FUJ00091411

Document Title: HNG-X - Branch Database

Document Type: Architecture (ARC)

Release: HNG-X

Abstract: This document describes the architecture for the Branch Database. Counter through the Branch Access Layer uses Branch Database for capturing customer sessions and supporting back office activities inside the Branch. Other host applications get their transactional information from the Branch Database.

Document Status: APPROVED

This document contains sections that have been identified to POL as comprising evidence to support the assessment of named Acceptance Criteria by Document Review. These sections must not be changed without authority from the FS Acceptance Manager.

Author & Dept: Andy Beardmore

Internal Distribution:

External Distribution:

Approval Authorities:

Name	Role	Signature	Date
Graham Allen	HNG-X Development		
Jim Sweeting	Architecture		

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

(*) = Reviewers that returned comments

Documents are uncontrolled if printed or distributed electronically. Please refer to the Document Library or to Document Management for the current status of a document.

FUJ00091411

0 Document Control

0.1 Table of contents

FUJ00091411

FUJ00091411

FUJ00091411

FUJ00091411

0.2 Figures and Tables

FUJ00091411

0.3 Document History

Version No.	Date	Summary of Changes and Reason for Issue	Associated Change - CP/PEAK/PPRR Reference
0.1	26-Oct-06	Draft version	
0.2	06-Nov-06	Draft for review	
0.3	01-Dec-06	Draft for review	
1.0	16-Feb-07	Version for approval	
1.1		Draft version	
2.0	20-Jun-2007	Version for approval	
2.2	30-May-2008	Draft for review. Comments arising from Group Review Approved HNG-X Change Proposals (CP) applicable to this topic architecture Changes were made to the document for Acceptance by Document Review with the insertion of the Section containing the table of cross references for Acceptance by Document Review.	PWY_CP_4305
3.0	30-Oct-09	Review comments and for approval.	

0.4 Review Details

This document is subject to Group Review. Mandatory Reviewers should consult the Author to determine the details of the associated Group Review before submitting a formal comment sheet.

Review Comments by :	
Review Comments to :	
Mandatory Review	
Role	**Name**
HNGX-Solution Design/Infrastructure Design	David Harrison*
Architecture	Jim Sweeting*
Development	Steve Goddard*
SSC	Mike Peach*
Business Continuity	Adam Parker
RV Mig Test Manager	Graham Jennings
System Test	John Rogers
Optional Review	
Role	**Name**
HNG-X Programme Manager	Alan D' Alvarez
Applications Architecture	David Johns

FUJ00091411

System Qualities Architecture	Dave Chapman
Architect	Jason Clark*
Security Architect	Jim Sweeting*
Test Design	George Zolkiewka
Service Network	Ian Mills
Service Support	Kirsty Gallacher
SV&I Manager	Sheila Bamber
RV Manager	James Brett (POL)
POL Design Authority	Ian Trundell (POL)
VI & TE Manager	Mark Ascott
Head of Service Management	Gaetan van Achte
Head of Service Change & Transition	Graham Welsh
Data Centre Migration	Geoff Butts
Data Centre Migration	Peter Okely
Testing	Stephen Gilbert
Tester	Hamish Munro
HNGX Acceptance & Risk	Wayne Roberts(POL)
Integrity Testing	Alan Child
Integrity Testing	Michael Welch
Core Services	Ed Ashford
Core Services	Andrew Gibson
Business Architect	Gareth Jenkins
Development	Graham Allen
Security & Risk Team	CSPOA.Securit GRO
Issued for Information – Please restrict this distribution list to a minimum	
Position/Role	**Name**

0.5 Acceptance by Document Review

The sections in this document that have been identified to POL as comprising evidence to support Acceptance by Document review (DR) are listed below for the relevant Requirements:

POL NFR DR Acceptance Ref	Internal FS POL NFR Reference	Document Section Number	Document Section Heading
ARC-428	ARC-480	2.4.2	Implementation of Cut-off Reports uses JSN.Dates
SEC-3231	SEC-3304	2.4.3	Recovery
SEC-3233	SEC-3207	2.4.3	Recovery

0.6 Associated Documents (Internal & External)

Reference	Version	Date	Title	Source
ARC/GEN/TEM/0001 (DO NOT REMOVE)			Fujitsu Services Post Office Account HNG-X Architecture Document Template	Dimensions
AD/DES/041			TPS Agents for BI3 High Level Design	PVCS
AD/DES/047			TPS Tables and Mappings for CSR+	PVCS

FUJ00091411

ARC/APP/ARC/0001			HNG-X Reference Data Architecture	Dimensions
ARC/APP/ARC/0004			HNG-X Branch Access Architecture	Dimensions
ARC/APP/ARC/0007			HNG-X Batch Application Architecture	Dimensions
ARC/APP/ARC/0008			HNG-X Branch Database Architecture	Dimensions
ARC/APP/RTM/0006			Requirements Traceability Matrix for Branch Database	Dimensions
ARC/GEN/REP/0001			HNG-X Glossary	Dimensions
ARC/MIG/STG/0001			HNG-X Migration Strategy	Dimensions
ARC/NET/ARC/0001			HNG-X Network Architecture	Dimensions
ARC/PER/ARC/0001			HNG-X System Qualities Manual	Dimensions
ARC/PPS/ARC/0001			HNG-X Platforms and Storage Architecture	Dimensions
ARC/SEC/ARC/0003			HNG-X Security Architecture	Dimensions
ARC/SOL/ARC/0001			HNG-X Solution Architecture Outline	Dimensions
ARC/SOL/ARC/0005			HNG-X Counter Training Offices Architecture	Dimensions
ARC/SVS/ARC/0002			HNG-X Customer Services Architecture	Dimensions
ARC/SYM/ARC/0003			HNG-X System and Estate Management Monitoring	Dimensions
ARC/SYM/ARC/0005			HNG-X Estate Management Component Architecture	Dimensions
ARC/SYM/DPR/0002			HNG-X Estate Management Component Delta Changes	Dimensions
AS/DPR/013			Design Proposal for Track and Trace Integration	PVCS
BP/DES/023			LFS to SAPADS and SAPADS to LFS Application Interface Specification	Dimensions
DE/HLD/023			HNG-X Migration High Level Design for Horizon	PVCS
DES/APP/HLD/0020			Branch Database High Level Design	Dimensions
DES/APP/HLD/0021			Branch Database Scheduling High Level design	Dimensions
DES/APP/HLD/0022			Branch Database Sizing Spreadsheet	Dimensions
DES/APP/HLD/0023			Branch Support Database High Level Design	Dimensions
DES/APP/HLD/0024			Branch Support Database Sizing Spreadsheet	Dimensions
DES/APP/HLD/0025			Branch Support Database Scheduling High Level design	Dimensions

FUJ00091411

DES/APP/HLD/0053			HNG-X BAL Reporting High Level Design	Dimensions
DES/APP/HLD/0083			HNG-X Counter Subsystem : Recovery Management	Dimensions
DES/APP/HLD/0109			HNG-X Counter Reporting - High Level Design	Dimensions
DES/APP/HLD/0113			HNG-X HLD - BRDB Processing of the In Day Migration Data	Dimensions
DES/APP/IFS/0007			Branch Database to Legacy Host Interface Specification	Dimensions
DES/APP/IFS/0007			Branch Database to Legacy Host Interface Specification	Dimensions
DES/GEN/REP/0001			Evaluation and Testing of FSC Primergy BladeFrame For HNG-X	Dimensions
DES/GEN/STD/0001			HNG-X Host Applications Database Design and Interface Standards (HADDIS)	Dimensions
DES/MIG/HLD/0001			HNG-X Migration High Level Design For Branches	Dimensions
DES/MIG/HLD/0002			HNG-X Migration High Level Design for Data Centres	Dimensions
DES/PPS/PPD/0015			Branch Database Physical Platform Design	Dimensions
DES/PPS/PPD/0072			Branch Database Server - Standby Physical Platform Design	Dimensions
DES/PPS/PPD/0073			Branch Support Server Platform Physical Design	Dimensions
EA/IFS/002			POL Finance Systems to TMS/Horizon Transaction Corrections Interface Specification	PVCS
EA/IFS/006			Horizon to POL MIS Application Interface Specification	Dimensions
EP/LLD/029			Horizon to HNG-X End of Day Migration Low Level Design	PVCS
EP/LLD/030			Horizon to HNG-X In Day Migration Low Level Design	PVCS
LF/DES/003			Logistics Feeder Service High Level Description	PVCS
NB/HLD/015			DRS High Level Design Delta for NBE Replacement	PVCS
PA/PER/003			Horizon Capacity Management and Business Volumes	PVCS

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

FUJ00091411

0.7 Abbreviations

.

Abbreviation	Definition
APS	Automated Payment Service
BAL	Branch Access Layer
BCDB	Branch Configuration Database
BCMS	Branch Change Management System
BP	Balance Period
BRDB	Branch Database
BRSS	Branch Support System
CSN	Cut-off Sequence Number
CTO	Counter Training Offices
DCS	Debit Card System; service that supports payment by Debit or Credit Card
DMZ	De-Militarised Zone; area behind a Firewall
DNS	Domain Name Server
DR	Disaster Recovery
DVLA	Driver and Vehicle Licensing Agency
EDG	Post Office's External Data Gateway
EMDB	Estate Management Database
EPOSS	Electronic Point Of Sale Service
ETS	Electronic [Phone card] Top-up Service
FAD	Finance Accounts Division, part of Post Office Ltd
FRTS	First Rate Travel Services: the Post Office partner in the provision of currency and traveller's cheque sell and buy services
GCS	Oracle Global Cache Service
GES	Global Enqueue Services
HLD	High Level Design
HR SAP	Human Resources Standard Accounting Package - the SAP System used by Royal mail Group's Human Resources to pay sub-postmasters
IP	Internet Protocol
JDBC	Java Database Connectivity
JSN	Journal Sequence Number
LAN	Local Area Network
LFS	Logistics Feeder Service
LGWR	Log Writer process

FUJ00091411

MIS	Management Information System; Horizon application set that provides access to and analysis of the information held in the Data Warehouse
NBS	Network Banking Service
NPS	Network Banking Persistence Service
PAF	Postal Address File
POL FS	Post Office Ltd Finance System. SAP based system providing financial accounting for the branch-based business. This is the production system. There are other SAP systems in the data centre to support development and test.
QoS	Quality of Service
RAC	Real Application Cluster
RMAN	Oracle Backup and Recovery software
SAPADS	SAP Automated Distribution System. POL's Advanced Distribution System (based on the SAP package) that interfaces to LFS
SRDF	Symmetrix Remote Data Facility
SLT	Service Level Target
SRS	System Requirements Specification
SSN	Session Sequence Number
SYSMAN	Systems Management
SU	Stock Unit
T&T	Track and Trace
TCP	Transmission Control Protocol
TPS	Transaction Processing System
TSN	Transaction Sequence Number
UDP	User Datagram Protocol
USN	Uniqueness Sequence Number,
WAN	Wide Area Network
XML	Extensible Mark-up Language

0.8 Glossary

See also HNG-X Glossary (ARC/GEN/REP/0001).

Term	Definition
Branch Database Components	Collective term for Branch Database, Standby Branch Database and Branch Support System.
Cluster	A cluster is a group of loosely coupled computers that work together closely so that in many respects they can be viewed as though they are a single computer. Clusters are usually deployed to improve performance and/or availability over that provided by a single computer.
Database	A collection of records stored in a systematic way. The software used to manage and query records is known as the Database Management System. This document

FUJ00091411

	uses the term 'Database' to cover both meanings.
Hydra	The migration state within the HNG-X programme for supporting dual running of HNG-X and Horizon infrastructure.
Instance	An instance is composed of memory structures and the Oracle background processes that run on a server.
Node	Any device connected to a network such as a server. In the document, the term 'Node' includes the Oracle Instance.
Other Host Applications	Other Host applications, both batch and on-line. Batch includes - TPS, APS, LFS, DRS, TES, RDMC and RDDS and on-line includes NPS and APOP.

0.9 Changes Expected

Changes
Release 2 CP's.

0.10 Accuracy

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.

0.11 Copyright

FUJ00091411

1 Introduction

1.1 Scope

An overview of HNG-X solution architecture is in *HNG-X Solution Architecture Outline* (ARC/SOL/ARC/0001).

In HNG-X, branch data is centralised into the Branch Database.

This document (ARC/APP/ARC/0008) covers the topic architecture of the HNG-X Branch Database, and is an architectural solution to meet the requirements from the following sources:

- The HNG-X business requirements, in the DOORS tool
- The HNG-X Customer Services requirements, in the DOORS tool
- The HNG-X System Requirements (SR) in the DOORS tool
- The HNG-X contract schedules
- Approved HNG-X Change Proposals (CP) applicable to this topic architecture

The document elaborates into the following High-Level Designs (HLD) and Interface Specifications (IFS). These documents carry detailed information about the Branch Database components:

- Branch Database High Level Design (DES/APP/HLD/0020)
- Branch Database Scheduling High Level design (DES/APP/HLD/0021)
- Branch Database Sizing Spreadsheet (DES/APP/HLD/0022)
- Branch Support Database High Level Design (DES/APP/HLD/0023)
- Branch Support Database Sizing Spreadsheet (DES/APP/HLD/0024)
- Branch Support Database Scheduling High Level design (DES/APP/HLD/0025)
- Branch Database to Legacy Host Interface Specification (DES/APP/IFS/0007)

1.2 Background

Post Office Ltd operates in both the retail and financial services industries. The Post Office's main channel to market is a network of approximately 12,000 branches. A set of IT systems know as 'Horizon' support these branches.

The objective of the HNG-X programme is to reduce support and maintenance costs by developing a replacement to Horizon.

1.3 Solution Outline

1.3.1 Data storage

HNG-X stores branch data in a new Branch Database in the Data Centre. This contains:

- Customer transaction data
- Reference data to be distributed to branches
- Data that applies only to individual branches, such as users, stock units and messages

FUJ00091411

- Branch report data
- Recovery data
- Journal data
- Migration data to facilitate branch migration from Horizon to HNG-X

The Branch Database replaces the Riposte Message Store used by Horizon.

1.3.2 Branch transactions

The Branch Database supports a variety of transactions in the branch:

- Sale of fixed price goods and services
- Payment of bills -- utilities, local government, etc
- Banking services – deposits, withdrawals and balance enquiries
- Mobile phone electronic top-ups
- Bureau de Change
- Payment by Debit Card
- Electronic data capture
- Parcels and letters

Branch transactions take place within Customer sessions. There are different types of session. The session type controls what transactions can take place in a session. A customer session links transactions initiated by a customer, such as buying stamps, topping up a mobile phone, and then paying for these items with cash or debit card. When the value of the goods and services purchased equals the value of the payment received the session is settled. The net value of the transactions in a session is zero.

Transactions performed at a Post Office branch are either on behalf of Post Office Limited or on behalf of a third party such as a utility company. Transactions performed on behalf of a third party are mainly financial, for example paying bills or withdrawing cash. Through its Data Centre, HNG-X links to third party systems for authorisation and reconciliation, and to Post Office Ltd for accounting and management systems.

The HNG-X Data Centre (Branch Database) receives transaction details when a Customer session is settled. Some transactions involve access to third party systems before the end of the session. These transactions may make changes in these systems, such as reserving funds for withdrawal. If there is a failure before the end of the Customer session, there will be a mismatch between the settlement data and the transactions. Section 2.4.3 describes the recovery of these transactions.

The Branch Database also supports branch back office functions, including:

- Counter and Office Reports
- Stock Unit Declarations
- Accounting Summaries
- Stock Unit and Office accounting and rollover functions

Branch Database also acts as repository for Service Level Target (SLT) data generated by other HNG-X applications. Based on this data are number of reports are produced.

FUJ00091411

Clients of Branch Database are responsible for complying with the PCI standards when storing data in the database.

Clients wishing to store sensitive data in the Branch Database should encrypt the data before it is stored. Branch Database does not offer any Oracle or other encryption facilities.

1.3.3 Database characteristics

The database uses Oracle version 10gR2. It uses an Oracle Real Application Cluster (RAC), which runs the database over multiple nodes (servers).

Partitioned tables store branch specific data. This provides high performance and scalability. Applications need to know in which partitions data is stored and which nodes manage these partitions. They use a convention based on FAD codes (aka Branch codes).

The design of the Branch Database supports non-stop trading during core hours.

- Oracle RAC is resilient. If one node fails, the remaining nodes carry on running and the database remains available for use. The database can meet its performance targets if one node fails.
- The standby database allows very fast recovery if there is a data corruption that takes the live database offline. The maintenance of the standby database is automatic.

A disaster recovery site remotely mirrors the data. The mirroring of data is synchronous. This guarantees that no data is lost if there is a catastrophic site failure.

1.3.4 Training

Counter Training Office (CTO) use the live system for training. Training data is stored in database tables owned by the training user. No separate training system is required, other than dummy versions of external services such as Banking. CTO is similar to a normal branch.

1.4 This document

Section 2 describes the Branch Database architecture in detail.

- Section 2.1 covers the volume and performance assumptions used to design the solution.
- Section 2.2 covers the database clustering and partitioning.
- Section 2.3 covers how the applications meet the requirement to understand partitioning.
- Section 2.4 covers special processing rules: calculating unique identifiers, date processing, recovery, and reversals.
- Section 2.5 covers the design of the database schema.
- Section 2.6 covers the processing within the Branch Access Layer (BAL) that provides the branches with access to the Branch Database.
- Section 2.7 covers the aggregation and reporting supported by the Branch Database.
- Section 2.8 covers the interfaces between the Branch Database and other data centre systems.
- Section 2.9 covers the training capabilities supported by the Branch Database.

The remaining sections describe other facets of the Branch Database architecture.

- Section 3 covers the Hardware, Software and Storage requirements for Branch Database.

FUJ00091411

- Section 4 covers the Network requirements.
- Section 5 covers System Management and Supportability.
- Section 6 covers Security compliance.
- Section 7 covers Recovery and Resilience including Data Centre failure.
- Section 8 covers the Capacity and Performance requirements of the Branch Databases.
- Section 9 covers Migration from the Horizon solution.
- Section 10 covers Testing and Validation of the solution.
- Section 11 captures any major Risks and Assumptions.
- Section 12 documents the incoming requirements and is incomplete. The next version of this document should have the section completed. .

FUJ00091411

2 Architectural description

2.1 Assumptions

HNG-X centralises data within the Data centre such that no transactional information is stored at the Counters or at the Branches. The transactional data is stored in the new Branch Database within the Data centre.

The important targets for Branch Database, based on ARC/PER/ARC/0001 and analysis of Horizon system are:

- Transaction capture from approximately 35,000 counter positions
- A transaction capture rate of 830 transactions per second during the peak 5 minutes
- A peak session (settlement basket) commit rate of just under 385 sessions per second during the peak 5 minutes
- A requirement to store recovery records for 300 transactions per second (for some transactions there may be multiple updates to recovery records)
- Daily, Weekly and Monthly reporting loads peaking at approximately 100 report requests per second
- User logon peak of 100 requests per second at start of trading
- Aggregation and Transfer of all captured transaction information for onward transfer to other host applications and external interfaces
- Auditing, archiving and housekeeping

In addition, the Branch Database needs to be available 24x7, other than in agreed maintenance outage windows. No Branch Database = No Branch trading.

During Hydra, Branch Database is able to handle migration of 250 branches per night.

Individual branches trade exclusively either as a Horizon branch or as a HNG-X branch on any given trading day, but never both.

2.2 Database Architecture

2.2.1 Single database model

The choices for implementing Branch Database are - ‘single database model’ or ‘multiple databases model’. Use of multiple databases also provides a scalable solution. During design investigations and prototyping, it became clear that the multiple databases option is complicated to implement and manage. Hence, the choice is single database model.

The figure below shows the single database model:

FUJ00091411

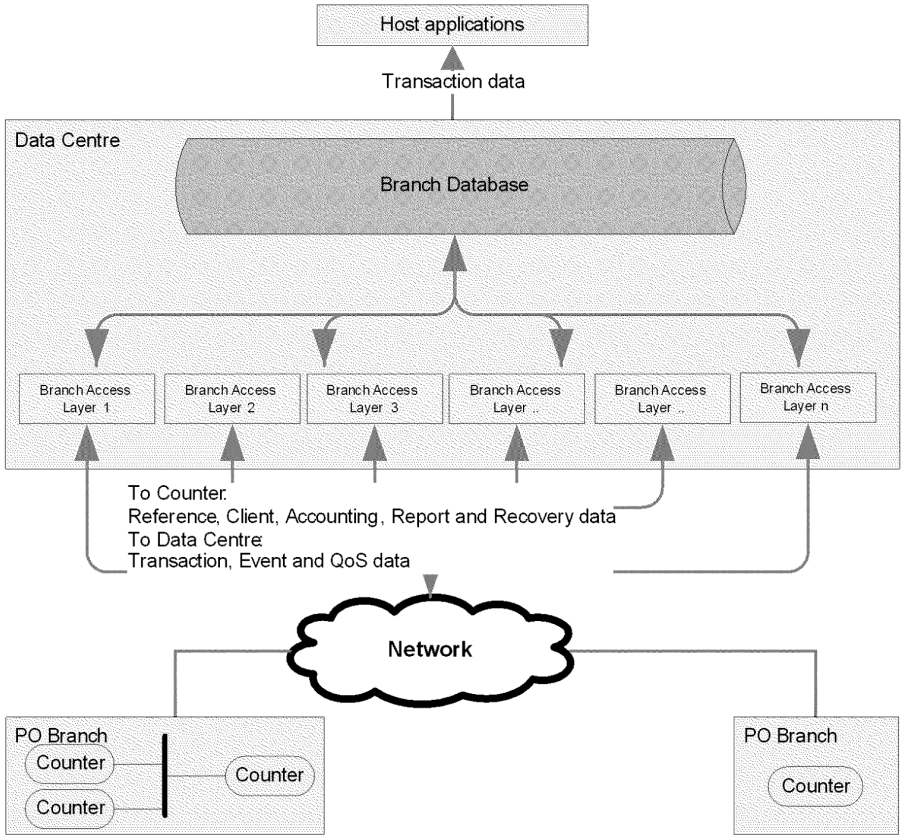


Figure 1 – Single Database Model

2.2.2 Clustering

There is a significant risk that the single database model may not perform adequately because it is difficult to tune it for both a high commit rate and a high volume of database queries.

By distributing the workload across multiple nodes, one can reduce this risk. The Oracle 10gR2 Real Application Cluster (RAC) can provide scalability and load balancing across multiple nodes. Hence, use of RAC reduces the performance risk.

This approach is similar to the Correspondence Server cluster that provides scalability and load balancing in Horizon.

2.2.2.1 Oracle Real Application Cluster

An Oracle RAC allows workload distribution across more than one physical node. The number of Nodes within the cluster is configurable. If the cluster underperforms, it is possible to add new nodes to increase its performance. It is also possible to remove nodes if the cluster is over-specified or the workload has decreased. Hence, processing power is very scalable.

FUJ00091411

The connection from the Counters to the Branch Database is through the Branch Access Layer (BAL). The number of BAL servers is more than the number of Oracle RAC nodes. Therefore, the HNG-X system looks similar to the diagram below:

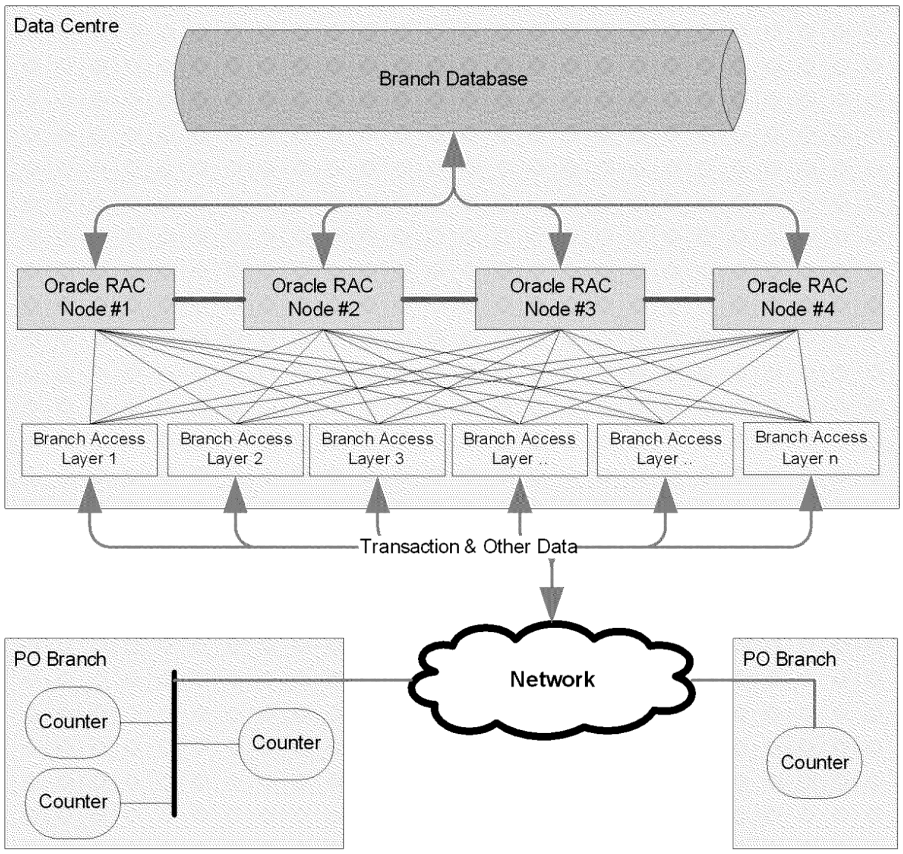


Figure 2 – Oracle RAC in a Tiered Architecture

The diagram shows that the individual Counters can send and receive transactional and other information via the Web Services that run on BAL servers within the Data Centre. The requests are routed (not shown on diagram) to the BAL servers via a load balancer (network appliance) to ensure even loading of the servers. *HNG-X Branch Access Layer* (ARC/APP/ARC/0004) describes the architecture for BAL, including the interactions between the Counter and the BAL.

Section 2.3.2 describes the BAL access of Branch Database.

2.2.2.2 Leveraging Oracle RAC

One of the issues with a single node database is that an Oracle Instance only contains a single Log Writer process (LGWR) and associated *redo allocation latch*. This becomes a bottleneck on larger multiprocessor systems since processes end up waiting for the LGWR process to complete its task.

An Oracle RAC consists of more than one node (instance) accessing the same database.

The 4-node cluster shown in the diagram provides good scalability. An Oracle instance runs on each node thereby increasing the number of LGWR processes. This is because each instance has a LGWR process and generates its own redo logs. The RAC configuration therefore provides greater scalability

FUJ00091411

than a big single instance machine. This is due to the reduction in the number of waits for 'redo log write' confirmations after a commit.

However, this in its own does not fully solve the scalability issues for systems that perform high rates of data insert and select such as the HNG-X application. There are two reasons for this:

- In applications where instances share the same data blocks, the excessive overhead in transferring common blocks between instances (called pinging) can negate any performance benefit of RAC
- With many database connections, on different nodes, all performing a high rate of inserts and selects, lock contention can cancel out some of the performance benefits of RAC.,

To circumvent both of the above issues, each node in the RAC must write to its own independent disk areas within the database. This involves partitioning the 'hot[1]' database tables such that individual partitions are only accessible by a single node in the cluster.

The 4-node cluster also provides high availability. The database has no single point of failure because of multiple nodes in a cluster. Multiple nodes also reduce the probability that a set of simultaneous failures can cause a complete loss of service.

2.2.3 Partitioning

Partitioning of 'hot' database tables allows the RAC to deliver best scalability and performance. For faster data access and easier maintenance, the best option for partitioning is to use keys based on Date/Time plus a value within the transaction that is meaningful from a business perspective. For Post Office FAD code is ideal because Post Office branches are not required to share data. The benefits of this approach are:

- True scalability of reporting since the request for data by any one Branch will access only a single partition.
- Unique indexes can also contain the FAD Code, since uniqueness is only required within a Branch. In this case, the database maintains index uniqueness in a truly scalable manner because the uniqueness check is constrained to a single (index) partition.

However, these benefits are at the cost of implementing routing intelligence within the BAL. The following section discusses the implementation of this routing intelligence. The proposed architecture for routing intelligence means that we cannot use Oracle Transparent Application Failure (TAF) feature. The BAL manages node failures explicitly.

Evaluation of load balancing and TAF features provided by Oracle showed that they did not offer same scalability and performance as bespoke implementation of routing intelligence within the Application server.

2.3 Application Architecture Awareness

In order to maintain architectural purity, the application software should not have any knowledge of the database architecture built into it. This allows the underlying database architecture to change without affecting the business applications and therefore facilitates changes to the architecture with low associated costs and risks. However, the model proposed above introduces some knowledge of the architecture into the database schema and BAL to achieve desired load balancing. Intelligent routing of messages also helps to achieve the desired load balancing. A well-balanced RAC helps in leveraging maximum performance and scalability from the cluster during peak loads.

[1] Frequently accessed

FUJ00091411

The BAL maintains a pool of persistent connections because the overhead of making a database connection is high. Under this condition, Oracle's automated load balancing does not work. Load balancing is therefore the responsibility of BAL. The BAL servers ***must*** load-balance the Oracle Nodes using a routing algorithm. The routing algorithm uses metadata in the Branch Database. In case of node failure automatic updates to the metadata in real time ensures that the routing algorithm functions correctly.

Load balancing intelligence is also required in the Harvesters copying data between the Branch Database and other host applications.

2.3.1 Solution

For load balancing, both intelligent routing of messages and copying of data is required. The FAD code of the branch is the obvious choice because branches are not required to share information with each other and therefore there will be extremely limited cross FAD queries.

FAD code based partitioning means that FAD 000001 always writes to partition #1. In addition, whenever FAD 000001 wants to perform any reports or back-office tasks, the data always comes from partition #1.

However, in order to retain the scalability of the rate of insert operations, only Oracle Instance #1 may write to Partition #1. Therefore, the transaction data belonging to FAD 000001 routes to Oracle instance #1. The burden and complexity of this routing is on the BAL.

Using actual FAD codes as a routing and partitioning mechanism is difficult to set up and to manage because there are many branches and branches can open or close frequently. This gives rise to many new FAD codes, and many defunct FAD codes. The alternative is to assign FAD codes to hash buckets. The algorithm used for assigning existing FAD codes to hash buckets uses Branch trading volumes to guarantee that the buckets are well balanced. This happens prior to Branch Database going live. Once the Branch database goes live, a hashing algorithm assigns new FAD codes to existing buckets. For new FAD codes (branches), trading volumes are unknown and use of a Hash algorithm is the best option. The Horizon counter already uses an algorithm for assigning hash values to FAD codes; the Branch database uses the same algorithm. For the Branch Database, hash values are in the range 0 and 127.

Branches have two FAD codes assigned to them. These are Branch_Accounting_Code and Branch_Code. This is necessary to support Cluster trading in future and the Training solution described in section 2.9. For business transactions, back office functions and routing Branch_Accounting_Code is used. For reference data and estate management functions, Branch_Code is used. Except for CTO branches, both codes have the same value. For CTO branches, Branch Database allocates a new Branch_Accounting_Code for each Counter when the CTO branch is set up in the Branch Database. For non-CTO branches the Branch Accounting Code and Branch Code is the same.

2.3.2 Branch Access Layer Access

2.3.2.1 Message Routing

The figure below shows the concept of intelligent message routing pictorially. The pink arrow shows the routing of all transactions and other database traffic for branches whose hash value is zero to RAC node 1.

FUJ00091411

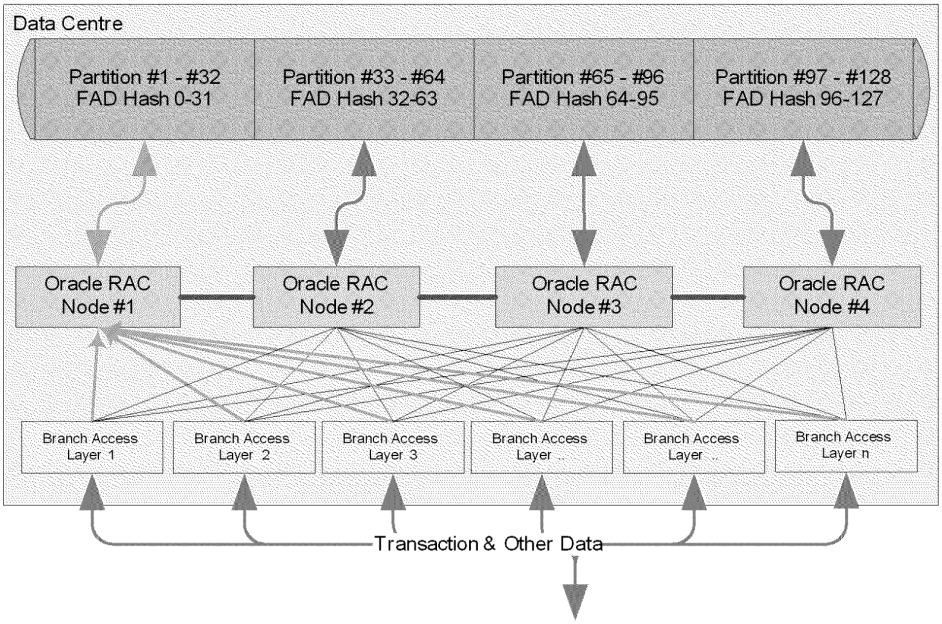


Figure 3 – Intelligent Message Routing

2.3.2.2 Load balancing

Regardless of node availability, the FAD hash values must remain well balanced across the nodes. Addition or deletion of nodes must maintain a balanced and performing system.

When a message arrives from the Counter, the BAL needs to look up the FAD hash value and use the value to determine the right node connection for the request. The cross-reference table below provides the necessary mappings for this look up. However if a node fails for any reason, the requests targeted at that node must be balanced across the remaining nodes.

FAD Hash Value	Oracle Node
0	BranchDatabase_Node_01
1	BranchDatabase_Node_01
2	BranchDatabase_Node_01
3	BranchDatabase_Node_01
..	
32	BranchDatabase_Node_02
..	
64	BranchDatabase_Node_03
..	..
96	BranchDatabase_Node_04

FUJ00091411

..	
127	BranchDatabase_Node_04

Table 1 – FAD Hash to Node Mapping

2.3.2.3 Failover

If one node fails, the other nodes still carry on running. The remaining three nodes can meet the performance targets given in section 2.1.

The database traffic for the failing node requires re-routing across the remaining nodes. By adding priority levels in the cross reference table above, the transactions for FAD Hash 0 are re-routed to Node #2, FAD Hash 1 are re-routed to Node #3, FAD Hash 2 are re-routed to Node #4 and FAD Hash 3 are re-routed to Node #2 and so on. In this way, re-routing of transactions originally bound for Node #1 happens in a controlled and predictable manner. This guarantees that FAD hash values across the remaining nodes are well balanced.

FAD Code Hash Value	Priority	Oracle Instance Name
0	1	BranchDatabase_Node_01
0	2	BranchDatabase_Node_02
0	3	BranchDatabase_Node_03
0	4	BranchDatabase_Node_04
1	1	BranchDatabase_Node_01
1	2	BranchDatabase_Node_03
..		
..		

Table 2 – FAD Hash to Node Priority Mapping

2.3.3 Connection Pooling

Opening a connection to the database has a performance overhead. Consequently, the recommendation is to re-use connections. BAL pre-allocates database connections on start-up and re-cycles them for message processing. For RAC, a connection pool is required for each node (instance) in the cluster. JDBC connection pooling feature improves performance and use of connection pooling is mandatory. The Oracle has made a number of performance extensions such as statement batching to their JDBC driver. If feasible, BAL should use these extensions.

2.3.4 Other Host Applications Access

As mentioned earlier, an Oracle RAC is highly scalable if the handling of a table partition is restricted to a single node in the cluster. This is even more important when copying large quantities of data between the Branch Database and other host applications. Hence, copying jobs need to be FAD hash aware. This

FUJ00091411

ensures that the data is copied in or out from the correct node. Failure to do so can result in a large amount of inter-nodal traffic thereby affecting the performance of the copying job and the cluster itself. The Branch Database maintains a FAD hash partition to node mapping; the copy jobs use the mapping to copy data from the correct node.

To support above the Branch Database has a number of harvesters to copy data between the Branch Database and other host applications. Section 2.8 describes the harvesters.

2.3.5 Summary

Partitioning of database tables by a combination of Date/Time and FAD hash is highly scalable and does not impose a large overhead of implementing routing in BAL or the Harvesters. This option provides the best reading and writing performance, smaller table partitions and the ability to have highly scalable unique indexes (as long as the index is based on FAD). In Oracle, terminology the database tables are 'range' partitioned on Date/Time sub-partitioned by 'list' on FAD hash.

2.4 Processing Rules

2.4.1 Unique Identifiers

Similar to Horizon system, there are various places in the HNG-X system where unique identifiers are required. The identifiers have many uses such as in auditing, in reports, printing on receipts and relating separate parts of a transaction. For auditing purposes, the unique identifiers should have no gaps in the sequence and it should therefore be possible to check whether data is missing. Other unique identifiers (e.g. Session and Transaction) are required in various places, without a requirement for them to be contiguous (e.g. Transaction ID used to match an [R1] to a [C0]) in Banking Transactions.

2.4.1.1 Solution

Each branch is assigned a unique identifier, and each logical Counter within the branch is assigned a unique position – if a Counter is physically replaced its logical position does not change. These two numbers are sufficient to identify the source of any message sent to the Data Centre.

The Branch Access Layer (BAL) journalises certain types of messages. Some examples are Customer session settlement message or an Event message recording a significant Counter event. Each journalised message for a given Counter has a unique identifier. For journalised messages, these identifiers form a dense set to show that the journal is complete. The Branch Database tables hold a sequence number, known as the Journal Sequence Number (JSN) for each Counter in the estate. The Counter is supplied a JSN at logon which it then maintains. The JSN is incremented each time the Counter sends a journalisable (auditable) message and receives confirmation that the message was successfully processed. Without such confirmation after several tries, a Counter has no choice but to eventually force a log-off. On Counter, log-on to the Data Centre tells the Counter the next JSN to use. In this way, the Counter and Data Centre are always in step.

If the User session terminates tidily, the Counter supplies the latest JSN, which the Branch Database stores. However, if the session terminates abnormally, the Branch database does not have the latest JSN used by the Counter. However, at next logon the Branch Database can determine the highest JSN used from its tables and can therefore provide the Counter with the next JSN for it to use[2].

On-line transactions need a Transaction Recovery Identifier. To support this, the Branch Database holds a sequence number, known as the Uniqueness Sequence Number, or USN, for each Counter in the

[2] Abnormal termination of a session is detected at the next logon and will result in a journal entry to that effect, thus incrementing the JSN and avoiding the risk of using the same JSN for two sessions.

FUJ00091411

estate. The Data Centre supplies the USN for the Counter to use at logon. The Counter tells the Data Centre what the [updated] USN is as part of the settlement.

The FAD code, the Counter identifier and the current USN together make up a unique *Transaction Recovery Identifier*. Every recovery written to the Branch Database must have Recovery Identifier. The counter must increment USN after sending message that requires a recovery record.

If a session terminates abnormally, the Data Centre will not have the latest USN used by the Counter. However, at the next logon, it can determine the highest USN used in a Transaction Recovery Identifier, and can therefore provide the Counter with the next USN for it to use. Note that the Counter may have allocated a higher USN than this during the abandoned session, but only USNs written to the Branch Database need to be unique when next session starts.

The Authorisation Agents require a correlation identifier to tie together the Request, Authorisation and Confirmation messages that make up an online transaction. The Transaction Recovery Identifier can tie together these messages. However, the TPS expects this identifier to be a string with a particular syntax, hence:

pp-bbbbbb-cc-nnnnnnnnnn-uu

where

pp	is a fixed prefix; 00 for HNG-X, 44 for Horizon
bbbbbb	is the Branch Identifier
cc	is the Counter Identifier
nnnnnnnnnn	is the USN
uu	is a suffix, which for HNG-X is fixed at 1

Suppressing of leading zeroes for ‘bbbbbb’, ‘cc’, ‘nnnnnnnnnn’ and ‘uu’, gives identifiers such as 00-49934-2-1. HTxnNum is the name given to this form of identifier, and is common between Horizon and HNG-X, but the derivation of the fields is different between the two systems. Note HNG-X HTxnNum is always prefixed with ‘00’ where as the Horizon identifier is prefixed with ‘44’.

Recovery records for NBS, ETS and DCS transactions include the HTxnNum. When a Counter wishes to retrieve status information about a recoverable transaction, it uses HTxnNum and other identifiers held in the recovery record as the transaction identifier.

Implementation of Cut-off Reports uses JSN.

2.4.2 Dates

Branch Database establishes an end of day trading position for all branches regardless of their connectivity and log on status. The trading day is cut-off approximately at BST 19:00 hours, after which transactions for the day ending just before the cut-off are collected and sent to the clients. To facilitate this BAL assigns a ‘Trading date’ to all settlement transactions.

The high and low level design process will identify requirements for other Date/Time attributes to be stored in the Branch Database

Time at the Counter is closely synchronised with the Data centre.

Dates are all UTC unless explicitly stated otherwise.

FUJ00091411

2.4.3 Recovery

HNG-X must cope with failure to deliver, or a perceived failure to deliver a session's settlement data to the Data centre and subsequent cancellation of the session. End-to-end Session recovery is not in scope for this document, so a summary is included to understand the Branch Database requirements.

Transactions written to the session basket at the Counter are either cancellable or recoverable. Some transactions are cancellable up to a point before becoming recoverable. A class of transaction that is recoverable is one that uses communication with a third party for authorisation, causing a change of state and implicit transfer of funds between accounts based on that transaction. Another recoverable transaction class is one that creates an item of value and/or acquires a resource. To aid recovery, for each recoverable transaction in the customer session a recovery record is written to the Branch Database.

On successful completion of the customer session, the update of the recovery record happens in the same database commit as the session settlement.

Where network or application problems prevent a Clerk from completing a session successfully, they can cancel the session. Depending on the point in the session at which the cancellation occurs, settlement data for the session may have already been committed. The Clerk follows a well-defined process for cancelling cancellable transactions and settling recoverable transactions, printing a Recovery Receipt and any Customer Receipts required (which may include void receipts for cancelled transactions). The application then forcibly logs the clerk off. For further details see *HNGX Counter Subsystem : Recovery Management* (DES/APP/HLD/0083)

The Branch Database tracks logon and logoff for each counter. During counter logon, the Branch Database can detect if a previous session had terminated abnormally (i.e. a logoff did not happen). This triggers a recovery process at the BAL and the Counter. The recovery process attempts to bring the Data Centre's view of the session state in line with the Counter's view. For example, if the settlement data had been committed, that data might contain transactions that the Counter regards as cancelled, and the system would then need to reverse these transactions.

The Counter can retrieve the recovery records. A recovery record may simply identify the transaction, requiring a further request to the Data Centre to retrieve the transaction status, or it may contain all the transaction details necessary for recovery, such as the transaction amount. No sensitive data is stored in a recovery record.

BAL where appropriate (e.g. banking) writes a recovery record to the Branch Database (Recovery table) before forwarding the message to the correct On-line service. The recovery data store consists of a partitioned table that stores the Branch, Counter and User details, Date/time and Status of the transaction in progress. On session completion, the BAL updates the recovery table when processing the settlement message. For performance, an index is required for this table. Recovery records are only necessary for certain On-line services (e.g. banking).

2.4.4 Banking, Debit Card and E-Top Up Reversals

The business rules for an Authorisation Agent may require guaranteed delivery of reversals. If an agent sends an explicit reversal to its client and fails to get an acknowledgement it will repeat the request up to a configured retry limit. To guarantee reversals sent from the Counter are not lost, two delivery routes are used. Counter requests a reversal explicitly through the BAL, and it includes the reversal in the session's settlement data. When the session is settled, the reversal is committed to the Branch Database. A near real-time process (known as Guaranteed Reversals process) transfers reversals from the Branch Database to NPS. On NPS, the Authorisation Agent processes them. The Agents discard any duplicate reversals, they do so by using the transaction status table in NPS.

For Debit cards there is no requirement to support the Guaranteed Reversals process.

FUJ00091411

Note for a reversal request, BAL does not need to write a new recovery record to the Branch database. The recovery record written for the original request remains in the Branch Database until the settlement is completed. However the counter may choose to update this record. For a cancelled session, the recovery record is sufficient to cause a reversal in the recovery process.

2.5 Database Design

2.5.1 Overview of Branch Database use

The Branch Database is the repository for all branch transaction and event capture. It also stores other Branch data such as Users, Stock units, Transaction corrections, LFS Pouch data, Messages and Reference data.

Business data is committed to the Branch Database via the BAL. Logic within the BAL distinguishes between different transactions types in the XML message from the Counter. The BAL parses the XML message and inserts the data into the relevant Branch Database tables without further processing within the database.

The Branch Database also has a mechanism to store Service Level Target (SLT) data supplied by the Counter and other applications in the HNG-X solution. Later on, the overnight schedule can produce SLT reports based on this data.

Migration data from branches is pumped in overnight from the TPS host application. The batch database server should be sized to be able to support the additional workload to support the data copy into branch database.

The database design should be able to support the workload listed in section 2.1.

2.5.2 Design Principles

In order to minimise loads on the Branch Database during peak periods the design follows the principles listed below:

The insertion of data is as simple as possible. This means:

- Report data and the Main Transaction Store have minimum indexes
- No maintenance of running totals, overnight aggregations do the totalling
- Extract of data for reporting and accounting will not require sorting or subtotalling. The counter does the sorting and subtotalling.

Partitioning of transaction, reporting, aggregation and reference data tables achieves load balancing between the RAC nodes. The partitioning is by:

- Date/Time to allow ease of retrieval by period
- Suitable splitting of FAD codes

For high scalability and performance, the inter node communication for Oracle Global Cache Service (GCS) and Global Enqueue Services (GES) must be minimised. This requires:

- Individual partitions are in their own tablespaces
- Tables must be denormalised wherever appropriate to minimise the number of table accesses required to service a request

To simplify recovery in case of database corruptions, individual tables and partitions have their own tablespaces. However, some consolidation is inevitable to avoid having a large number of tablespaces.

FUJ00091411

2.5.3 Schema design

Schema design requires division of database into logical sub-sections that satisfy the various demands placed upon the data. The figure bellows shows the main logical subdivisions of the Branch Database. Section 2.6 describes the use of the schema by the Branch Access Layer.

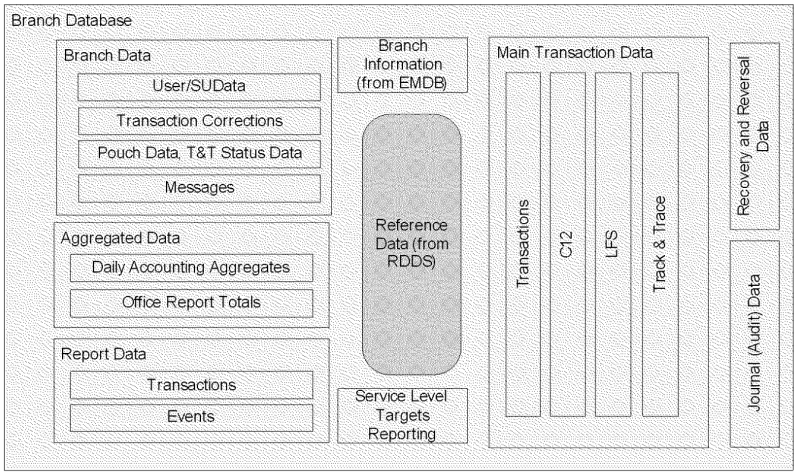


Figure 4 – Logical subdivisions of the Branch Database

The different logical subdivisions or data stores in the database are described below.

2.5.3.1 Journal Data

BAL is responsible for inserting a journal record containing the raw XML message it received from the Counter. Individual transactions or events in the message or messages for online services such as banking do not require journal entries. The context between the Web service and the Counter decides whether the message journaling is required. For messages that require journaling, the counter needs to supply the Journal Sequence Number (JSN) (see section 2.4.1).

The journal table stores the compressed raw XML message and has a unique index based on Date/Time, FAD hash, FAD code, Node identifier and Journal Sequence Number. An index violation signifies successful processing of the message earlier. In this case, the Web service must return an appropriate response back to the Counter.

For security reasons, delete and updates to the journal table are disabled, and only certain privileged accounts can insert data in to this table.

2.5.3.2 Main Transaction Data

BAL Settlement Service is responsible for inserting transaction data directly into the Main Transaction store. The Main Transaction store facilitates delivery of data to POA external interfaces via other host applications.

The main store requires access by other host applications that deliver the information to the POA external interfaces. Partitioning of the data facilitates bulk extracts of data by the host applications and no indexes should be required in order to increase the efficiency of this task.

Different transaction types within the main store are stored in separate tables like the TPS application.

FUJ00091411

2.5.3.3 Report Data

The Settlement Service also inserts a small sub-set of transaction attributes directly into a Report Data store. Report data is therefore available immediately after the end of a customer session. The design of Report Data store optimises Counter/Branch Daily/Weekly reporting.

The number of Report data store attributes needs to be minimised so that query performance is good. It is also important to include all the attributes that are required for Branch reporting, so that there is no need to retain data in the Main Transaction store for reporting. The Report Data store design is a trade-off between these.

It is necessary to classify reports in to groups such as 'Last Post Reports', 'End of Day Reports' to improve performance[3]. Report groups allow bulk retrieval of data for more than one report in a single request to the Reporting Service by the Counter. After retrieving the data, the Counter application renders individual reports on the Counter. This grouping mechanism also reduces the number of reporting request from the Counter and consequently the load on the Branch Database.

It is assumed that report sorting and subtotalling is performed at the Counter and therefore the database is not required to sort or subtotal branch data. (This distributes the processing power that is required for these processor-intensive tasks.)

The report tables are partitioned by Date/Time and FAD hash.

In addition, the Counter uses the Report data store for Existing Reversals. In these cases, only part of the original transaction is required; and Session/Transaction Id is required for its retrieval.

2.5.3.4 Aggregated Data

Daily batch processes are responsible for aggregation of data in the Main Transaction and Reporting data stores. This is stored in the Aggregated data store, which facilitates efficient calculation of Stock Unit and Branch totals for declarations and rollover processes. Declaration totals and Stock Unit Cut-Off totals are also stored in this area.

2.5.3.5 Branch Data

Branch database maintains inbound data from LFS, TPS and Messages for broadcast to clerks.

BAL Web Service is responsible for Counter user administration. The Branch data store stores Counter user details such as usernames, passwords (either encrypted or hashed) and roles. BAL is responsible for encrypting or hashing the passwords. User identifiers are unique within a logical branch, and a user with an appropriate role performs user administration within the branch (c.f. existing solution)[4].

User names are unique within a logical branch and their use is restricted to that branch (known as Local users). A user can only log into one machine at any one time. The solution also provides a small number of centrally administered users (known as Global Users) that are valid in all branches. Help desk staff have an interface to reset passwords for (global and local) user accounts.

Branch Data store also stores LFS Rem-Out sack contents, Track & Trace life history, Message Broadcasts and LFS Message information.

2.5.3.6 Reference Data

To reduce network traffic each HNG-X Counter has a local copy of Reference Data. This is similar to the Horizon Counter.

[3] Some reports source data from the same table

[4] There are some types of user that require access across all branches (such as Auditor and Engineer)

FUJ00091411

Reference data is also required internally within the Branch Database. In most implementations, the application uses central Reference Data to validate values within each customer session. However, since each Counter position uses the same Reference Data as the centre, it is safe to assume that the data received from the Counter is correct. The BAL services can skip the onerous task of validation. Futher details are in *HNG-X Architecture Branch Access Layer* (ARC/APP/ARC/0004).

In HNG-X, delivery of reference data to the Branch Counters is either via the Branch Database or via SYSMAN (2 or 3). The Reference data delivery architecture ensures that both delivery paths see the reference data purely as files and require no awareness of the file contents.

RDDS delivers reference data files and associated inventory updates to the Branch Database by using the Branch Database harvester. RDDS also does the necessary housekeeping on deliverables within the Branch Database by calling an interface. Further details are in *HNG-X Reference Data Architecture* (ARC/APP/ARC/0001).

The delivery of emergency Reference data and currency spot rates is through a process, which runs on the Counter and polls the Branch Database periodically to check whether any new information is present.

2.5.3.7 Recovery and Reversal Data

The Recovery Data store captures information to ensure correct behaviour of the Counter application in error situations (e.g. banking recovery records).

The Reversal Data store also captures guaranteed [C0] Reversals to ensure correct outcome of on-line transactions. The Branch Database harvester passes this data to NPS in near real time.

2.5.3.8 Service Level Reporting (SLT) Data

The Branch Database produces some of the SLT reports required for the HNG-X contract. Some examples are SLT reports for Reference Data delivery, Counter to Data Centre online performance and Capacity Management. The individual Topic Architecture and Design documents describe the information gathering process for the reports they need to provide. The overall architecture for SLT reporting is in Customer Services Architecture document (ARC/SVS/ARC/0002). This document identifies the data sources, their relationship and information routes for the SLT reports required for the HNG-X contract. To support SLT reporting, the Branch database provides a reporting mechanism similar to the one used in TES and APOP. The mechanism allows for overnight calculations, aggregations, and report delivery in CSV or XML format. DE/HLD/012 (PVCS) and AP/HLD/005 (PVCS) describe TES and APOP reporting mechanisms respectively.

2.5.3.9 Branch Information

Estate Management (EMDB) informs Branch Database of any changes in branch topography up to 10 days in advance. Where necessary it also supplies values such as IP addresses to enrich the data already provided. This happens the evening before the changes in topography become live.

2.5.4 Deletion of Data

Branch Database deletes (house keeps) data when one of the following happens – retention period expires, a set of criteria are met (for example status flag is set to a specified value) or both. For flexibility, metadata in the Branch Database controls the deletion process. Below are some examples.

- For tables in the Main Transaction data store the retention period is 3 daily partitions which retains data for up to 3 days. After sending the detailed transaction data to external interfaces, it is no longer required in the Branch Database. The retention period is to facilitate support.

FUJ00091411

- For branch reporting and other back office functions only a small subset of the attributes are required. These attributes are stored in the Reporting data store for period that exceeds the branch-trading period (as with the Horizon Message store retention period). This period is 60 days. A small proportion of the reporting data needs to be stored for 84 days for the Remuneration report.
- Audit data is also stored for a period of up to 3 days.
- Branch specific data such as usernames, stock unit details are stored indefinitely.
- Other data such as opening balances, cut-off markers are stored for given number of trading periods.
- Abnormally terminated user sessions are stored until recovered.

A combination of the Branch Database and Branch Support Database archives the data to flat files before deleting the data. The main recipients of the archived data are the Audit system and the SSC Archive server. The Audit system only receives data archived from the Journal tables. The data remains un-altered in the Audit System and is stored for a period of 7-years. Some legal proceedings may use audit data as evidence. The SSC Archive server receives all the data.

Also note the Branch Support Database provides greater retention periods for some data categories to aid the SSC, see Branch Support Database High Level Design (DES/APP/HLD/0023).

2.6 Branch Access Layer

The Branch Access Layer (BAL) supports a number of Web Services that the Counter can invoke.

2.6.1 Overview of Branch Access Layer processing

Requests (messages) from Counters arrive at the Web Services in the BAL in XML format. It is the responsibility of the Service Handlers to determine the nature of the message and to process the message appropriately. BAL Service Handlers are responsible for processing requests for On-line services such as Banking, Debit card and DVLA. For some On-line services such as Banking, the Service Handler writes a recovery record in the Recovery table. If there is a failure before the session is complete, recovery records are necessary to trigger a recovery process on the Counter. Counter recovery process ensures that the settlement of a failed session happens correctly. BAL Service Handlers also capture transaction and settlement information resulting from completion of customer sessions and other activities within the Branch estate. They are also responsible for Branch reporting and accounting.

The data in Session, Recovery, Event and other messages needs to be committed to the Branch Database in the same database transaction (i.e. by use of a single commit statement). Branch reporting and accounting use this data. Other host applications such as TPS, DRS and APS, further process this data and send it to external clients of Post Office Limited (POL). POL external clients include utilities and financial institutions like Streamline.

When required the Service Handlers write the XML message to the journal table. The journal table has two main requirements. It supplies data to the Audit system to satisfy legal requirements. The journal table also performs duplicate checking. This prevents the processing of a message more than once. The counter under error conditions can send the same message a number of times. Without the duplicate check, it is possible to process the same message more than once; a unique index is required on the journal table for duplicate checking. For duplicate messages, Oracle returns an index violation error to Service Handler. The Service Handler must return an appropriate response to indicate this outcome.

FUJ00091411

Post journalising and duplicate checking, parsing of messages takes place for further processing. Some settlement data does not require parsing. This is because the data has no use within the Branch Database or other host systems. Examples of this are:

- AP Additional Data
- Banking "C12" data

AP additional data passes directly to AP clients by the APS host and the DCSM agent does the Payment file generation from C2 data.

The following list is a categorisation of different types of messages (requests) processed by the Service Handlers in the Application Server:

- Reference Data
- Sessions (Transactions and Settlement)
- Transaction Recovery
- Events
- Reports
- Branch Messaging and Administration

The following sections describe the impact of these messages on different subdivisions (data stores) of the Branch Database.

2.6.2 Reference Data

To facilitate the delivery of reference data to Counters, RDDS maintains an inventory of the reference data packages available. The Branch Database stores the master copy of the inventory. Counter applications use the inventory to verify their reference data status. Counters with expired reference data can request delivery of the latest reference data from the Branch Database. BAL routes the reference data to the Counters. The delivery mechanism assumes that both the Branch Database and BAL see reference data purely as files and require no knowledge of their contents.

2.6.3 Sessions

Customer sessions consist of the following transaction types:

- EPOSS
- APS
- Banking, Debit Card & ETU Confirmations
- LFS Remittances
- Track & Trace
- Stock Unit Transfers
- Bureau Transactions (including revaluations)

Session data is any transaction that causes the movement of stock and financial value or movement of financial value or movement of data to the client. Recording of this transaction in the Branch Database is essential for Branch Accounting and Reporting purposes. Other host systems distribute this data to POL FS, HR SAP, POL Clients and other external interfaces.

BAL Web Services write Session data to the following data stores in the Branch Database:

1. **Journal**

FUJ00091411

The full XML Message requires capture within the Journal table. The Audit system receives data from this table.

2. **Main Transaction**

Session data needs to be analysed to determine its type and then written into the appropriate data tables within the Main Transaction store. This document categorises the transactions within a session into four main areas.

- **Transactions**

  This includes:

  - EPOSS
  - APS
  - Banking, Debit Card & ETU Confirmations
  - Stock Unit Transfers
  - Bureau Transactions
  - Transactions that need to be routed to POL FS, HR SAP, POL MIS, AP Clients, Data Warehouse and FRTS via the TPS application. The extract of this data by TPS is daily and based on the Trading Date.

- **C12**

  - The C12 message contains Banking, Debit Card & ETU Confirmations. DRS application receives this data as a single daily feed. Based on the C12 feed, the DRS generates a single daily C2 feed for TES and DCSM agent. The DCSM generates a Payment file based on the C2 data sent by DRS.

- **LFS**

  - LFS transactions include Pouch Delivery and Collection as well as Cash Declarations. The timeliness and content of this transactional information is different from all other external interfaces. The description is in *Logistics Feeder Service High Level Description* LF/DES/003 and *LFS to SAPADS and SAPADS to LFS AIS* (BP/DES/023). Partitioning of the table should be able to support periodicity of data extract by the LFS service.
  - Details of Pouches used for Remitting In and Out

- **Track & Trace**

  - EDG receives Track & Trace transactions from the NPS database. Branch Database supplies NPS with Track &Trace transactions in near real time.

- **Transfers**

  - Pending transfer information.

- **Postal Services Inventory**

  - Bar code inventory to support local collect and Mails despatch processes.
  - Record of Date for Last Posting per Carrier

3. **Report**

BAL also writes a subset of Session data to the Report data store. For full details of the session data that requires capture for reporting purposes, see section 2.6.5.

FUJ00091411

The main use of the data within the Main Transaction Store is to pass such data on to the external interfaces (via other host applications).

2.6.4 Transaction Recovery

BAL Web Services recognise messages that require recovery records and insert these records in to the recovery table. The BAL then routes the request to the correct On-line service if the transaction is for a third party product sold in the Post Office such as banking.

2.6.5 Events

Events such as log-on, log-off, print requests, etc., are mainly committed to the database outside a customer session. BAL writes the Event data to the following data stores:

- **Audit**

  The compressed full XML Message requires capture within the Journal table for auditing.

- **Transaction**

  The TPS application copies some events to the POL MIS system. The Counter application recognises these events and transfers an additional record as part of the event "session". This additional event transaction record is stored in the transactions part of the Main transaction store for the sole purpose of transferring this information to the POL MIS system via the TPS System. Refer to *Horizon to POL MIS Application Interface Specification* (EA/IFS/006) for further details of event transaction harvesting to POL MIS.

- **Report**

  BAL also writes Event data to the Report data store.

2.6.6 Reports

This section describes reporting data written to the Branch Database. Section 2.7 describes the use of this to produce aggregations and reports.

BAL writes a subset of attributes from the Event data and Session data into the Reporting data store. The structure of the Reporting data store minimise row size to aid efficient querying of this data.

Analysis of the Reports and Receipts (*HNG-X Reports and Receipts* (DES/GEN/SPE/0004)) shows that the Event Log report uses distinctly different attributes to other Counter and Branch reports. It is therefore useful to keep the two separate.

Further details on reporting are in DES/APP/HLD/0053 and DES/APP/HLD/0109.

2.6.6.1 Event Reports

The attributes reported on are FAD, Stock Unit, User Name, TP (Trading Period), BP (Balance Period), Node, Event Date/Time, Event Name and Event Description; and are stored in the Event report table.

2.6.6.2 Session Reports

BAL writes Session data in the Session reports table. The main attributes necessary for reporting are FAD, Stock Unit, TP (Trading Period), BP (Balance Period), Product ID, Mode Code, Transaction Date/Time, Node ID, Session ID, Transaction ID, Volume, Value, and User Name. Some further attributes, such as Bank Sort Code, Account Number or Cheque Number, are not always present and are

FUJ00091411

required for a small number of reports. There is therefore no reason why these data items cannot be stored as an XML in a single column within the table. This provides an easy and extensible mechanism for adding new reporting attributes should a need arise in the future.

In addition to the necessary storage of the above data, additional data needs deriving from reference data as part of the presentation phase of the reporting functionality. The different ways of using Reference Data during reporting (such as Product Name, Accounting Node description etc) are described below:

1. Store the Report data in a normalised form (as codes only – no descriptions). This minimises the amount of data stored, maximises the number of rows to each block and therefore minimises the time taken to query the report tables. However, the results need joining to the reference data in order to enrich the result with the necessary product descriptions etc in order to present a readable report.
2. Store the Report data in a de-normalised form along with all the information necessary to facilitate printing on the report. This does not require any joining with reference data during the report query activity but, as a result, causes larger tables and fewer rows per database block – thereby slowing query performance.
3. Store the Report data in a normalised form and enrich the query results with the necessary product descriptions within the Counter. This maximises the query performance and minimises the data volumes sent to each Counter. However, it increases the complexity of the Counter reporting application and requires additional reference data at the Counter The reference data is also required for a longer period.

Option 3 is the choice because it exerts the least load on the Branch Database and is the most scalable. However, it does add some additional development cost for the Counter application.

2.6.7 Branch Messaging and Administration

The Branch Data store is a repository for all the types of information that need to be readily accessible to the Counters within a Branch for the purposes of messaging and administration such as user and stock management. The following sections provide more details.

2.6.7.1 Branch and Counter Data

The Branch and Counter data store contain details about Branches and the Counters within the branches. Counter details include its position in the Branch, its IP address, sequence numbers described in section 2.4.1 and the termination status of the last logon session. BAL uses the termination status to trigger recovery on the Counter for aborted sessions. BAL during the logon process uses Counters IP address to verify the authenticity of the logon request.

Branch data mainly includes the list of branches that are open (hence allowed to trade).

Estate management systems supply changes to the physical Branch estate such as opening and closing of offices, addition and removal of counters and branch relocation to the Branch Database. The changes come from Estate management application (EMDB).

2.6.7.2 User and Stock Unit Data

This data store contains details about:

- Users and Passwords – Most users are local and only unique within a logical branch. Logins are restricted to single session within user's logical branch. In addition, the HNG-X system also supports a small number of centrally administered users. These global users are valid in all branches. BAL is responsible for managing user's and their passwords.

FUJ00091411

- Stock Unit management – Stock units are local and unique with a logical branch. Stocks are either single or shared. Shared stock units can have multiple users using it simultaneously. User attachments to Stock Units – The BAL is responsible for attaching users to stock units based on rules identified in the Use Cases.

2.6.7.3 Transaction Corrections

Branch Database receives Transaction Corrections from the SAP Financials system through the TPS application. The transaction corrections remain active until the postmaster actions them. The postmaster must action them before rolling over in to a new Accounting period. Once actioned, the Counter application must mark the transaction corrections as processed in the Branch Database.

The number of transaction corrections is small – refer to *POL Finance Systems to TMS/Horizon Transaction Corrections Interface Specification* (EA/IFS/002) if more detail is required.

2.6.7.4 LFS Pouch Details

LFS requires various persistent stores to record information about the content of Stock, Cash and Foreign Currency pouches.

- Replenishment Delivery Notices

The SAPADS system sends these notices via the LFS host application. The Counter uses these during the remittance of Cash or Foreign Currency pouches. After a few days (configurable) the messages are no longer required (whether or not they have been accessed).

- Pouch Contents

The Counter LFS application needs to know the contents of each packed Pouch ready for collection. The data captured by the Counter LFS application is useful for Pouch Reversals or for sending to SAPADS after the pouch is despatched. Counter application does Pouch validation to check for duplicate pouch numbers within the Branch. The application also stops illegal operations on the Pouch (for example a despatched Pouch can't be reversed).

- Collection Sack Details

The postmaster places the pouches in a Collection Sack before they are collected. Once collected these Pouches are marked as dispatched. Note: only pouches in the Collection Sack can be marked as dispatched.

The volumes of Pouch data are relatively low – refer to *Logistics Feeder Service High Level Description* LF/DES/003 and *LFS to SAPADS and SAPADS to LFS AIS* (BP/DES/023).

2.6.7.5 Track & Trace Data

The Track & Trace requirement is similar to the LFS Pouch Data requirement. A Track & Trace barcode creation within a Branch happens when a Customer sends a parcel or when a Postman delivers a parcel to the Branch for Local Collect. After registration in the Branch, the Parcel goes through various state transitions. Validation of state transitions is similar to Pouch data to ensure that invalid operations cannot happen and there are no duplicates. A unique index on FAD code plus Barcode is sufficient to guarantee uniqueness. Partitioning of this data on FAD hash improves index performance.

Sole use of this data is to record and validate the state of Track & Trace barcodes in the Branch. The data is not for transmission to EDG. The Track & Trace tables in the Main Transaction Store (see section 2.6.3) are used for this purpose.

Design Proposal for Track and Trace Integration (AS/DPR/013) suggests that there is a maximum throughput of 11.66 transactions per second. Not all transactions create a new Track & Trace barcode

FUJ00091411

and therefore the number of Track & Trace records stored in the Track & Trace Data Store is less – possibly 1/3rd of that sum. Track & Trace data generally has a lifespan of less than 3 days and therefore the data store size is small.

There is a Track and Trace report of barcodes for despatch (waybill). This is a custom report that is outside the scope of normal Counter reporting. The source of the report data is the contents of this data store rather than the contents of the Report Data store.

Copying Track & Trace data destined for EDG via NPS is in near real time.

2.6.7.6 Message Broadcast & LFS Messages

POL target messages at the whole estate, sets of branches, or individual branches. These messages are stored in the Branch Database and read by users in the targeted branches to which they targeted. There is an expiry process run as part of overnight schedule to delete expired messages.

Additionally, SAPADS send a file of Planned Orders to the Branches. These are targeted to the branches. Planned Orders sent to Branches remain active until read and processed by the postmaster. Note that the LFS Advice Notices are no longer used and have been withdrawn from the interface specification *Logistics Feeder Service High Level Description* (LF/DES/003) and *LFS to SAPADS and SAPADS to LFS AIS* (BP/DES/023).

The volume of messages is low.

A process runs on the counters and polls the Branch Database periodically to check whether any new messages are present supports the delivery of above messages.

2.7 Reporting and Aggregation

This section describes the aggregations and production of reports within the Branch Database. Section 2.6.5 describes the reporting data written to the Branch Database.

2.7.1 Aggregated Data

The Aggregation process runs after midnight and summarises previous days captured data. These aggregations are the actual computed values/stock counts for each product within a Stock Unit and within the entire Branch. The aggregation attributes are FAD code, Stock Unit, Trading Period, Balance Period, Date of Aggregation, Product ID, Transaction Mode ID, Total Volume and Total Value.

These aggregations help in determining the value of a Stock Unit or the entire Branch at any time. To get up-to-the-second figures, aggregation totals from previous day are summarised together current days transactions. These figures are required during Balance and Trading Period Rollovers to ensure that the declared and the calculated figures match.

The aggregations use the Reporting Data as the source data.

2.7.2 Accounting Aggregates

This data store includes stock unit declarations of Stamps, Cash, Foreign Currency and other Stock for the purpose of stock unit accounting and Period Rollovers. Balance Period and Trading Period Brought-Forward figures will also be stored here (Both SU and Office totals).

FUJ00091411

2.7.3 Counter Daily/Weekly Cut-Off Reporting

These reports are run daily or weekly and contain all products that are associated with a specific Post Office Accounting Node that have been sold since the last time the Stock Unit manually cut off the report contents. The act of performing the cut-off zaps the contents of the (next) report. Subsequent reports only list product sales since the last cut-off.

At the point of cutting off, the BAL may write totals to the Aggregated Data Store. Office Daily/Weekly Cut-Off Reports may use these report totals.

At the point of cutting-off a report, all transactions since the last cut-off point are listed on the report. The cut-off should place a marker such that the reported transactions do not appear on the next report but the system must also ensure that all transactions appear once (and only once) on a cut-off report.

Further details on Cut-Off reporting are in DES/APP/HLD/0053 and DES/APP/HLD/0109.

2.7.4 Office Daily/Weekly Cut-Off Reporting

Some Office Cut-Offs are simply a summary of the Stock Unit (Counter Daily/Weekly) cut off reports. These Office Cut-off reports the details of all the Counter Daily cut-offs that have been done for the same Report Type since the last Office Cut-off for that report type has been performed.

The Cut-off report details described in the previous paragraph are selected and the totals (either by individual product or overall) are shown on Office report.

Some Counter and Office cut-off reports are mandatory before the Stock Unit/Office can rollover. These details about the reports are stored within the report configuration/meta-data. If a report is mandatory, then the Stock Unit rollover processing checks that the Stock Unit has been dormant since the last cut-off report. In addition, the Office rollover process checks that all the Stock Units have been cut-off and rolled over and the Office Report cut—off taken place.

Further details on Cut-Off reporting are in DES/APP/HLD/0053 and DES/APP/HLD/0109.

2.7.5 Other Branch Reports

In addition to above, the branches can also request additional reports from the Branch Database described in DES/GEN/SPE/0004. These include:

- Bureau de Change reports
- EPOSS Transaction and Event log reports
- Mails and Label reports

2.7.6 Service Level Target (SLT) Reports

A number of SLT reports required for the HNG-X contract. If required, Branch Database can facilitate the production of these reports during the overnight Batch schedule. The report generation mechanism is a combination of TES and APOP reports. TES supports CSV format while APOP supports XML format.

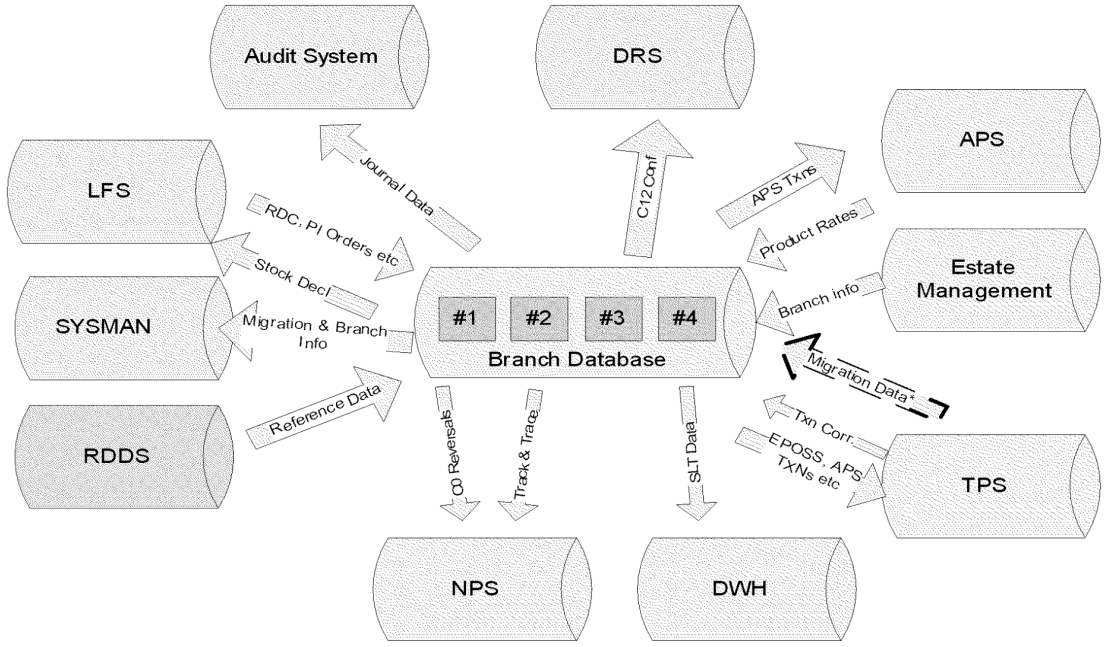
2.8 Other Host Interfaces

This section describes the flow of data between the Branch Database and other host applications. It also describes the architecture for supporting a new harvesting mechanism between the Brach Database and these applications.

FUJ00091411

2.8.1 Data Flows

The diagram below shows the flow of data between the Branch Database and other host applications.



EM Push!

Figure 5 – Data flows between Branch Database and other host applications

Peak volumes for copying data across the various interfaces are documented in HNG-X System Qualities Architecture (ARC/PER/ARC/0001).

In general the providers (source databases) push data to the consumers (target databases) when it is ready for delivery. This means that the consumers do not have to poll for the data thereby ensuring that the data arrives in the target databases in a timely manner.

2.8.2 Harvesters

As mentioned earlier), an Oracle RAC can provide high scalability if individual table partitions are accessed by a single node in the cluster. This is even more important when copying large quantities of data between the Branch Database and other host applications. Hence jobs copying data between the Branch Database and other host applications are FAD hash aware to ensure data is copied in or out from the correct node. Failure to do so can result in a large amount of inter-nodal traffic thereby affecting the performance of the copying job and the cluster itself. The Branch Database maintains a FAD hash partition to node mapping and use of this is mandatory to ensure that the correct node copies the data.

2.8.2.1 Design

To enforce the above the Branch Database provides a set of harvesters that are used by the other host applications to copy data in to the Branch Database and by the Branch Database itself to harvest data to these applications. The harvesters copy data to and from the correct node in the Branch Database. The detail design for the harvesters is in DES/APP/HLD/0020.

FUJ00091411

The harvesters are a collection of UNIX Shell scripts, Pro*C programs and Oracle PL/SQL procedures with a set of well-defined interfaces that can be used by other host applications and the Branch Database. DES/APP/IFS/0007 documents the interfaces. All the available nodes in the cluster are capable of running the harvesters.

The diagram below illustrates the harvesting mechanism.

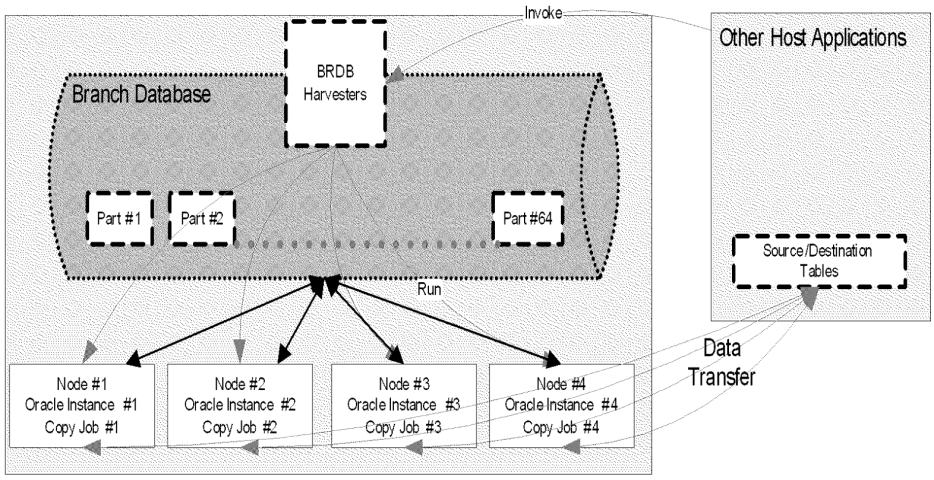


Figure 6 – Branch Database harvesting

The harvesters support both bulk and near real time copying of data.

2.8.2.2 Other host Applications

When the data is ready in the providing application, it remotely[5] calls the appropriate harvester to transfer the data to the Branch Database for consumption. The harvester then starts the copy jobs on each of the available nodes to transfer the data from the provider application to the consumer (Branch Database). The copy procedure uses the metadata in the Branch Database to ensure that each node transfers the data for the partitions it is managing.

The Audit system interface uses flat files. An overnight process produces these files.

2.8.2.3 Branch Database

When the data is ready in the Branch Database, a job locally calls the appropriate harvester to copy data to other host applications. Once again, the copy procedure uses the metadata in the Branch Database to ensure that each node in the cluster copies data for the partitions it is managing.

[5] Either directly or via a job through TWS schedule.

FUJ00091411

2.9 Training

2.9.1 Training Solution Architecture

HNG-X Solution Architecture Outline (ARC/SOL/ARC/0001) and HNG-X Counter Training Offices Architecture (ARC/SOL/ARC/0005) describe the HNG-X training solution. This section describes how the Branch Database architecture supports the training solution.

The diagram below shows the overall training solution architecture:

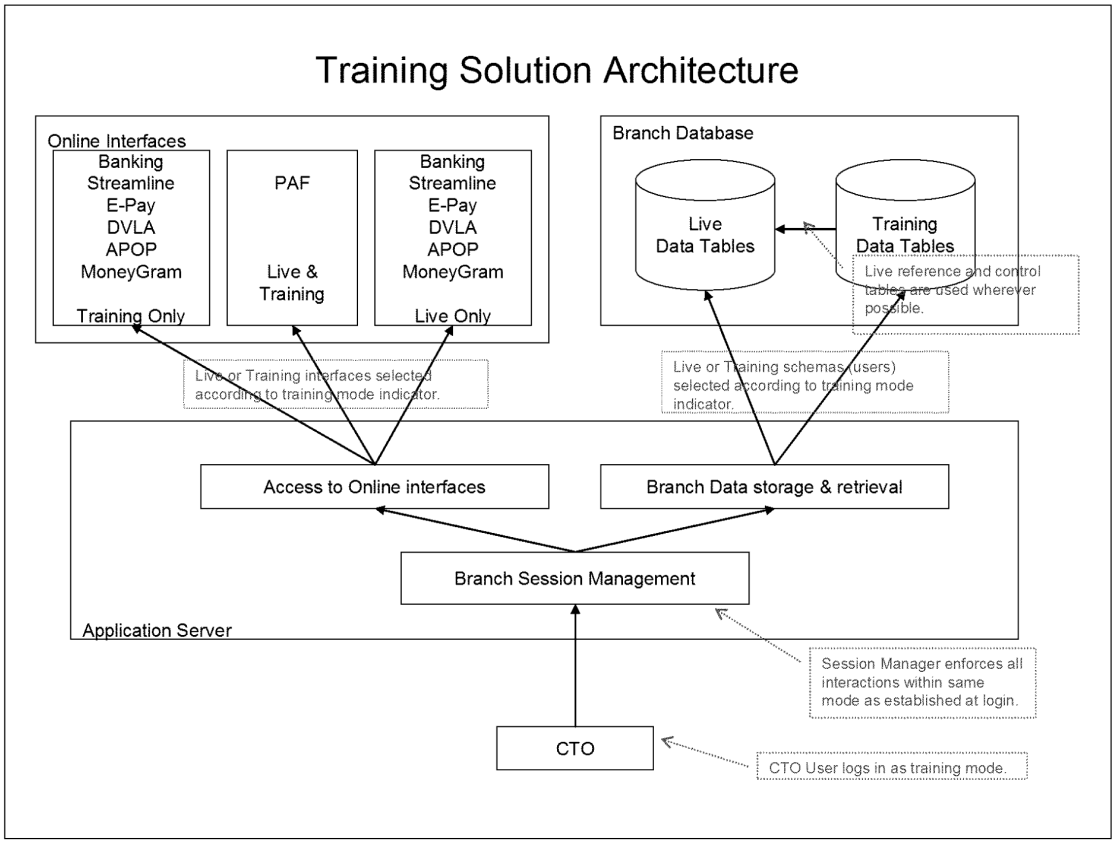


Figure 7 – Training Solution Architecture

2.9.2 Branch Database

The Branch database separates live and training data by using separate database users and schemas for the training and the live service. The training user has its own transactional, reporting and aggregation tables but shares the reference and control tables with the live user wherever appropriate. External clients or the audit streams do not receive training data because live and training data is stored in physically different tables. To maximise code re-use both live and training service use the transaction, reporting and aggregation tables in the same way. The tables also have the same name, structure and share the underlying storage. However, the training database user is the owner of these tables

FUJ00091411

A FAD code uniquely identifies a Post Office branch. Partitioning of Post Office data is by Branch (FAD code), because branches do not share data. Branch staff training takes place in special offices called Counter Training Office (CTO). Individual trainees do their training courses in virtual branches. Implementation of virtual branches is by assigning an Accounting FAD code (Branch_Accounting_Code) at the beginning of the training course. The accounting FAD allocation is static and uses the CTO FAD code (Branch_Code) and Counter position to look it up. The accounting FAD, is a number in the range 910000 to 919999. The CTO branch code is in the range 900000 to 909999.

The Branch Database also provides a reset capability so that each Counter can start a new training course with a fixed starting position. The fixed starting position is determined from metadata tables within the Branch Database. For flexibility, the metadata tables can support different combinations of fixed starting positions though currently we only expect to generate one such set.

The training data is deleted by the reset function. Log On to a Training Branch is not allowed more than N days since the last reset.

Training table sizes are approximately 1% of the live volume. They share the storage (tablespaces) with the live tables.

2.1.3 Branch Access Layer

The Branch Access Layer (BAL) must use the training user for capturing session data and running reports from training sessions. For all other sessions, the live user is used.

The Branch_Accounting_Code is used for inserting session data, retrieving data for branch reporting, stock unit rollovers and routing SQL statements. The Branch_Code is used for retrieving reference data etc.

2.1.4 Counter

The Counter software allows users to start new training session or reset the session. During the logon process the Counter is assigned its pre-allocated Branch_Accounting_Code by the Branch Database. All training messages between the Counter and BAL have a Branch_Accouting_Code (i.e. it is mandatory) and training flag.

FUJ00091411

3 Platforms

The Branch Database is a Linux RedHat 4 Oracle Real Application Cluster (RAC) based on Fujitsu-Siemens/Egenera BladeFrame (BF400) technology. The Oracle RAC consists of four identically specified Processing Blades (Pblades) sharing a common storage.

Each Pblade has a minimum of 32 GB Memory with 4 CPU's and 2 network interfaces (eth0 and eth1) defining Public (eth0), Private and Virtual network connectivity (both on eth1).

The operating system deployed on the RAC nodes is RedHat 4 AS 64 Bit. Each node has its own operating system locally installed on SAN disks. The operating system swap space exists on an independent disk.

Each node has its own copy of Oracle software locally installed on SAN disks.

A BladeFrame Internal switch is set up for interconnect traffic. This switch has eth1 plumbed for each node; and is referred to as the Private Interconnect.

A public network connection for normal node access is also set up. The Virtual network connections use the public network connection for its additional intercommunication.

Shared SAN disks provide storage for the Branch Database. All RAC nodes share the disks as common database disks.

SAN disks provide storage for OCFS2 filesystem for shared files such as OCR (cluster registry) and VOTING disk.

Oracle ASM provides storage for the Branch Database.

All RedHat Packages that are required to comply with Oracle Standards for a RAC build are applied.

All the four Pblades must be close to each other in the same BladeFrame. This improves performance as the RAC nodes all share the same fast interconnect backplane.

Oracle Clusterware 10.2.0.4 provides the Cluster Manager for Oracle RAC.

Oracle Enterprise software 10.2.0.4 with partitioning should be installed.

Block devices and ASMLib offers better performance than raw character devices.

For Backups RMAN (Recovery Manager) is used. Branch database is a 24 x 7 system and requires hot backups. Level 0 (full) backup is done on Sundays and Wednesdays and a Level 1 (incremental) backup is done on the remaining days. Standby Branch Database does not require a backup. However, when the Standby database becomes live, all normal operations such as running the TWS schedule, taking backups etc. are applicable.

Note: The Branch database high-level design provides a detailed section on backup and recovery.

The database storage requirements are Branch Database components are in the sizing spreadsheets DES/APP/HLD/0022 and DES/APP/HLD/0024.

Standby Branch Database is identical to the Branch Database.

Table below lists the platform codes allocated for Branch Database components and where to find the platform definitions.

Platform Type	Platform Code	Physical Platform Specification	Operating System	Instances
Branch Database	BDB	DES/PPS/PPD/0015	Red Hat Linux 4	4
Branch Database Server Standby	BDS	DES/PPS/PPD/0072	Red Hat Linux 4	4 (1 while replicating, 4 when operational)

FUJ00091411

Branch Support Server	BRS	DES/PPS/PPD/0073	Red Hat Linux 4	1

Data exchanges between SQL Server databases such as EMDB and Oracle Branch Database are initiated by the SQL Server database. This is using ODBC 'Linked' tables. Suggested approach avoids the need for buying additional software for the Branch Database platform.

FUJ00091411

4 Networks

Branch Database component systems plumb in to the Data Centre LAN infrastructure in conformance with the Network Domains model defined in the *HNG-X Technical Network Architecture* (ARC/NET/ARC/0001).

The Branch Database is accessible from all the Demilitarized Zones through a firewall. Client connections to the Branch Database and Standby Branch Database are restricted from Applications within the Data centre. Client connections to Branch Support System are allowed from workstations (e.g. SSC workstation). The Oracle Listener manages the network traffic between the Oracle Database and clients. The Listener listens on a specific network port and forwards network connections to the Database. For security reasons the default port is not used. The Listener security complies with Oracle Security guidelines and those specified in ARC/SEC/ARC/0003 and HADDIS.

All the four Branch database nodes are close to each other in the same BladeFrame. All inter nodal communication is done through the backplane.

FUJ00091411

5 Manageability and Supportability

5.1 Manageability

Systems Management of the Branch Database is within the monitoring framework described in ARC/SYM/ARC/0003. It is also supported by System and Estate Management (ARC/SYM/ARC/0001 - Timesynch and Scheduling), Provisioning (ARC/SYM/ARC/0007), Software Disribution and Asset Management (ARC/SYM/ARC/0002) and Remote Support (ARC/SYM/ARC/0004).

The manageability conformance required includes:

- Use of SYSLOG and or diagnostic logs
- Documenting recommendations for forwarding of any information in these logs to the Enterprise management system and the subsequent operational action required.
- Use of Oracle Enterprise Manager (OEM) Grid Control with the diagnostics pack for monitoring and managing the Branch Database
- Documenting recommendations for forwarding of any alerts from the OEM to the Enterprise management system and the subsequent operational action required.

5.2 Supportability

Third line support (SSC) access to the Branch Database is limited and controlled. This is to safeguard the performance of the Branch against the support actions of SSC. A separate support database (single instance) is available for third line support. Some of the key features of the Branch Support System (BRSS) are:

- Branch Support System (BRSS) is the repository for historical branch data. BRSS contains 60 days of detailed transaction data and 180 days of reporting and aggregation data.
- Oracle Streams replicates the information in the Branch Database and populates the BRSS database. The streams replication is asynchronous to minimise the performance impact on the Branch Database. The replication is logical; this allows support to use the BRSS database while replication takes place in the background. The BRSS database is no more than 15 minutes behind the Branch Database.
- To meet the security requirements of the HNG-X solution all support actions require auditing.
- Third Line support use BRSS for data analysis and manipulation. However, third Line support has no permission to modify the historical branch data. Any corrections are only applied to the Branch Database after approval.
- The BRSS provides a 'scratch area' for third line support users to create temporary tables, import dumps etc. SSC manage the scratch area and Core Services (ISD) do the routine DBA tasks.

FUJ00091411

6 Security

The Branch database conforms to the security framework described in *HNG-X Technical Security Architecture* (ARC/SEC/ARC/0003) and HNG-X Host Applications Database Design and Interface Standards (HADDIS) (DES/GEN/STD/0001). Branch Database components are in 'Core Services PCI-CE Domain' security domain.

6.1 Application

Applications connecting to Branch Database use an Oracle user. The user has minimum privileges necessary to function. Refer to *HADDIS* for more details. Applications accessing the Branch Database either use the Key Server (KMNG) or Oracle client-side Wallets to store password credentials for connecting to the Branch Database.. Hard coding of passwords in application code is not acceptable.

The Branch Database implements the following security guidelines from *HNG-X Technical Security Architecture* (ARC/SEC/ARC/0003) and HADDIS:

1) Install only what is required
2) Lock and Expire Default User Accounts
3) Change Default User Passwords
    a) Change default passwords of administrative users
    b) Change default passwords of all users
    c) Enforce password management
4) Enable Data Dictionary Protection
5) Practice the principle of least privilege
    a) Grant necessary privileges only
    b) Revoke unnecessary privileges from the public user group
    c) Grant a role to users only if they need all privileges of the role
    d) Restrict permissions on run-time facilities
6) Enforce access controls effectively and authenticate clients stringently
    a) Authenticate client properly
7) Restrict Operating System Access
8) Restrict Network Access
    a) Use a firewall
    b) Protect the Oracle listener
    c) Monitor who accesses your systems
    d) Check network IP addresses
    e) Encrypt network traffic
9) Apply all security patches
10) Audit all access to Branch Database

FUJ00091411

6.2 Platform

Access control for database users is by a role-based access control system. Provisioning of the system is by the Identity and Access Management security service. Provisioning of service or application accounts and human user accounts is through Active Directory (see DES/APP/HLD/0020 for these accounts). If Oracle users are required for these accounts, then their provisioning is through scripts supplied by the host development team.

The Oracle Database platform will be hardened following the guidance in the Vulnerability Management section of ARC/SEC/ARC/0003 relating to Red Hat Enterprise Linux 4.

Auditing of database access from the UNIX command line is necessary. The guidelines for auditing are in *HADDIS*.

6.3 Data

Access to data is restricted to users who are authorised to access the information. When providing data access principle of least privilege is followed. Data cannot be read, created, modified or deleted without authorisation.

FUJ00091411

7 Recovery and resilience

The overall solution design minimises disruption to Post Office trading. The architecture and design of the Branch Database supports non-stop (24x7) trading, other than for agreed maintenance outage windows.

7.1 Hardware and software failure

Oracle RAC provides a high degree of resilience. If the hardware or software on one node fails, the surviving nodes carry on running. BAL re-distributes the workload of the failed node on to the surviving nodes. The workload distribution is through re-routing of messages based on metadata stored in the Branch Database. The workload distribution is even to ensure that the surviving nodes remain well balanced. Bladeframe also provides a high degree of hardware resilience, see ARC/PPS/ARC/0001.

The node configurations chosen ensure that the system can run at its design limits if a single node fails.

7.2 Data corruption

It is possible, but extremely unlikely, that the data written to disk is corrupted. However, Oracle will identify this type of data corruption and raise an alert. In most cases, data corruptions can be resolved without affecting services. In rare cases, the recovery of the entire Oracle database may be necessary. This can take many hours, as data recovery is from backups and then archive logs have to be re-played.

To avoid this situation, and ensure uninterrupted service to Post Office the system keeps an additional local copy of the data. The maintenance of the local copy is by using Oracle Data Guard. This acts as a standby database. By using a standby database, restoration of trading is within minutes. For performance reasons standby database and the Branch database must be in the same BladeFrame cabinet. The standby database uses the same nodes as live database while recovery of the live database takes place. This delivers similar performance to live. After live database has been restored, switch over to live from standby only happens outside core Post Office hours.

The high-level design or support guide will provide a list of recovery scenarios including the steps the DBA needs to follow to recover from the situation.

7.3 Disaster recovery

The DR Data centre has an Oracle RAC configuration that is identical to the live data centre. Both live and DR Data centres have identical copies of business data. Synchronous replication between the two Data centres helps in maintaining identical copies. No business data is lost if the primary Data centre suffers a catastrophic failure.

Local DR is through the standby Branch Database.

FUJ00091411

8 Performance

The Branch Database is required to support contracted volumes as defined in HNG-X System Qualities Architecture (ARC/PER/ARC/0001) and support Fujitsu Services' Service Level Agreements set out in Schedule C1. The key targets are:

- Capacity and Performance targets give in section 2.1
- Provide a highly available 24 x 7 database solution
- No single point of failure that causes the business facility to fail
- Failure should not cause capacity issues

The targets above presented a risk that the Branch Database could suffer from performance issues; this in turn could affect the activities performed within the Counter estate. The HNG-X Architecture pivots around the centralisation of data, hence the risk had to be assessed, quantified and mitigated very early in the solution lifecycle. , To mitigate this risk, prototypes of the Branch Database and load generators developed. , The load generators development had to simulate peak Session Capture and Reporting loads. The prototypes were close to the live implementation as possible.

Prototyping exercises conducted on the target BladeFrame Linux platform. The results have shown that the proposed architecture comfortably achieves the performance targets. Hence, any performance risks around Branch Database are successfully mitigated. The prototyping results are available in DES/GEN/REP/0001.

The Branch database simultaneously supports both Transaction capture and Branch reporting. This is can cause performance problems. Transaction capture rates depend on the customer activity within Branch estate and are difficult to predict. On the other hand, we have a better understanding of Branch reporting. By reducing reporting loads, it is possible to eliminate any performance problems. For example, the weekly reporting peaks are on Wednesday between 17:00 and 18:00 hours; and Wednesday's peak is double the peak on Monday. The Wednesday peak is due to the office rollover activity. By consolidating reports in to groups, it is possible to reduce the number of 'SELECT' statements run on the Branch Database. This in turn reduces the workload on the database. By grouping reports, multiple reports use the same dataset; but the report contents are unchanged.

FUJ00091411

9 Migration

This section approaches migration in two ways. The first three major subsections describe various aspects of the migration process, concentrating on particular topics rather than on the ordered phases of migration. The last major subsection describes how separate migration phases identified in DES/MIG/HLD/0001 and DES/MIG/HLD/0002 affect the Branch Database components.

The approach adopted results in some duplication of description, but neither view is sufficient on its own.

9.1 General

HNG-X Migration Strategy (ARC/MIG/STG/0001) has the overall Branch migration description. Other documents such as DES/MIG/HLD/0001, ARC/SYM/DPR/0002 and ARC/SYM/ARC/0005 also contain useful information. For Branch Database the primary focus is the migration of the branch data and building of history as shown in the figure below. An individual branch must trade exclusively as a Horizon branch or as a HNG-X branch on any given trading day, but never both.

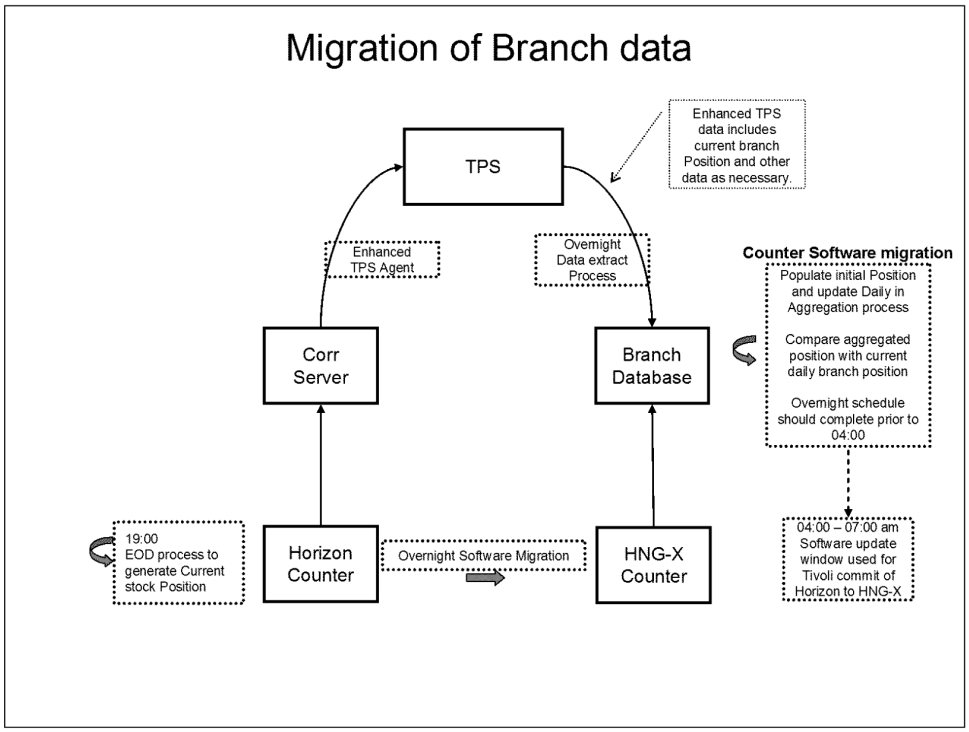


Figure 8 – Data Migration Process

The following categories of Branch data are migrated from Horizon to the HNG-X Branch Database:

1. Transaction History
2. Current stock positions (per Stock Unit)
3. A full list of the users in the system
4. Mapping of users to roles
5. A list of Stock Units (SU)

FUJ00091411

6. The stock units to which the users are attached
7. Cut-offs (markers) by report (markers)
8. Track and trace data for mails local collect (markers and barcodes)
9. Track and trace data for despatch (markers and barcodes)
10. LFS Pouch barcodes
11. Foreign currencies shown on rate board
12. Foreign currencies shown on rate report
13. Selected Riposte Objects, Messages and Local Persistent Objects - this is configurable through Reference data.

The transaction history (1) and current stock position (2) constitute the main volumes of data to be migrated. A daily process copies the transaction history and current stock positions to the Branch Database (known as Migration preparation). TPS host application is the source of this data. The remaining data is captured on the day of the migration (know as In-Day migration). The In-Day migration process is re-runnable. The In-Day migration data source is also TPS. Following is a description of Migration Preparation and In-Day Migration.

DE/HLD/023, EP/LLD/029 and EP/LLD/030 describe the Horizon Counter changes to support migration. These documents are in PVCS.

9.2 Migration Preparation

9.2.1 Transaction History

Following migration, the branch has access to previous (Horizon) transactions for use in reporting and reversals (where business rules allow). Cut-off reports run after migration pick transactions performed under Horizon but not cut-off. To support these business requirements the Branch Database builds transaction history. The history building must start several weeks[6] before the first branch is migrated to HNG-X. This following describes the history building processing.

1. A special process extracts transactions for all branches from TPS application to populate the transaction and report data stores within the Branch Database. This happens daily as part of the overnight batch schedule.
2. The normal Branch Database Aggregation processes run and aggregate the data copied by above (1).

Note: In Horizon TPS host application does not receive all the transaction data; for example, there are additional fields that are required for APOP and Personal Banking reports. The enhanced HNG-X TPS Agent harvests these additional fields. The fields are stored in a new column XML_DATA in TPS transactions tables. The Branch Database then gets a copy of theses fields.

9.2.2 Current stock positions

It is important to verify that the two systems (Horizon and HNG-X) are in step before migration of the branches to HNG-X can take place. This is done as per below.

[6] Ideally it should be 84 days to support the Remuneration Report. However, during the initial pilot, a shorter period may be OK, though restrictions will apply to the functionality available.

FUJ00091411

1. A new Counter End-Of-Day (EOD) process generates the current stock position for each Stock Unit for that day up to the EOD cut off markers.
2. Once switched on, the process runs as part of the EOD process every day.
3. A XML blob within a Riposte message records the data.
4. Initially this would need to go back to the latest SU cut over position data, and work forwards from that position.
5. Subsequently, it would start from the previous day's position and only need to work through new messages until the new EOD markers.
6. Apart from the initial use of latest SU position, it would not be necessary to use the subsequent SU position data[7].
7. An enhanced TPS Agent picks the stock position data. The agent harvests the data and loads it into new tables in TPS.
8. A special Branch Database process extracts the current stock position data to populate the Branch Database table.
9. On the very first run, the current position data populates the starting position for current stock and aggregation tables.
10. After the first run comparisons of Branch Database stock positions with the current daily stock positions from the branches takes place. The Branch Database generates reports showing any discrepancies in stock positions. Discrepancies require investigations by third line support.
11. The above reconciliation process runs for several weeks to ensure the consistency of data between the two systems.

9.3 In-Day Migration

The branch migration from Horizon to HNG-X should cause minimum disruption for Branch staff. Post migration the Branch has access to all the Horizon data it requires to conduct its business. To achieve this goal, transaction history (described in 9.2.1) and the following information on the day the migration is necessary. A special counter process extracts the In-Day migration information. The process writes this information as a XML blob in a Riposte message. Enhanced TPS Agent extracts and writes this information to migration tables in TPS. A special migration process copies, parses and stores the data in relevant Branch Database tables. The counter migration process extracts the following information.

- a full list of the users in the system
- mapping of users to roles
- a list of stock units
- the stock units to which the users are attached
- cut-offs by report (markers)
- track and trace data for mails local collect (markers and barcodes)
- track and trace data for despatch (markers and barcodes)
- LFS Pouch barcodes

[7] It may be appropriate to perform some additional validation of the calculated position against the new rollover data to check that no transaction data is missing.

FUJ00091411

- Foreign currencies shown on rate board
- Foreign currencies shown on rate report
- Data presented in the Pre Migration Report

The counter migration process is soft driven by reference data. This allows for late changes if the above list is incomplete.

Note: the Branch manager initiates the Branch migration to HNG-X after suitable training of staff. The Branch Database has no prior knowledge of a branch migrating.

Once, all the In-Day migration data is successfully processed, Branch Database raises an event so SYSMAN2 can issue a Tivoli commit of Horizon to HNG-X on migrating branches.

This is described in more detail in DES/APP/HLD/0113.

9.4 Post Migration

After successfully migrating to HNG-X, the Post Office manager accesses the Branch Database using the HNG-X Counter application to request 'Post Migration Reports. Once the reports are available and their contents are satisfactory the Branch can start trading in HNG-X mode. The Counter Business Application sets the Branch_Status flag indicating that migration is complete. Once the flag is set, Branch Database supplies a migration complete event (aka Point of No Return) to Estate Management (EMDB) and Systems Management (SYSMAN2) applications.

9.5 Migration Phases

This section describes the changes applicable to Branch Database components for the Migration Activities defined in DES/MIG/HLD/0001 and DES/MIG/HLD/0002. Branch Database components are Branch Database (BRDB), Standby Branch Database (SBRDB) and Branch Support System (BRSS).

9.5.1 Branch Router Rollout

This phase has no impact on the Branch Database components.

9.5.2 HNG-X Migration Enabling Upgrades for Data Centres

This phase has no impact on the Branch Database components.

9.5.3 Data Centre Build

This activity installs and commissions new hardware and software in the Data Centres at IRE11 and IRE19, initially for use in the ST, SV&I, RV, V&I and LST test environments. The installation is incremental as test environments are established. The Branch Database components are configured using tag files.

9.5.4 Move Wigan Network Management Servers

This phase has no impact on the Branch Database components.

FUJ00091411

9.5.5 Data Centre Preparation

This phase follows completion of V&I testing.

The Branch Database components need to be reset back to the initial state for live operation. The simplest approach is to repeat the build done initially (see 9.5.3).

9.5.6 Cutover Rehearsals

This phase has no impact on the Branch Database components.

9.5.7 Migration of POL FS

In general, this phase ("Weekend A") has no impact on the Branch Database components.

9.5.8 Migration of Batch Services

This phase ("Weekend B") has a small impact on the Branch Database components. Branch Database starts receiving migration preparation data described in section 9.2

9.5.9 HNG-X Specific Services

This phase has no impact on the Branch Database components.

9.5.10 Live Reference Data Proving

This phase has no impact on the Branch Database components.

9.5.11 Migration of Online Services

This phase ("Weekend C"), has no impact on the Branch Database components

9.5.12 Migration of Audit Services

This phase has no impact on the Branch Database components.

9.5.13 Migration of Branch Services

This phase ("Weekend D") has no impact on the Branch Database components,

9.5.14 Move Bootle Network Management Servers

This phase has no impact on the Branch Database components.

9.5.15 Decommission Wigan and Bootle

This phase has no impact on the Branch Database components.

FUJ00091411

9.1.16 Horizon Counter Changes for PCI Compliance

This phase has no direct impact on the Branch Database components.

9.1.17 Migration Enabling Upgrades for Counters

This phase has no impact on the Branch Database components.

9.1.18 HNG-X Application Pilot and Rollout

This phase has direct impact on the Branch Database components. As HNG-X Counters are rolled out, Branch Database starts receiving transactions are from HNG-X Counters via the HNG-X BAL. During branch migration, the Branch Database supplies migration information for SYSMAN to carry out the actual software upgrade.

9.1.19 Counter Event Management Changes

This phase has no impact on the Branch Database components.

9.1.20 Counter XP Upgrade

This phase has no impact on the Branch Database components.

9.1.21 Post-application ADSL Changes

This phase has no impact on the Branch Database components.

9.1.22 Final Decommissioning

This phase has no direct impact on the Branch Database components.

9.1.23 Estate Management Upgrade

This phase has impact on the Branch Database components. Branch Database starts to receive branch information such as counter IP addresses from EMDB.

FUJ00091411

10 Testing and validation

The Branch Database is pivotal to the HNG-X proposal. Extensive prototyping helped to mitigate any performance and scalability risk. To facilitate prototyping, Java based Session Capture and Reporting simulators were developed. DES/GEN/REP/0001 documents the design of the prototype and the results achieved. A modified prototype can help with the testing outlined below.

The following are some guidelines for testing and validation of Branch Database. Note the list below is only to facilitate testing by helping testers to design their testing and validation strategy.

- Build scripts

  Scripts for building the 4-node RAC cluster and Branch database need testing both on live and test size rigs. Manual verification of the configuration parameters set by the build script against the design documents is required.

- Unit testing

  All Branch Database components (modules) need standalone testing. These include partition management, aggregation, auditing, purging of old data (housekeeping), copying of reference data, harvesting and so on. This may require use of stubs, harnesses (similar to prototype above) etc. to enable testing in isolation.

- System testing

  System testing combines Branch Database modules that have been unit tested in to larger groups. In some cases, the larger group may require integration with other applications (for example BAL, Counter and other host systems) to test specific functional requirements. However, in this phase most of the functional and non-functional requirements can be tested using simulators, harnesses and stubs. The prototype after small modifications can prove to be a useful simulator / harness for system testing.

- Solution testing

  The purpose of solution testing is to verify functional and non-functional requirements of the HNG-X solution. In this phase, the testers must build the complete HNG-X solution by combining individual applications (Branch Database, Branch Access Layer and the Counter) and supporting infrastructure. From the Branch Database perspective, the testing must include the following

  - Ensure that Branch Access Layer can access all 4 nodes of the Branch database cluster
  - Targeting of database queries (SQL statements) at the correct node (Oracle instance) in the cluster
  - BAL handles node failure correctly and the workload is distributed evenly across the surviving nodes
  - Branch Database schema supports all the Use Cases and any functional and non-functional requirements from the SRS
  - Batch harvesting of data uses checkpointing. Failed copy jobs should restart from the last checkpoint
  - By adding and removing nodes (both in a controlled and uncontrolled way), the stability of RAC cluster is tested

- Migration testing

  The purpose of migration testing is to validate that all the required data is migrated from Horizon counter to the Branch Database. The modified TPS Agent is responsible for copying this data from Message store into TPS host. The overnight schedule copies the data in to the Branch Database.

FUJ00091411

- Performance and volume testing

  The testing in this phase must verify the performance targets for Branch Database identified in 2.1. Verification of performance targets requires a simulated environment. Testers should either write their own simulators or reuse parts of the prototype. The testing in this phase must also verify any system limits such as size of Settlement message by transacting a customer session with 1000+ items in the basket.

- Business continuity testing

  This phase includes:

  - Testing of Branch Database replication to the standby Branch database (for example. there is no loss of data even under peak load and node failures).
  - Test Branch Database backup performance.
  - Test the failover of live Branch Database to standby Branch Database.

Note: Testing must include SLT report generation from the Branch Database as part of the overnight schedule.

FUJ00091411

11 Risk and assumptions

11.1 Risks

This section captures the major risks and assumptions in this architectural space. This is of particular value early in the development of the architecture.

The Branch Database uses Oracle10gR2 Real Application Clustering technology, which is an industry proven database solution for high availability (24 x 7) systems. A number of case studies and white papers on this topic are available on the Internet and www.oracle.com.

Extensive prototyping helped to mitigate any performance risks. The prototyping took place very early in the solution lifecycle. To facilitate prototyping, Java based Session Capture and Reporting simulators were developed. The simulators used live[8] Horizon data to performance tests on the Branch Database. The HNG-X and Horizon data is similar. The target platform (BladeFrame and RHEL) had series of prototyping exercises performed on it. The results have shown that the proposed architecture comfortably achieves the performance targets in section 2.1. Hence, risk mitigation for the Branch Database has been successful. DES/GEN/REP/0001 documents the prototyping results.

Routine maintenance activities such as adding storage, reconfiguring storage or taking backups does not involve any downtime. The steps involved in performing such activities are already well established. Most of the availability targets for Branch Database also apply to NPS. NPS is 2-node RAC database used by Horizon and has been operational since 2004.

If any node in the RAC cluster were to fail, the remaining nodes share the workload equally. A cross reference (metadata) table mapping branches to cluster nodes facilitate this.

In the event of a node failure, Oracle RAC's inbuilt resilience detects that one of the servers has failed and carries on providing access to the Branch Database through the remaining nodes. The failed node can re-join the cluster during the overnight schedule.

There is sufficient CPU, Memory and I/O bandwidth available on a node to ensure that a single node failure does not cause capacity problems on the surviving nodes. The prototyping report highlights this and proposes a minimum hardware configuration.

Synchronous data replication between the live Data Centre and the DR Data Centre ensures that the replicated copy of the data is the same as the primary. For data replication, EMC[2] product SRDF is used.

The Horizon system uses SRDF for replication, and has done so for a number of years. There is complete confidence in this technology. EMC[2] are also world leaders in storage technology.

In the event of a Data Centre failover, starting the RAC cluster and opening the Oracle database are only actions required to resume the service. Based our experience of the 2-node NPS RAC cluster, we feel confident that the database can be made available for use well within the 2-hour failover SLA.

The SLA in case of catastrophic failure is 1 hour. The use of the standby Branch Database mitigates this risk.

By providing a Branch Support System (BRSS) for third line support, there is no impact on the performance of Branch Database.

The provisioning of maintenance activities like applying critical patches to software and firmware stills needs to be decided. Failure to apply critical patches can lead to failure and service outage. An approach to reduce this risk is under discussion.

[8] The HNG-X data is similar to Horizon.

FUJ00091411

Clients need to follow certain rules when accessing the Branch Database. The following section lists these rules. Failure to obey these rules can affect the performance and scalability of the Branch Database.

11.2 Assumptions

In order for the Branch Database to meet its functional, performance and scalability requirements, client applications have to follow certain rules. The assumption is that the client applications follow these rules. This section is an attempt to list them.

- The BAL separates Live and Training use of the Branch Database.
- The BAL targets SQL statements to the correct node based on a metadata (FAD hash mapping table) in the database.
- The minimum configuration required to support peak volumes is 3-nodes.
- When a node fails, the BAL must refresh its mapping by re-reading the metadata.
- Transient network failures have the same symptoms as node failure. BAL can distinguish between the two failures with the help of metadata (available nodes table) in the Branch Database.
- The BAL maintains the Trading Date. The Trading Day changes at 7pm every day. However, this must be configurable.
- To maintain cluster performance, other host applications must harvest data from the correct node.
- Support staff can only access the Branch Database when it is an emergency or to apply fixes. The Branch Support System (BRSS) is for support workload.
- The Journal Sequence Number is a dense set. The Counter plays an important role in this. The sequence must only be incremented when sending a journalisable (auditable) message to Branch Access Layer. The Counter application must terminate the user session if it fails to do deliver this message to BAL.
- Some reports (SQL) can take long time to run. Report Requests either should have large timeout value or not timed out by the BAL or the Counter.
- The BAL must write the data for a given Counter request in a single commit.
- For security reasons the application code must not have passwords hard coded in them.
- The Branch Database does not encrypt data on behalf of its clients. The clients are responsible for encrypting their sensitive data.
- The clients are also responsible for making sure that the data in the Branch Database is PCI compliant.
- Individual branches trade exclusively either as a Horizon branch or as a HNG-X branch on any given trading day, but never both.

FUJ00091411

12 Requirements traceability

Traceability of Business, Customer Service and System Requirements is detailed in a separate Traceability Matrix designated ARC/APP/RTM/0006.