Not Included 1 - Included
LFS	bit	Indicates inclusion in the LFS grouping 0 – Not Included 1 - Included
MEMOS	bit	Indicates inclusion in the MEMOS grouping 0 – Not Included 1 - Included
OTHER	bit	Indicates inclusion in the OTHER grouping 0 – Not Included

FUJ00176545

		1 - Included
AB	bit	Indicates inclusion in the AB grouping 0 – Not Included 1 - Included
C	bit	Indicates inclusion in the C grouping 0 – Not Included 1 - Included
D	bit	Indicates inclusion in the D grouping 0 – Not Included 1 - Included
Runtime	bit	Indicates inclusion in the RUNTIME grouping 0 – Not Included 1 - Included

7.3.37 StateIndex

Detail

Status	Existing
Description	Contains all possible ARQ statuses

Table Structure

Column	Type	Description
StateID	int	State Index
State	varchar(50)	State description

Values

StateId	State
1	Created
2	Acknowledged
3	Processing
4	ProcessOK
5	ProcessNOK

FUJ00176545

7.3.38 TapePools

Detail	
Status	Existing
Description	Names of the tape pools implemented for varying retention periods prior to the move to disk based archival. While this table no longer has any real purpose within the system, a number of components still reference the data held.

Table Structure		
Column	Type	Description
TapePool	varchar(50)	Tape pool name
Comment	varchar(50)	Supporting comments

7.3.39 Volumes

Detail	
Status	Existing
Description	Tape volume details. It is highly likely that this table is no longer used.

Table Structure		
Column	Type	Description
Volume	varchar(50)	Volume name
OnLine	bit	Tape on-line indicator
Barcode	varchar(100)	Volume barcode

7.3.40 RFIEventsASPLink

Detail	
Status	Introduced for the Audit Strengthening CP (CP0336)
Description	This table links all possible combination of Transaction Audit Point/Sub Points to Event Audit Point/Sub Points. It is used so that in the client when the “Include Events” box is checked it is known where the relevant Events data is held.

Table Structure		
Column	Type	Description
EALID	int	Index
TransactionAP	varchar(50)	The Audit Point where the transaction data is.
TransactionASP	varchar(50)	The Audit Sub-Point where the transaction data is.
EventsAP	varchar(50)	The Audit Point where event data corresponding to the

FUJ00176545

		transaction data may be.
EventASP	varchar(50)	The Audit Sub Point where event data corresponding to the transaction data may be.

7.1.41 stp_FileStatusAction Stored Procedure

Called from the Extractor client, this stored procedure updates rows on the ActiveFiles table for the specified ARQ based upon the location of Audit Tracks within the UserArea ARQ directory and the presence of rows in the SEALERSQL database [Check Seal table table].

7.1.42 spArchivePANActivity Stored Procedure

Called at the end of the UpdateAPsADDRec schedules task this store procedure creates a files containing the contents of the RFIPANDecryptLog table.

The file is created in the D:\Archiveserver\AS_AUDIT_TRACK directory with a file name constructed as follows:

FN01_AUDIT_*ASP*_*SERVER_CHAR*_PAN_yyyymmdd_yymmdd_hhnnss_V001.txt

Where *SERVER_CHAR* = “W” or “B”

And

ASP = HxLog1 if *SERVER_CHAR* =”B”

= HxLog2 if *SERVER_CHAR* =”W”

The output file will be gathered by Audit along with other Audit specific files.

FUJ00176545

8 Application Development

The Audit Retrieval and Filtering system is developed using the following development tools and technologies:

- Microsoft Visual Studio 6.0
  - Visual Basic 6.0
  - Visual C++ 6.0
- Microsoft SQL Server 2000
- Windows Scripting Host
- Microsoft Internet Explorer
- Microsoft Office

FUJ00176545

9 Migration

9.1 Issues

The Audit Servers will be replaced with machines running Windows Server 2003 Standard, and the existing Audit workstations will be upgraded to Windows XP in line with the HNG-X platforms architecture requirements. Both of these operating systems include version 2.8 of the Microsoft Data Access Components (MDAC).

Remote Data Objects (RDO), the data access method adopted in the Visual Basic components, is now obsolete, and has not been supported in MDAC since 2001. To what extent it is implemented in version 2.8 of MDAC is unknown.

Initial testing of existing Audit components revealed a number of potential Issues that could be attributed to RDO. Further investigation is required to determine what, if any, action is required.

FUJ00176545

A RFISQL Database Schema