FUJ00079425

Document Title: **EPOSS Operational Support Guide**

Document Type: Operational Support Guide

Release: CSR+

Abstract: This document provides details of the operational support considerations and guidelines provided in support of the EPOSS Product. It is intended for those staff needing to understand the operation, use and application of the EPOSS Software and associated Tools. This guide is not a definitive guide to the Riposte Service in which EPOSS is supported. The guide also covers the operational procedures and support provided by the EPOSS Development Unit and how these fit into the overall service delivery.

Document Status: APPROVED

Author & Dept: Phil Hemingway - EPOSS Development – Senior Team Leader

Contributors: Tim Canniffe, Phil Hemingway, Steve Kay, Richard Laking, Nam Pandher, Keith Toh, Steve Warwick, John Warwick, Walter Wright

Reviewed By: Walter Wright, Francesco Chiarini, Nam Pandher, Mik Peach

Comments By: *On-Going*

Comments To: Document Controller & Author

Distribution: Walter Wright, Francesco Chiarini, Nam Pandher, Mik Peach, Pete Lindsey, Kevin Barrett

FUJ00079425

0 Document Control

0.1 Document History

Version No.	Date	Reason for Issue	Associated CP/PinICL No.
0.1	27/04/00	DRAFT Issue	N/A
0.2	28/09/00	Merged Document	N/A
1.0	30/10/00	APPROVED	N/A

0.2 Approval Authorities

Name	Position	Signature	Date
Mik Peach	Customer Service Manager		

0.3 Associated Documents

Reference	Version	Date	Title	Source
EP/DOC/002	0.1	18/07/99	EPOSS Development – Documentation Roadmap	ICL Pathway
EP/DES/024	0.1	02/05/00	EPOSS Glossary of Terms	ICL Pathway
EP/DES/019	0.1	11/04/00	EPOSS High Level Design	ICL Pathway
EP/DES/020	1.1	01/08/00	EPOSS Reporting Service High Level Design	ICL Pathway
EP/DES/021	1.1	10/08/00	EPOSS Balancing Service High Level Design	ICL Pathway
EP/DES/022	1.0	27/07/00	EPOSS Transaction Service High Level Design	ICL Pathway
EP/DES/025	1.0	11/07/00	EPOSS End of Day Service High Level Design	ICL Pathway
EP/DES/023	1.0	12/07/00	EPOSS Migration Service High Level Design	ICL Pathway
EP/MAN/002	0.1	26/05/00	EPOSS Support Reference Guide	ICL Pathway
EP/MAN/003	0.2	26/05/00	EPOSS System Error Catalogue	ICL Pathway
TD/ARC/001	4.6	22/03/00	Technical Environment Description	ICL Pathway

FUJ00079425

Tbs	Tbs	Tbs	Counter Management Support Guide	ICL Pathway
IM/MAN/012	1.0	04/08/99	Horizon Field Support Officer Guide Release 2 Section 4 (Manual Outlet Procedures)	ICL Pathway
IM/MAN/013	1.0	04/08/99	Horizon Field Support Officer Guide Release 2 Section 5 (ECCO Outlet Procedures)	ICL Pathway
EP/DES/002	6.1	02/10/98	EPOSS Attribute Grammar Catalogue	ICL Pathway
Tbs	Tbs	Tbs	Riposte Attribute Grammar Catalogue – Messages	ICL Pathway
TD/DES/109	2.0	03/04/00	Counter Application Scheduler High Level Design	ICL Pathway
TD/STD/005	0.1	23/06/00	Coding Standards	ICL Pathway
EP/DES/146	0.1	03/07/00	Physical Design for Counter at CSR+	ICL Pathway
LF/MAN001	0.1	05/06/00	Logistics Feeder Service Systems Manament Support Guide	ICL Pathway
BP/TRN/001	11	06/04/00	ICL Pathway Training and User Awareness Baseline Document	ICL Pathway

0.4 Abbreviations/Definitions

Abbreviation	Definition
API	Application Interface
APS	Automated Payment System
BES	Benefits Encashment Service
BP	Balance Period
CA	Cash Account
CAP	Cash Account Period
CP	Change Proposal
CS	Correspondence Server
CSR	Core System Release
DDN	Declaration Discrepancy Negative

FUJ00079425

Abbreviation	Definition
DDP	Declaration Discrepancy Positive
DEQ	Date Equal to
DGE	Date Greater than or Equal to
DLL	Dynamic Linked Library
DLE	Date Less than or Equal to
ECCO	Electronic Cash-registers On Counters
EDSC	European Development Support Centre
EOD	End of Day
EODHT	End of Day Harvest Trailer
EPOSS	Electronic Point Of Sale System
HFSO	Horizon Field Service Officer
HLD	High Level Design
LFS	Logistics Feeder System
LLD	Low Level Design
MAS	Migration Agent Server
MiEcco	ECCO Migration
MiMan	Manual Migration
OBCS	Order Book Control Service
ONCH	Cash On Hand
PC	Personal Computer
PI	POCL Infrastructure
PLU	Product Look Up
PinICL	Problem report in ICL's Incident/Defect Management System
POCL	Post Office Counters Limited
PVCS	Project Version Control System
QFP	Quality Filtering Process
RDMC	Reference Data Management Centre
REM	Remittance
RISD	Remittance in Supply Division
ROOP	Remittance Out Other Post Office
SAP	Systems and Application Products

FUJ00079425

Abbreviation	Definition
SAPADS	SAP Auto Distribution Centre
SSC	{ICL Pathway} System Support Centre
SU	Stock Unit
TED	Technical Environment Description
TIP	Transaction Information Processing
TPS	Transaction Processing System
UP	Unclaimed Payments
UR	Unclaimed Receipts
VB	Visual Basic
WBS	Work Breakdown Structure

0.5 Changes in this Version

Version	Changes
0.1	First Draft
1.0	Incorporation of changes following first formal review.

0.6 Changes Expected

Changes
Further Refinement and Clarification
Rework as a result of Review Comments
Completion of acknowledged missing inserts
Trial Use and Review from EDSC
Trial Use and Review from EPOSS Development Team
Addition of further EPOSS Systems Management and Problem Diagnosis processes as they become recognised. Examples may be identifying reference data dependencies, describing print management problem resolution or explaining the impact of expired messages
Update EPOSS NT Event Log Usage and EPOSS Audit Log Usage Catalogues with further clarification of reasons for logging Events
Include section describing the necessity for the counter message store to be provided on occasion rather than the correspondence server message store
Include Section(s) describing fault diagnosis in the EPOSS End of Day Processes
Table of Figures requires updating so that each figure is titled

FUJ00079425

0.7 Table of Contents

FUJ00079425

FUJ00079425

FUJ00079425

1 Introduction

This guide is intended for those staff needing to understand the operation, use and application of the Pathway EPOSS Product. The guide also covers the operational procedures and support provided by the EPOSS Development Team and how these fit into the overall service delivery.

It is one of a set of guides, which together cover all areas of systems management as deployed in the Horizon solution.

This document describes the recommended approach to analysing live calls passed to 4th Line Support for investigation.

4th Line Support is defined as the level of support where an incident, which has occurred on the system has been investigated and analysed and, in the opinion of the 3rd Line Support organisation, evidence exists which may indicate that there is a fault in the production software. Calls may also be passed down from 3rd Line Support where the support staff are simply unable to establish why the incident occurred.

4th Line Support are responsible for carrying out further analysis of the incident to confirm whether a genuine software fault exists or whether there is another explanation for the root cause of the incident. In the event that a genuine software fault is identified, where possible 4th Line Support update the incident (recorded using the PinICL 'Fault Management' software) with details of the root cause of the incident and pass the call to the appropriate development team for correction and release of an updated version of the software. Where no software fault is found, the call is updated with an explanation of why the incident occurred so that this explanation can be passed on to the call originator to seek closure of the call.

Training is provided for POCL Post Masters and Counter Staff. The description of the training programme may be found in the ICL Pathway Training and User Awareness Baseline Document [21].

This document provides information concerning the systems used to support EPOSS at CI4. Specifically the following topics are covered:

- The EPOSS Domains
- The EPOSS Service
- EPOSS Concepts and Terminology
- Overview of the EPOSS System and Tools
- A step by step walk through of the service
- EPOSS Diagnostics and Errors
- Trouble Shooting EPOSS

FUJ00079425

1.1 Why EPOSS?

EPOSS provides the point of sale service for the outlet to conduct the selling of POCL products and the accounting for those sales to POCL. EPOSS provides the facilities for the selling of products over the counter. It provides the accounting facilities for the outlet to balance the outlet at the end of the week against those sales activities and provides the reporting facilities to produce the required and mandatory reports for the outlet and POCL.

In addition to these "In-Day" Services, applications are provided which fall within the service delivery of EPOSS for the initial migration of outlets to Horizon and for the service management of delineating actions by processing days, in other words the end of day service. The scope of the entire EPOSS Service Delivery defines these services within Domains, defined in the next section and amplified in the EPOSS High Level Design (Reference [3]).

EPOSS cannot be divorced totally from either the counter environment in which it operates (viz. Riposte) nor other concurrent Counter applications (viz. LFS, OBCS and APS), though the support of these products is not the subject of this document.

FUJ00079425

2 Scope

This document is one of a series of support guides intended to provide the essential information on specific parts of the Horizon system. It is aimed at staff with different levels of knowledge and experience of the Horizon systems management system. It therefore fulfils the role of a **training guide** and a **reference document**. It also has **cross references** to other documentation which contain generic Pathway related information, provide specific detail of the part of Horizon being described or are work instructions which contain detailed procedures.

Provided in this document are:

- A general description of the EPOSS processes to enable the reader to become familiar with the subject
- A further description of the EPOSS application, the components and an explanation of the terminology
- Some guidelines on potential problems and where diagnostic information may be found

More detailed information is available in the appendices. Included here are:

- Examples of EPOSS control documents
- Samples of the EPOSS tool dialogues, inputs and output.

From the outset it must be recognised that the term EPOSS is referred to in a number of contexts:

- As a point of sale service EPOSS is a software application providing that service on the counter, so termed the EPOSS In-Day Service
- From a Service Delivery perspective EPOSS is a set of services not limited only to the point of sale element, each of which is provided by a software application. Together these define what is termed the EPOSS Product, and is supported by a specific Domain
- As a development unit within the POCL Infrastructure Delivery Stream of the Pathway Programme the EPOSS Development Team are not only responsible for EPOSS the Point of Sale Service and the EPOSS Service Delivery Elements, but also a second counter application OBCS, or the OBCS product

Taking this view of EPOSS it must also be recognised that the point of sale service operated alongside other counter applications and within a common Riposte Service and that it cannot be divorced from the rest of the counter environment or applications. The contents of this guide are however limited to all things EPOSS though it is recognised that support of the product necessarily involves use of other elements of the counter environment.

These differing perspectives of EPOSS in the context of this support guide are best explained diagrammatically as follows.33

FUJ00079425

Figure 1 represents the Pathway Application Products for which the EPOSS Development Team is responsible. In this context the diagram represents the organisational scope of the support for the EPOSS Product and defines the product about which this guide is written. The relevance this has to the provision of support of the EPOSS Product will become apparent later, save to say that documentation produced in support of the EPOSS Development Team's activities may well not be limited to either OBCS or EPOSS.

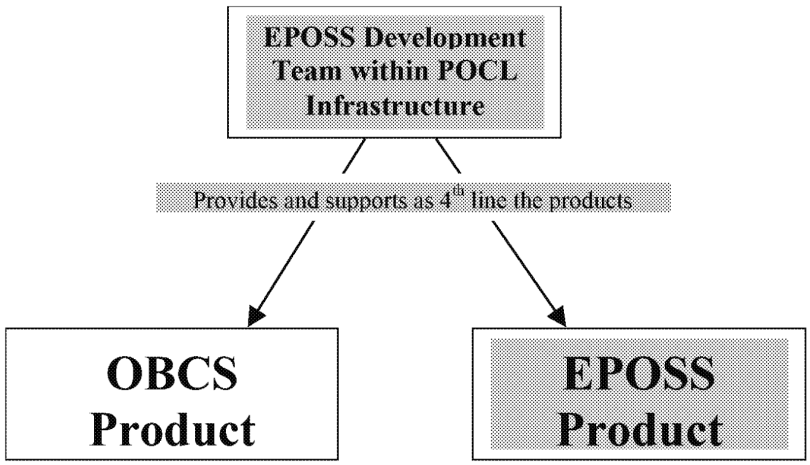


Figure 1 – Organisational Scope of the EPOSS Product

Figure 2 defines the EPOSS Product in terms of its domains. Not being limited to the point of sale service on the counter the EPOSS Development Team are responsible for EPOSS software within each of these domains. The diagram represents the domains as a Data Flow Diagram showing that the service provision within each domain is not independent of the other EPOSS domains. The content of Figure 2 is amplified later in Section 4 and exposes the design boundary of the EPOSS Product, but in the scope of service delivery support it defines the domains about which this document is written.

Figure 2 isolates the EPOSS point of sale service within the EPOSS Product and as a domain, so termed the In-Day Service, which is the main element of EPOSS. The In-Day Service is implemented by the EPOSS counter application. The main focus of this document is given to the EPOSS In-Day Service.

FUJ00079425

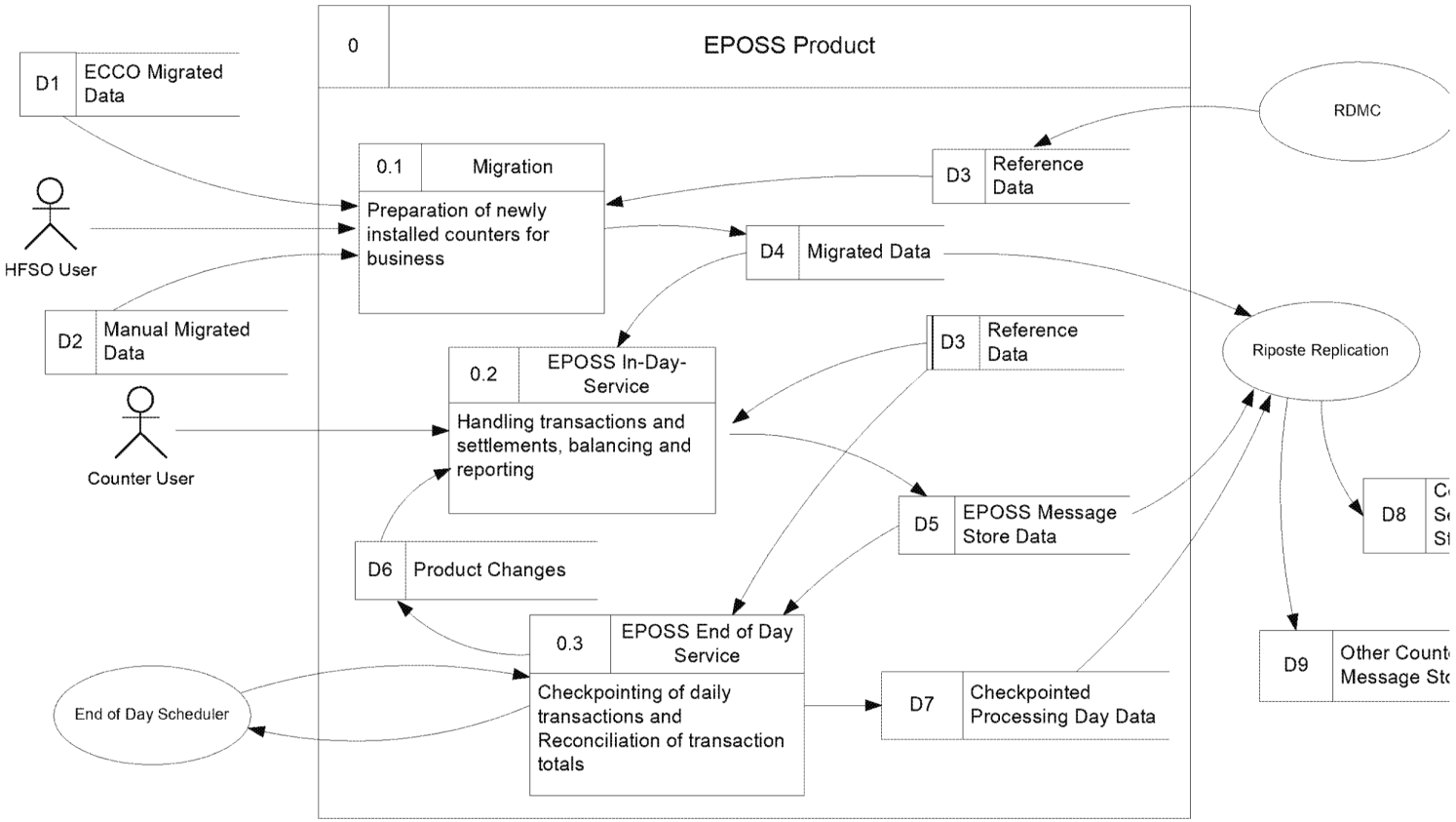


Figure 2 – EPOSS Product Domain Scope

FUJ00079425

2.1 Readership

It is assumed that the reader has a technical background and has some familiarity with the Horizon system especially the general features of the counter system, of Migration and End of Day Concepts.

Readers are also expected to have practical experience of the support tools such as Powerhelp, the SSC KEL, PVCS and PinICL.

Essential to all counter applications, and to the Migration is an understanding of Riposte. Readers are expected to have practical experience of the support tools provided by the Riposte Product.

Having been produced as an addendum to the EPOSS Design Documentation and the support view of the EPOSS Development Team this document has been produced from the perspective of 4th line support.

2.2 Documentation Roadmap

Essential to a complete understanding of the EPOSS product is the design documentation. The EPOSS Product Design Documentation is specified first at a High Level in terms of the Product, each Domain and the services provided within each Domain. The design is then defined at the Low Level by a specification for each software component that comprises the product and how that component provides support to implementing one or more services of the product.

In order to support the product as a whole the EPOSS Development Team provides other documentation that describe and define processes, procedures, definitions and designs for the team's terms and responsibilities in delivering EPOSS and OBCS.

As a consequence a complete EPOSS Development Team Product Documentation Set is available for reference and has been catalogued in a single reference, the EPOSS Development – Documentation Roadmap (see Reference [1]). The directly pertinent documents from this reference however are specified explicitly in Section 0.3 earlier and are referenced where appropriate throughout the narrative of the document.

As a compendium to the documentation roadmap the following diagram attempts to summarise the roadmap to all EPOSS Development Team Documentation.

FUJ00079425

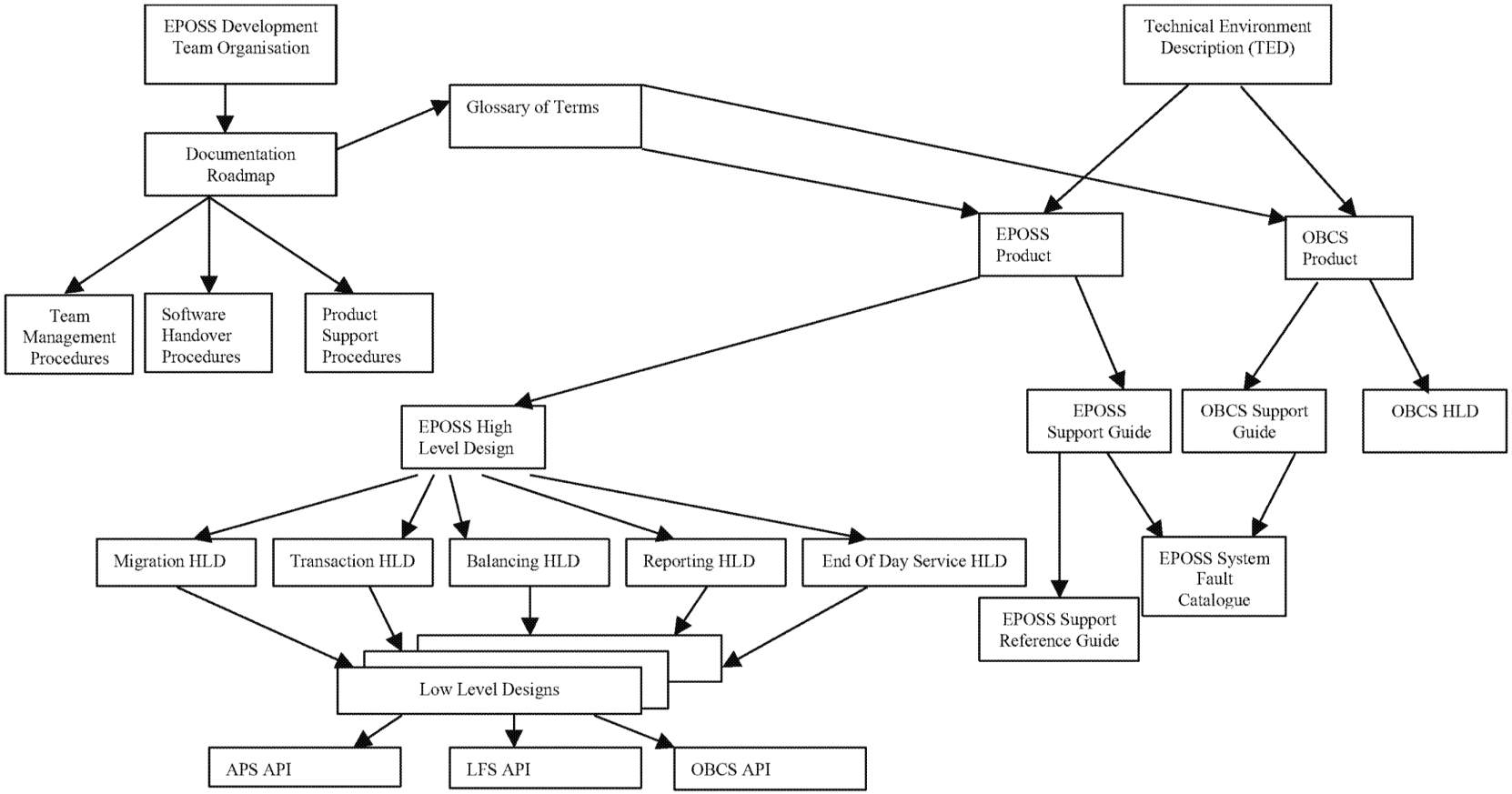


Figure 3 – EPOSS Documentation Roadmap Summary

FUJ00079425

3 Terminology

The terminology defining all aspects of the EPOSS Product is defined in the EPOSS Glossary of Terms (see Reference [2]).

FUJ00079425

4 Concepts

4.1 EPOSS Overview

The EPOSS Product provides the primary counter business functionality for transacting outlet products; it is the counter point of sale system. Incumbent in the product is software that delineates the day's processing at the outlet for the purposes of reconciling transactions posted against those later harvested to TIP.

Additionally the EPOSS Product provides the software to enable outlets to migrate to the Horizon system.

The EPOSS Product provides this functionality in three separate domains, each domain being supported by a different architectural framework as defined in the TED (see Reference [11]). The three domains are **Migration**, the **EPOSS In-Day Service** and the **End of Day Service**. The purpose of the EPOSS/Desktop overview is to walk the reader through all the operations of EPOSS, from the creation of stock units and users through to the rolling over of an office.

4.2 Stock Unit Types and Uses

4.2.1 Shared Stock Unit.

Shared stock units are stock units to which a number of users can be attached and can transact post office business at the same time. A user can only use one node, so the users will be attached to, and use, the stock unit from different nodes (*see diagram below*).

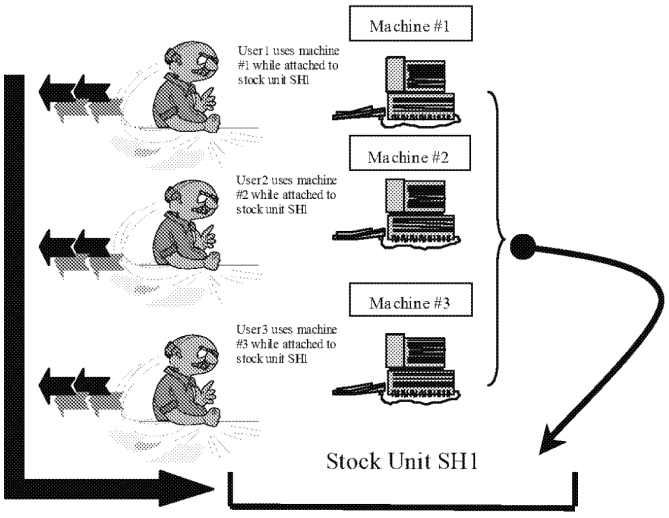


Figure 4 - Assignment of Users to Shared Stock Units

Any number of users can be attached to a shared stock unit at any one time, but only one of those users can be logged on to the system during balancing.

FUJ00079425

4.2.2 Individual Stock Units.

Individual stock units are for use by one person only at any one time (*see diagram below*).

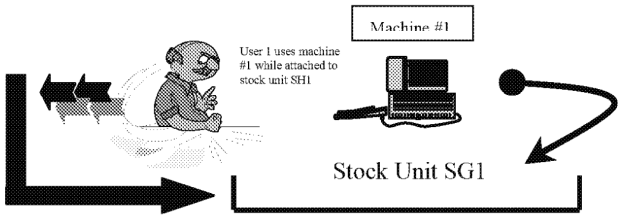


Figure 5 - Assignment of a User to an Individual Stock Unit

No other user can be attached to an individual stock unit until the current user is attached to a different stock unit.

4.2.2.1 Stock Unit Operation in relation to Cash Account Periods.

Stock units, and the office, work in periods of time called cash account periods (CAPs). CAPs represent a week, and there can be up to 53 CAPs in a year. At the end of each CAP (post office CAP run from Thursday to Wednesday), each stock unit will be balanced, and rolled over into the next CAP (e.g. CAP 1 to CAP 2). Once this is done, any transactions for the stock unit will be recorded in the next CAP.

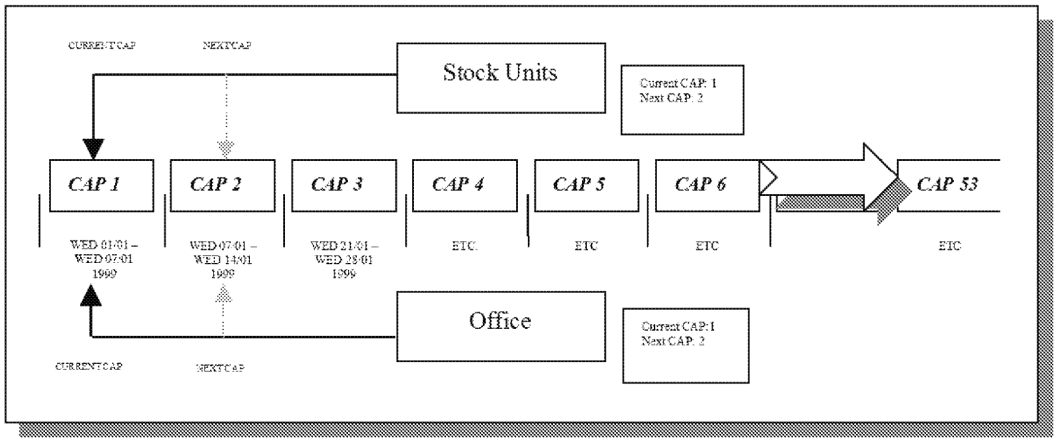


Figure 6- The Rollover of Stock Units into Cash Account Periods (CAPs)

The office also works in CAPs, and will be rolled over at the end of the week, but not until ***all*** the stock units have been rolled over first. This means that at any point in time a stock unit can be one CAP ahead of the office CAP.

FUJ00079425

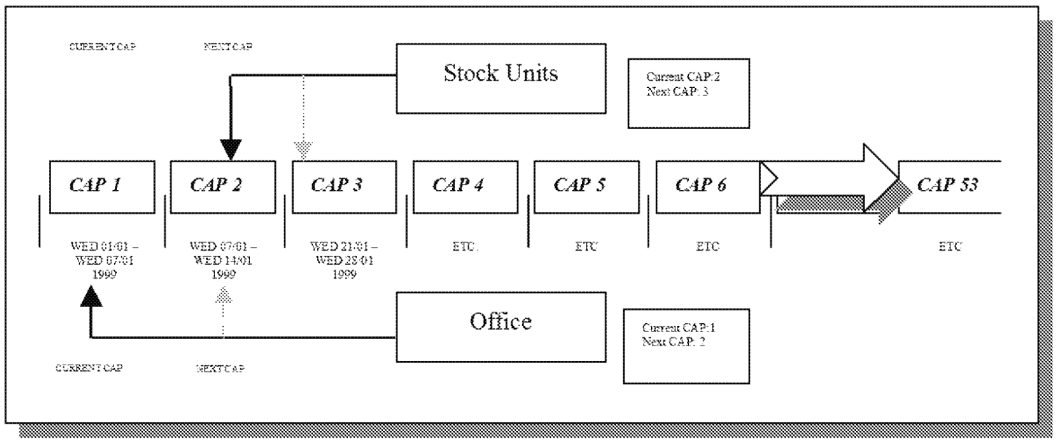


Figure 7 - The Relationship between Stock Unit CAP and Office CAP

4.2.3 Balance Periods.

Stock Units, in between CAP rollovers, can roll into new balance periods. There is no limit to how many BPs can be performed in a CAP. BP rollovers are normaly only used in branch offices when a new clerk takes over responsibility for a stock unit. It is common practice for individual stock units to be 'rotated' between clerks every 6/7 weeks in order to limit opportunities for fraud.

4.2.4 Stock Unit Creation

Stock units are represented as persistent objects in the message store. When created a set of markers and rollover trailers are written, this gives the stock unit a starting point in the message store (see below for an explanation of the rollover messages, and the stock unit persistent object).

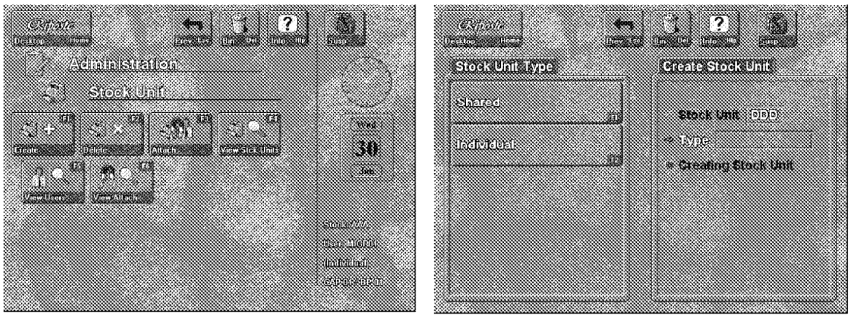


Figure 8 - Desktop Menus for the Administration of Stock Units

FUJ00079425

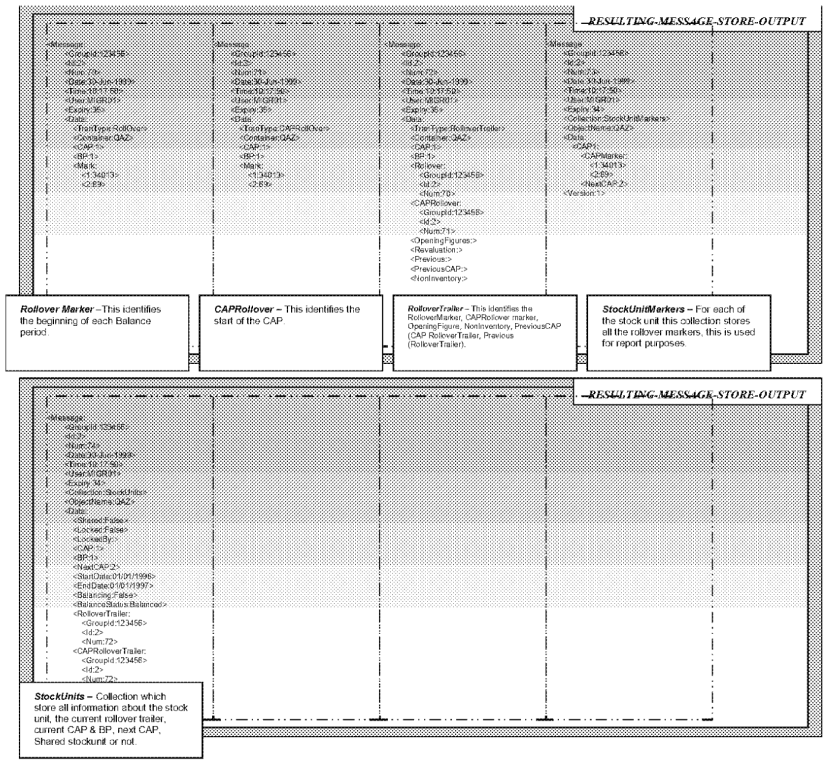


Figure 9 - Structure of Rollover Messages and the Stock Unit Persistent Object

4.2.5 User Creation

User creation is involves creating a user and assigning it to a group. Groups give the user rights to different parts of the system, for example the manager group has full access to the system, where as a Clerk can only use certain functions within the system.

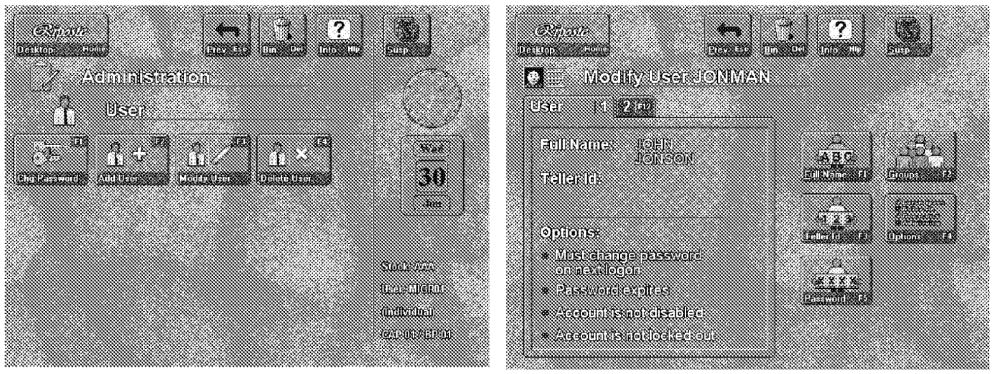


Figure 10 - Menus for the Administration of Counter Users

Below are example output messages for creating a user.

FUJ00079425

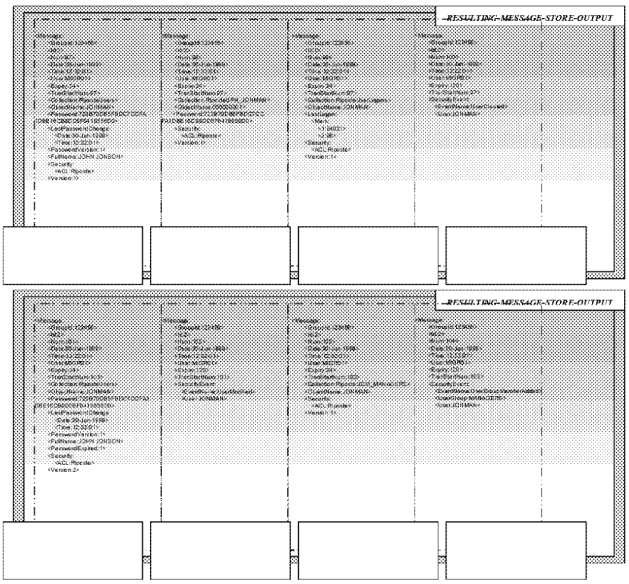


Figure 11 - Message Structures Associated with the Creation of Counter Users

FUJ00079425

4.2.6 Stock Unit Attachment.

Each user must be attached to a stock unit to use the system. On first entry to the system the user is automatically attached to the default stock unit ('DEF'), and from there the user can attach to a stock unit (see below for attachment message output).

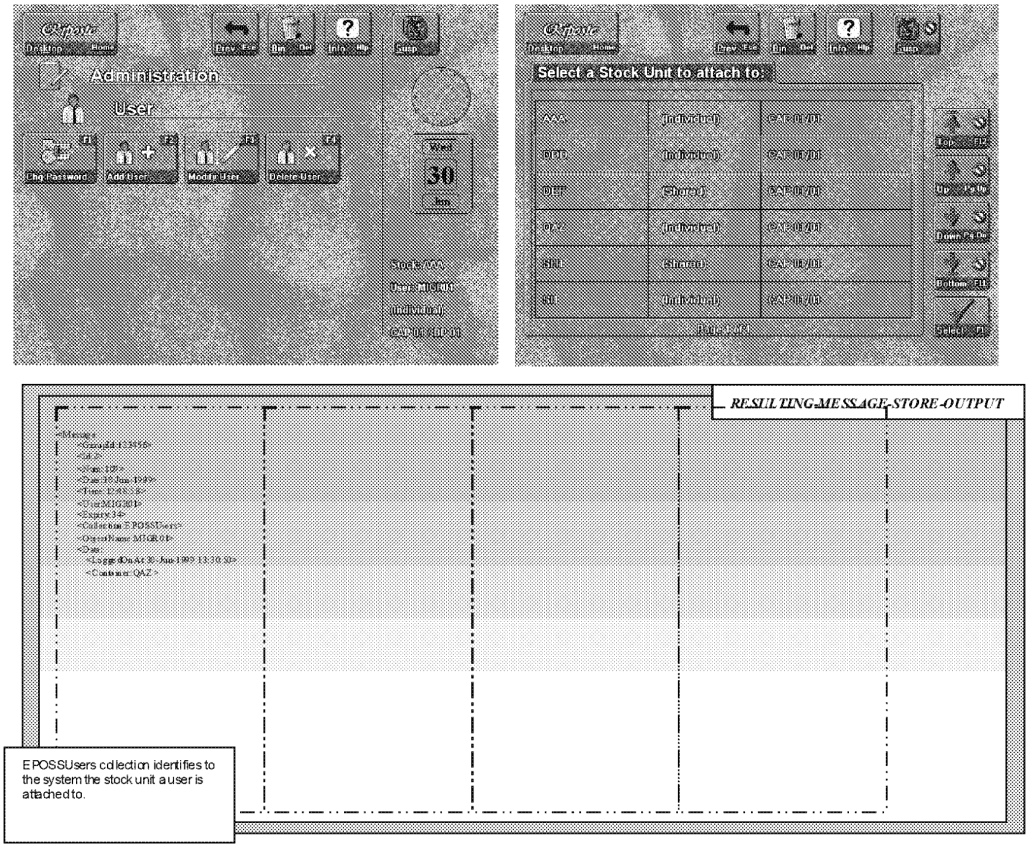


Figure 12 - The Attachment of Stock Units to Counter Users

4.3 EPOSS Operational Overview

This section will provide the reader with an understanding of the Desktop layout, transactions, messages, and the EPOSS system (this is from the perspective of a new office so also includes migration).

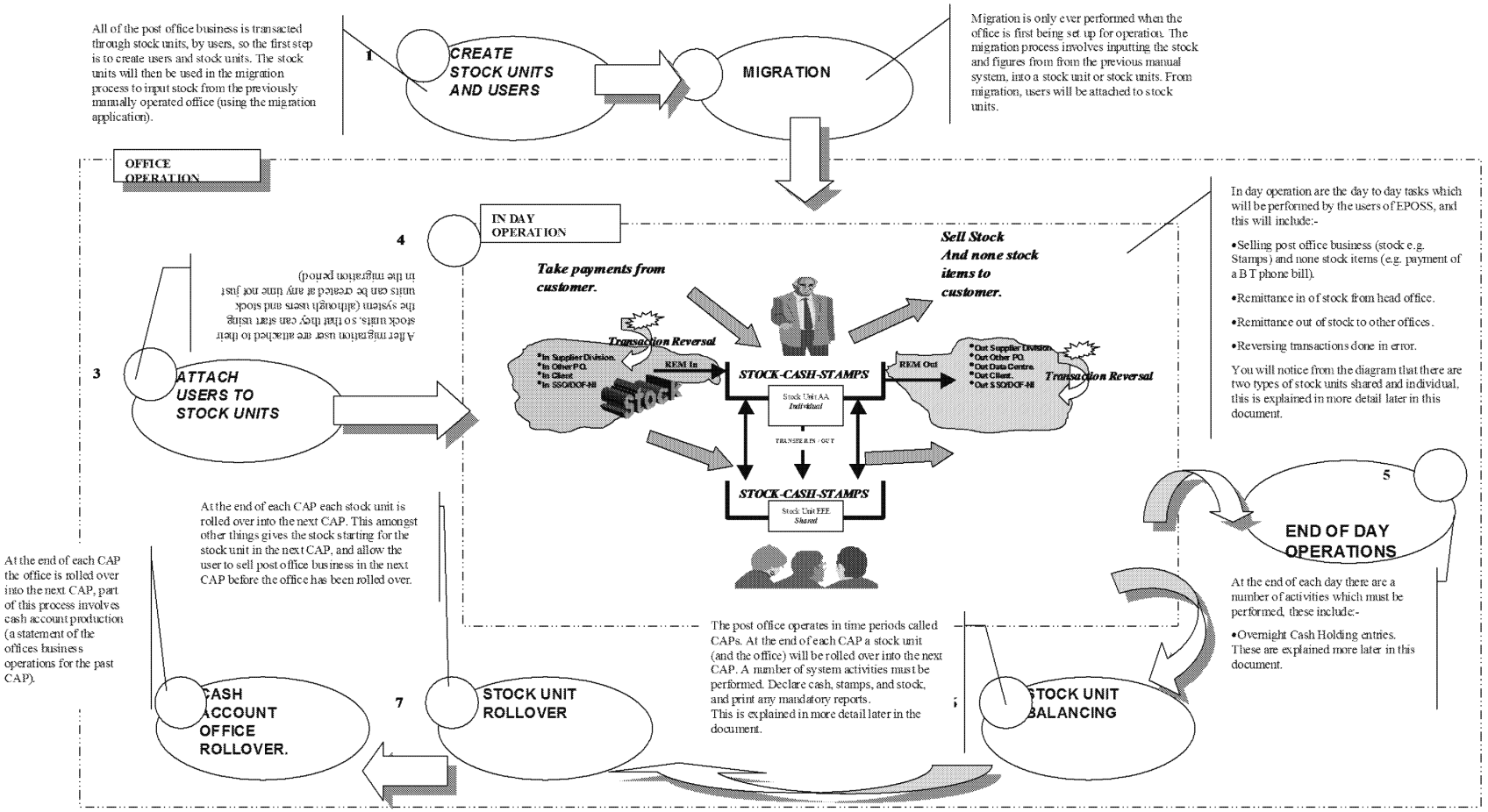


Figure 13 - The Operational Overview of EPOSS

FUJ00079425

4.4 Desktop Layout and Operation.

The diagram below shows the EPOSS Desktop.

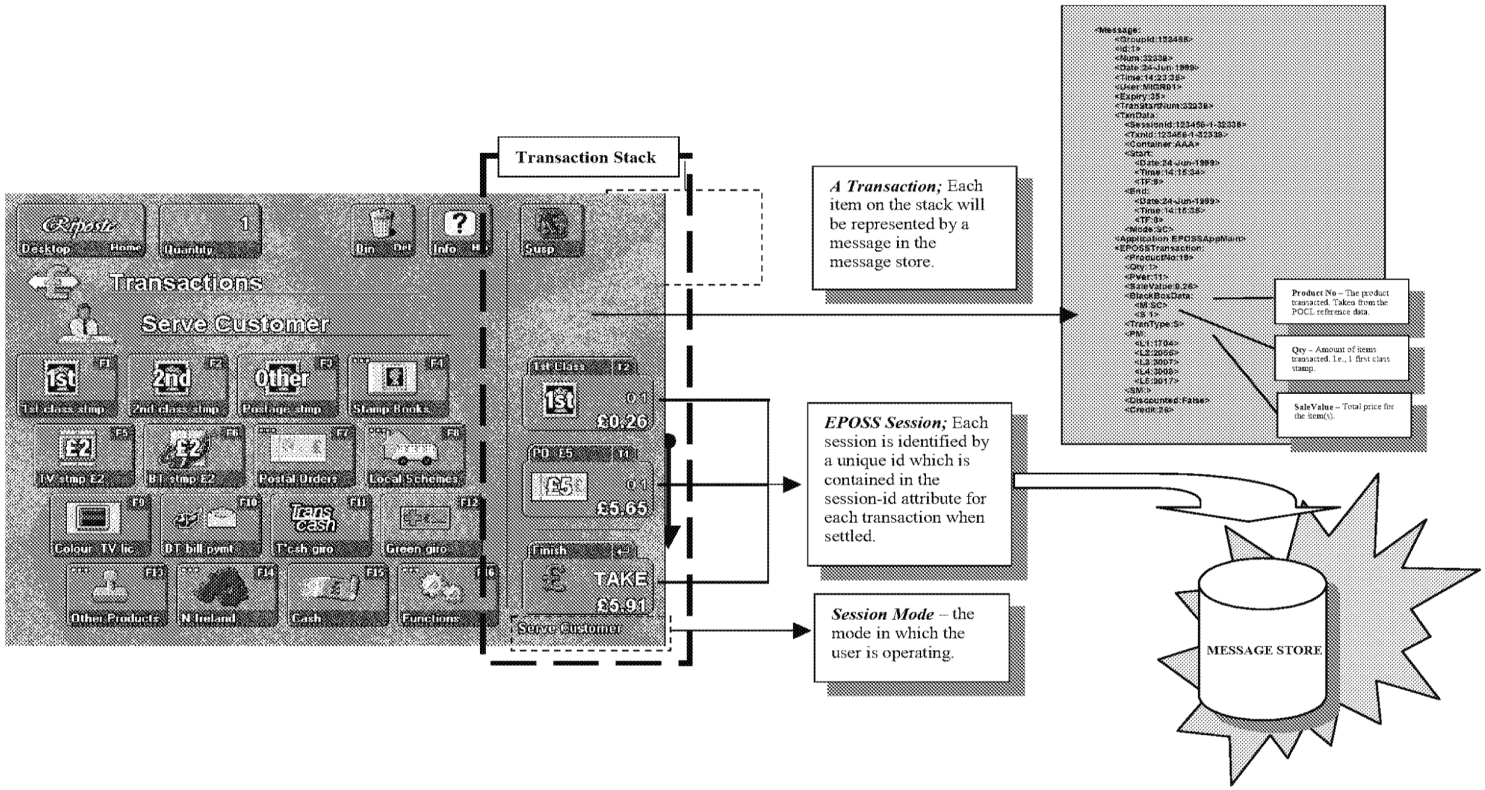


Figure 14 - A Sample Menu from the EPOSS Desktop

FUJ00079425

4.5 Migration

The Migration Service which operates within the Migration Domain provides the functionality to enable outlets to carry forward their operating business figures from their existing system of operation to the Horizon system. Two forms of migration are offered by the Pathway solution. An ECCO **Migration** allows outlets operating POCL's existing counter computer system ECCO, to be migrated through electronic capture of their existing figures and transmission into Horizon. **Manual Migration** provides a manual data capture mechanism on the counter for the outlet to submit their existing figures from whatever other form they currently operate.

In migrating figures from the outlets' existing system there are two primary phases of action, **Stock Unit Migration** and **Office Migration**. These phases are required to be undertaken whether undertaking an ECCO or Manual migration.

4.5.1 Stock Unit Migration

Outlet business is conducted with reference to a Stock Unit. A stock unit can be deemed to be a 'bucket' from which products are sold and into which payments are received. Often a stock unit relates to a physical pedestal stocked with products for sale and cash, from which and into which the products are taken for transacting and into which payments are made or taken. This bucket can then move around with the outlet user of the system (an Individual Stock Unit) or can be shared with other users. Often there is a one to one correspondence with the counters in the outlet. This may be represented diagrammatically (see Figure 4).

Migration requires that figures of each stock unit's holdings are migrated from the outlet's existing system. Hence there is a one to many migration of stock units.

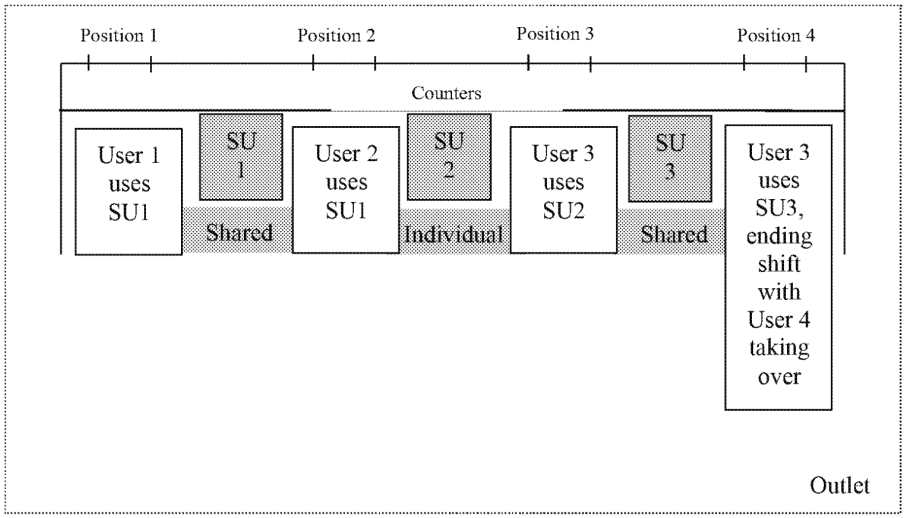


Figure 15 – Outlet Counter Position/Stock Unit/User Scenario

FUJ00079425

4.5.2 Office Migration

Whilst day to day business of the outlet is conducted via products transacted against a stock unit the office may also hold figures. These figures are termed suspense account items, or in other words items not attributable to any one stock unit. An example may be losses resulting from a robbery.

In migrating an outlet not only must all the stock unit figures be migrated but the office figures must also be migrated.

In migrating the entire set of figures from the stock units and the office it is incumbent on the postmaster to declare that the figures are correct. The figures can be checked by the migration service by asking for the net office balance position being brought forward from the outlet's existing system to be entered against which a check is performed.

Completing migration the office and stock units are rolled into Cash Account Period against which subsequent accounting within the Horizon system will continue.

4.5.3 ECCO Migration

ECCO Migration is provided for by the MiEcco EPOSS Application which resides on a Laptop provided to HFSOs for the migration of ECCO Offices. The Laptop is connected through the outlet Gateway counter PC.

The MiEcco application supports the migration of Stock Units from the outlet's existing ECCO system. It does not support Office Migration of the outlet. The data is provided on floppy disk to the Laptop from the resident MiEcco application which generates text files for transmission to the Migration Agent Server (MAS).

The MAS sits outside of the EPOSS domain. Agent software resident on the MAS is responsible for transcribing and routing the migrated data from the Laptop to the Correspondence Server supporting the outlet, for onward replication to the outlet counter message stores.

4.5.4 Manual Migration

Manual Migration is provided for by the MiMan EPOSS Application which forms part of the Counter Desktop.

The MiMan application supports the migration of stock units from outlets using either a manual system or any other computer system than ECCO. It provides a manual data capture mechanism for the entry of figures.

Additionally the MiMan Application supports the migration of Office figures either for an ECCO outlet or a manual outlet.

Use of the application completes the migration activity for the outlet by checking that the stock unit and office reconcile with the declared office Brought Forward position provided by the postmaster, rolling the stock units and office into the supplied CAP under which business is to continue in Horizon.

The migration concepts and processes are represented diagrammatically in Figure 5.

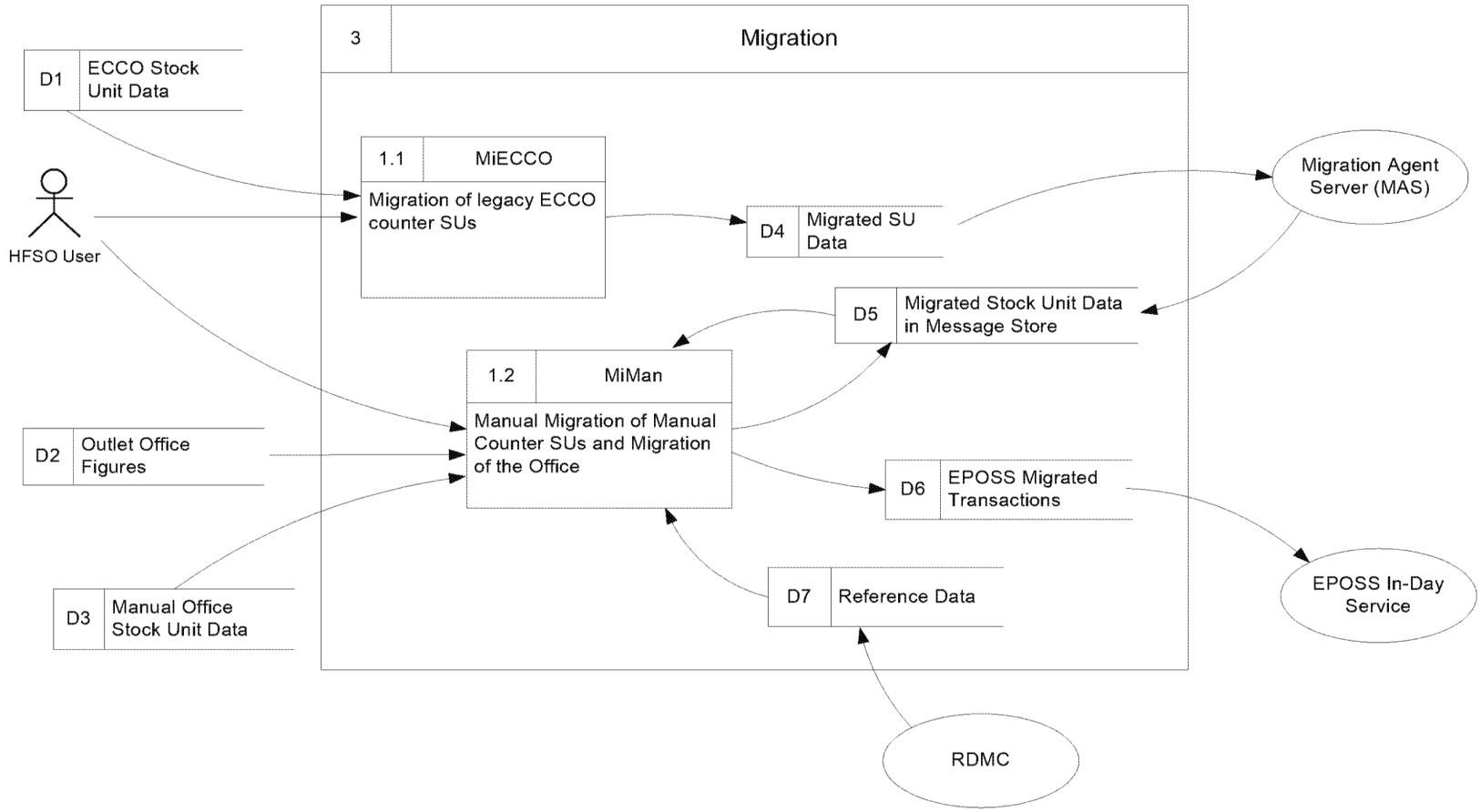


Figure 16 – Migration Service Domain

FUJ00079425

4.6 EPOSS In-Day Service

The EPOSS In-Day Service 'is EPOSS'. It is the point of sale counter application and runs within the Riposte Desktop Infrastructure. The EPOSS In-Day Service is also known as the EPOSS Counter Vertical Application, operating independently of other desktop applications such as LFS and APS, any interactions being served only through defined APIs and being supported by a common desktop and menu hierarchy.

The service provided by the EPOSS In-Day Service provides the business functionality to support the outlet point of sale service in undertaking the sale of POCL's product range, accounting for the sales at the end of week and servicing the provision of various reports for outlet and POCL use.

The EPOSS In-Day Service domain is logically provided for though four underlying services each providing or supporting some business functionality. This is represented in Figure 6.

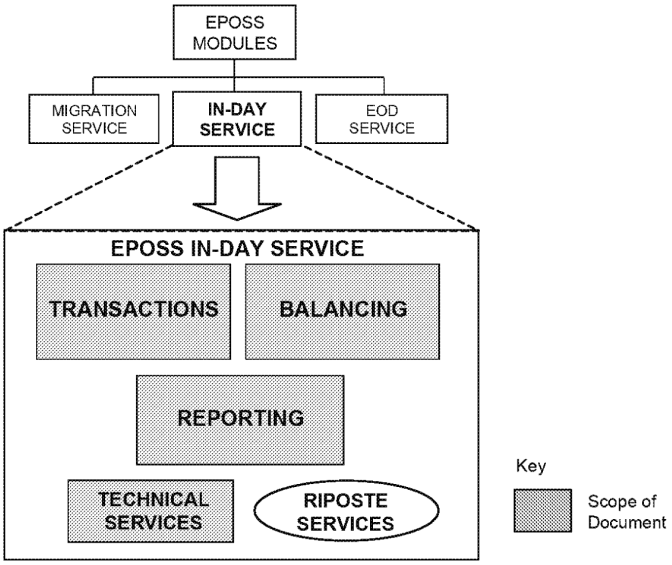


Figure 17 – EPOSS In-Day Service Domain

4.6.1 Transactions Service

The EPOSS Transactions Service supports the actual point of sale system to transact and settle products across the counter. It additionally supports the determination of desktop modes to be applied in navigating the counter desktop menus and the navigation to previous and the top level menus. It does not support the navigation around menus outside of this domain, which is handled by Riposte itself.

The transaction service provides facilities to transact products in different ways, for example via a desktop button, PLU or picklist, but it also provides an interface for other applications to initiate a transaction and settlement.

FUJ00079425

4.6.2 Balancing Service

Whilst the Transaction Service supports the provision for transacting products the outlet must also account for their sales at the end of the week (or cash account period). The balancing service supports the actions on the outlet to account for their business operation during the week.

4.6.2.1 The Accounting Node Hierarchy.

The accounting node hierarchy is supplied by Pathway and agreed with POCL and is made up of a selection of nodes which are used to accumulate product transactions in each of the different modes (e.g. Serve Customer, Remittance etc.).

Each Product when transacted in serve customer mode is given a set of primary mappings, and every transaction in other modes are transacted with Primary Mappings ***and*** Secondary Mappings.

These mappings tell the system where on the node hierarchy to accumulate each transaction.

FUJ00079425

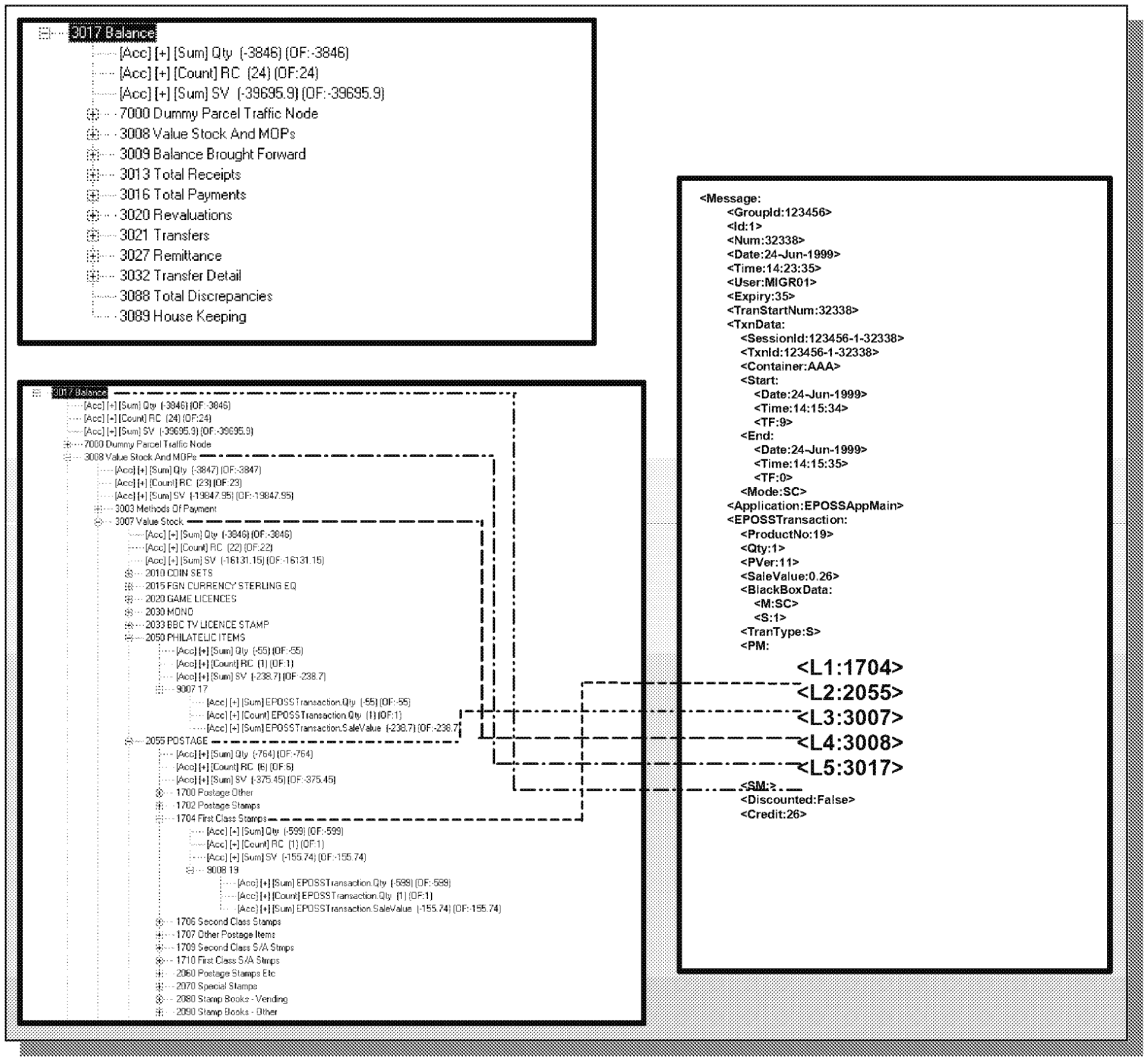


Figure 18 - Principles of the EPOSS Accounting Node Hierarchy

4.6.3 Reporting Service

In support of the Transactions and Balancing services the Reporting Service provides the facilities to produce defined reports as part of those processes or on demand.

4.6.4 Technical Services

Whilst the three previous services support the business functionality of EPOSS the implementation of the system has provided a number of applications which together form a set of technical services, merely servicing needs of the implementation and establishing common processes to service similar business needs.

4.6.5 Riposte Services

Whilst residing outside the EPOSS domain the dependency of the EPOSS application on the Riposte Infrastructure is paramount. Accesses from the EPOSS In-Day services

FUJ00079425

to the Riposte Message Store and the desktop are all initiated through APIs provided by Riposte.

4.7 EPOSS End Of Day Service

EPOSS provides End of Day functionality to the counter End of Day Service. In this context the end of day functionality of EPOSS is bounded by its end of day service domain.

The EPOSS End of Day Service domain provides the following functionality within the overall Counter End of Day Service:

- **Product Evaluation** of any Price changes in readiness for the next In Day service day
- **Daily Reconciliation** of the day's transactions
- Optionally **Weekly Reconciliation** of the week's cash account if the cash account has been rolled that day

The End of Day service provision provided by EPOSS is represented diagrammatically by Figure 7 and supports three separate applications with access to the Riposte Infrastructure service but operating outside of the Riposte Desktop. The End of Day Process is documented in detail in [7].

4.7.1 Product Evaluation

At the end of each processing day, identified by the Counter End of Day Service and outside of the EPOSS domain any reference data changes to product pricing will be evaluated and a set of defined revaluations notified to the next EPOSS In-Day Service if appropriate.

4.7.2 Daily Reconciliation

At the end of each processing day the Counter End of Day Service, outside of the EPOSS domain, will delineate the transactions for the day. The transactions bounded by the End of Day Markers will be reconciled. The EPOSS Daily reconciliation will perform two phases of reconciliation, firstly validating transactions and counting the number of transactions 'entered' that day, aggregating the total quantity of product sales for the day and aggregating the total sales value for the day. These totals will subsequently be reconciled by TPS against the transactions actually harvested for the day (and should be the same). Secondly the Daily Reconciliation will aggregate the day's transactions into a 'mini cash account', forming totals of the transactions as they would have appeared had they contributed to the Cash Account. Any errors will be recorded for later reporting.

The delineation of the processing day is required as the EPOSS In-Day Service may continue to operate into the evening and transactions for the day must sooner or later be cut off.Weekly Reconciliation

Weekly Reconciliation is only performed when on the same processing day as is being ended the office is rolled over. The reconciliation performed is to take all those mini cash accounts for the cash account as has just been rolled and compare the figures

FUJ00079425

produced against those figures produced for the actual cash account. Any errors will be recorded for later reporting.

A dependency exists on any processing day between the daily reconciliation having been completed before the weekly reconciliation commences.

FUJ00079425

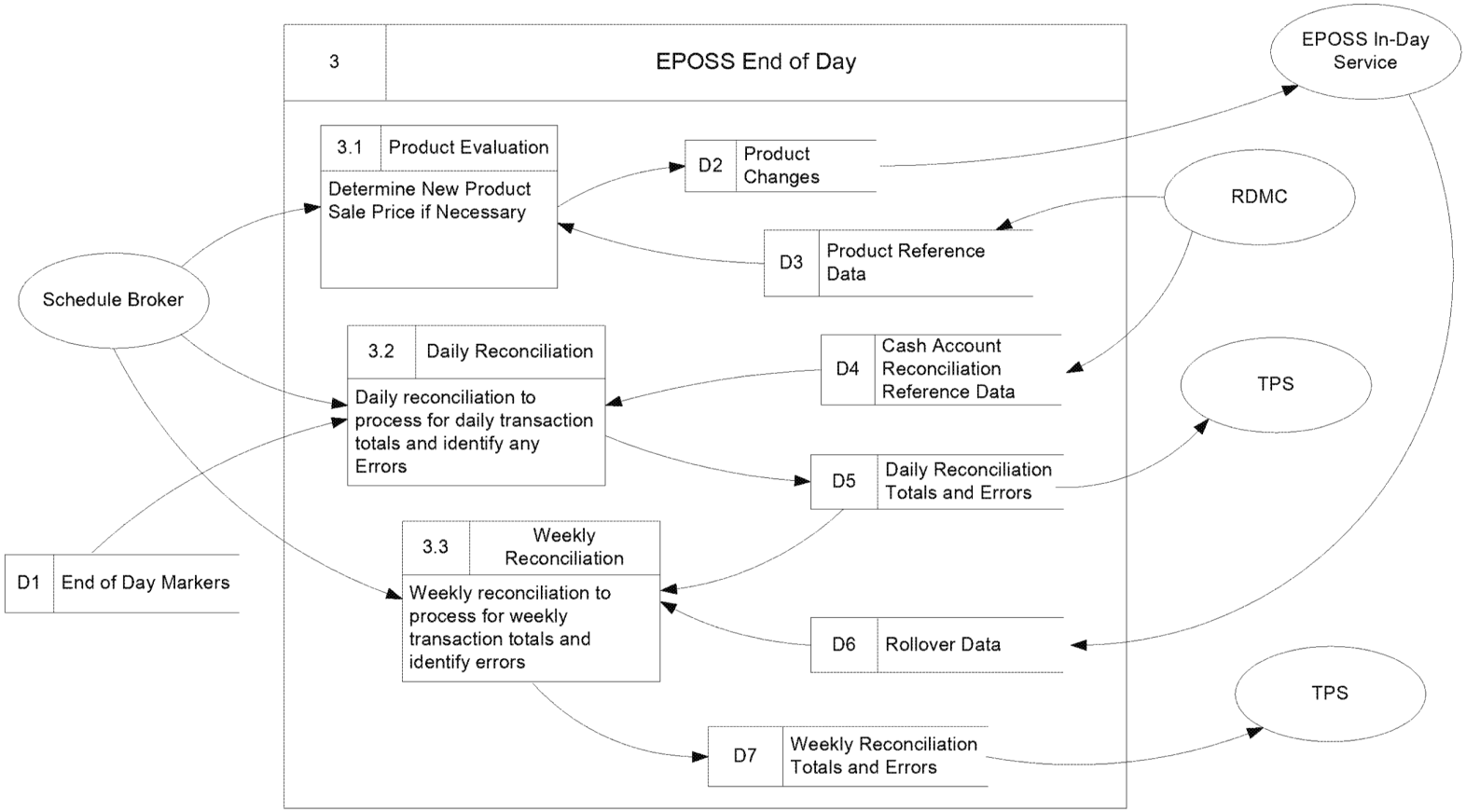


Figure 19 – EPOSS End of Day Service Domain

FUJ00079425

4.8 Coding Standards Used

Naming conventions should follow the guidelines defined in the coding standards [18]. Any deviations from the standard naming conventions will be highlighted in the low-level design documentation.

4.9 Platform Design

The EPOSS Application product comprises a number of component executables that operate in conjunction with a particular Riposte release and runs within the Windows NT platform using Internet Explorer. Therefore, there is dependency on the Riposte builds, Windows NT and Internet Explorer versions and service pack releases. The data relating to the versions and service pack releases may be found in the Physical Design for Counter at CSR+ [19].

5 Process Description

5.1 Migration

The migration of an outlet to the Pathway solution is undertaken as a one-off exercise.

An ECCO outlet is migrated in a two-stage operation, migrating their stock units first using the MiEcco Migration tools supported on the ECCO Migration Laptop, followed by migration of the office using the manual migration application.

A manual outlet is migrated again in a two-stage operation, migrating their stock units first using the MiMan Migration application on the counter desktop, followed by migration of the office using the same application.

5.1.1 ECCO Stock Unit Migration

The migration of the ECCO Stock Units is conducted using the MiEcco Migration Tools implemented on the ECCO migration Laptop. The exercise is undertaken by the HFSO, connecting the laptop to the gateway PC through to the MAS. The resultant migrated stock unit data is replicated automatically to the counter Riposte Message Store using Agents software.

5.1.2 Manual Outlet Stock Unit Migration

The migration of a manual outlet's stock units is conducted using the MiMan Counter Desktop Application run within the counter service. The exercise is undertaken by the HFSO and relies on the counter desktop service being available.

5.1.3 Office Migration

On completion of the stock unit migration for an outlet office migration is conducted using the MiMan Counter Desktop Application run within the counter service. The exercise is undertaken by the HFSO and relies on the counter desktop service being available.

FUJ00079425

On completion of migration the Office and stock units are migrated into Balance Period 2 of the chosen CAP.

5.2 In-Day-Service

The EPOSS In-Day Service provides the applications to the outlet users for undertaking day to day point of sale transactions and accounting for the outlet's balance position at the end of each chosen Balance Period and Cash Account Week.

The EPOSS In-Day Service runs in the Counter Desktop Service, which is reloaded at around 0300 each day. Transactions written to the counter message store are replicated automatically by Riposte to other counters and the correspondence server message stores.End of Day Service.

The EPOSS End of Day applications act as a service to delineate the day's processing. The applications are invoked by the End of Day Scheduler, and application itself invoked at the start of each day when the Riposte Desktop is brought up.

The scheduler, whilst active through the processing day, kicks off end of day actions automatically at or around the office closing for the day, in accordance with rules defined in [17].

There are three processes invoked:

1. **Product Evaluation** determines from Reference Data if any product price changes are due in the next three days. A result any changes are prepared for notification to the next end of day service.
2. **Daily Reconciliation** is performed only when the transactions for the day have been delineated. It sums details of transactions for the day for later reconciliation reporting by TPS.
3. **Weekly Reconciliation** is performed only when an office has rolled over and the daily reconciliation for that day is complete. It sums details of transactions contributing to the cash account for that period rolled for later reconciliation reporting by TPS.

FUJ00079425

6 System Design and Tools

The tools provided for the automatic and manual management of the EPOSS Product are limited to tools provided for the Migration of ECCO outlets residing on the ECCO Migration Laptop, the Pathway Service Management Software and the Riposte Infrastructure Product servicing all counter applications.

All but the tools used on the Migration Laptop lie outside the scope of this document, but see appropriate documentation provided in the next section. The tools provided for the migration of outlets are described in the Horizon Field Support Officer Guide Release 2 Section 5 (ECCO Outlet Procedures) – see Reference 14.

FUJ00079425

7 Step by Step Guide

7.1 ECCO Migration

The systems management of the migration of ECCO Stock Units is described in the Horizon Field Support Officer Guide Release 2 Section 5 (ECCO Outlet Procedures) – see Reference 14.

7.2 Manual Migration

The systems management of the migration of Manual Outlet Stock Units and the subsequent Office Migration of both ECCO and Manual outlets is described in the Horizon Field Support Officer Guide Release 2 Section 4 (Manual Outlet Procedures) – see Reference 13.

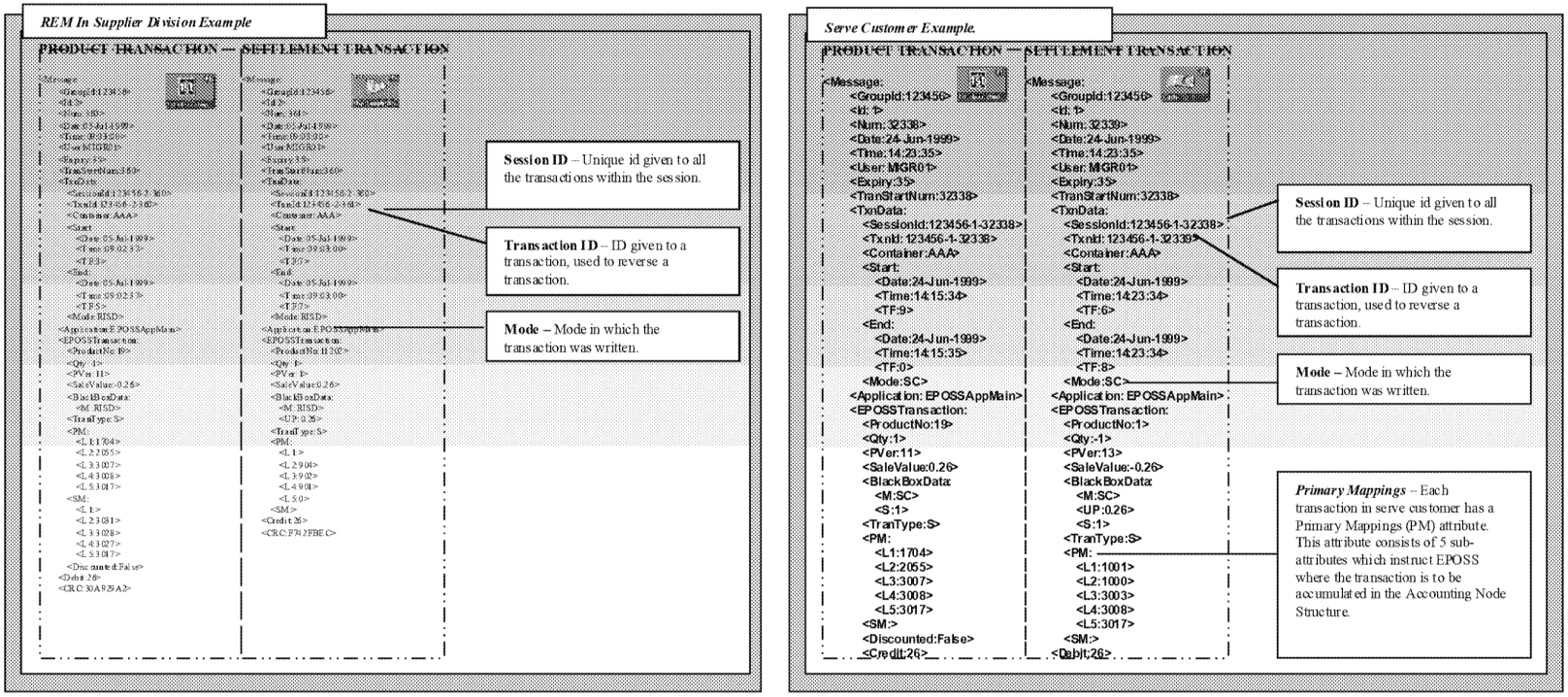
Manual Migration operates as a counter desktop application. In this context systems management is also undertaken as described in the Counter Management Support Guide – see Reference 12. The reference describes the systems management of the counter as a whole in which Manual Migration is but one application.

7.3 EPOSS In-Day Service

The systems management of the Pathway Counter Services is described in the Counter Management Support Guide – see Reference 12. The reference describes the systems management of the counter as a whole in which EPOSS is but one application.

Remitances are part of the In-Day-Service, where Cash and Stock remittances represent the movement of cash and stock items between an outlet and one of a number of external sources/destinations.

FUJ00079425



The above example is for a first class stamp transacted in REM In mode and also in Serve customer mode. Notice that the product is transacted negatively in REM In mode and Negatively in Serve Customer mode.

Figure 20 - Sample Transaction Messages for Remittance in and Serve Customer Sessions

FUJ00079425

7.3.1 REM In Stock.

Before selling stock, stock has to be present in the stock unit. User simply selects stock from the menu and settles when complete. These products will then become part of the stock units holding figures.

Each stock item, e.g., First class stamp, is held as a negative quantity in the messagestore. Every REM Transaction has secondary mappings which identifies the transaction as a REMed transaction.

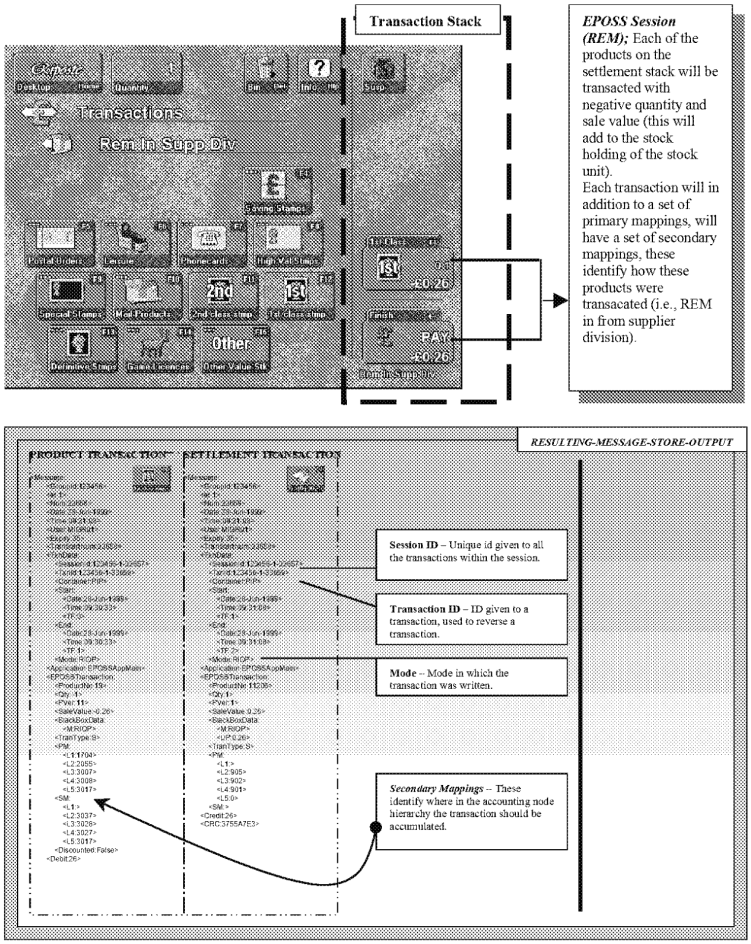
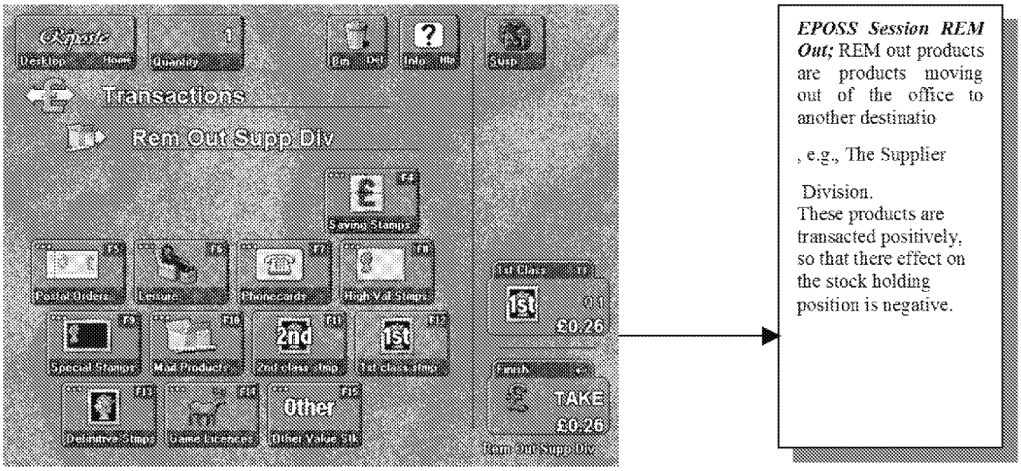


Figure 21 - The Menu and Message Structures for Remittance in Transactions

FUJ00079425

7.3.2 REM Out Stock.

REM out stock is the movement of stock from this office to another, or the supplier division etc.



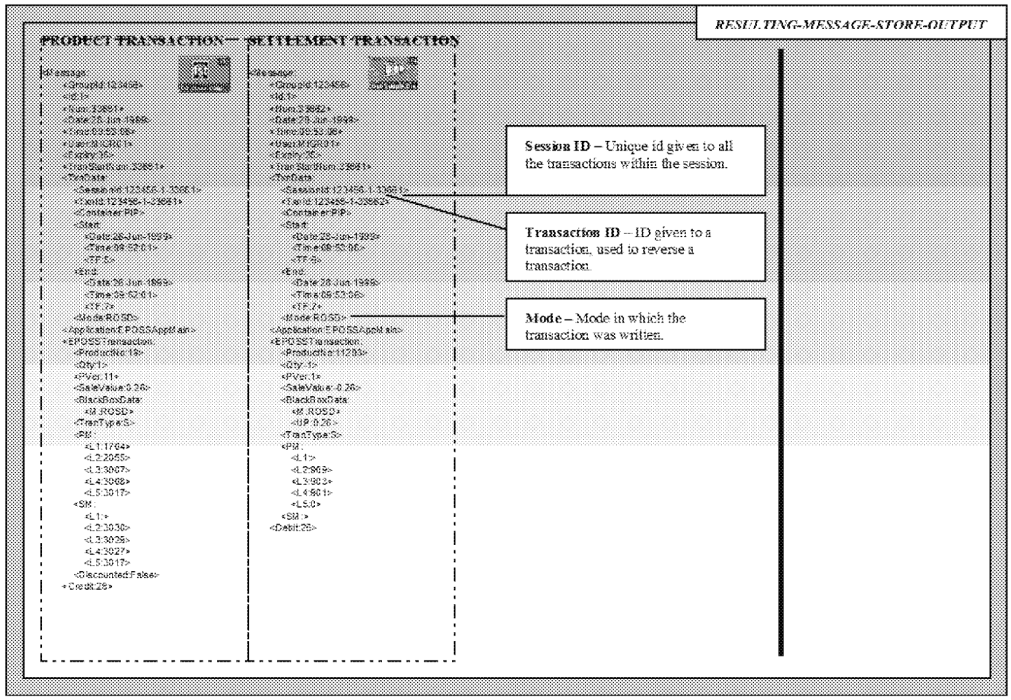


Figure 22 - The Menu and Message Structures for Remittance Out Transactions

7.3.3 Selling Stock.

When a serve customer session is settled each of the transactions are settled depending on the product type (Payments, Receipts, Stock). E.g.:-

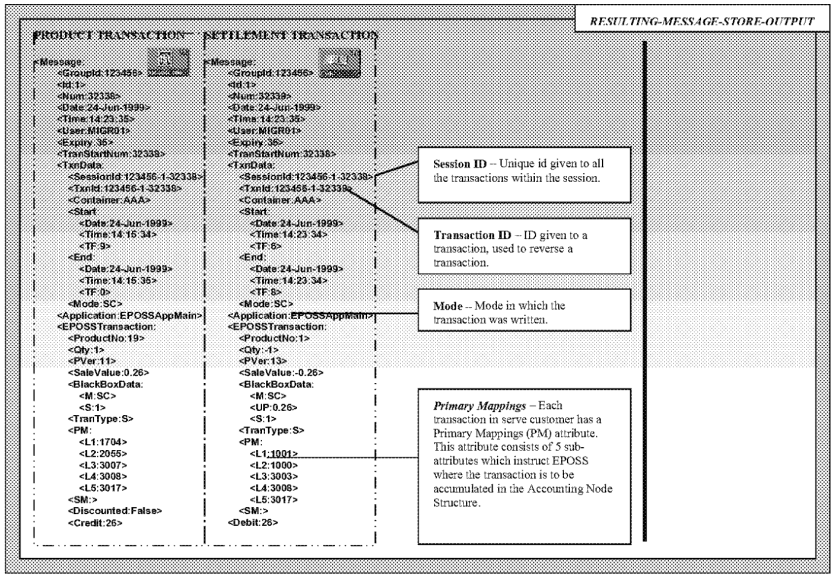
FUJ00079425

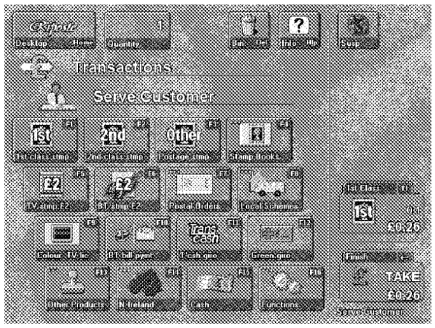
Stock – Is written positively (as stock is stored negatively this will decrease stock).

Payments – i.e., A BES payment is written negatively (As cash is going out).

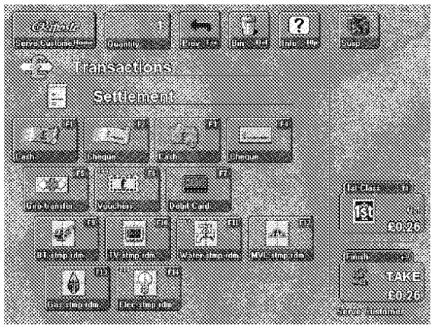
Receipts – i.e., Bill payments are written positively (as cash is coming in, and increasing your cash holding value).

The settlement transaction i.e., 'cash', is written with the same amount in cash as the session total but with the opposite sign.





- Serve Customer Session – Sale Of a First Class Stamp.



- Serve Customer Session – Settlement Menu, where user can select settlement type, e.g., Cash.

Figure 23 - Menu and Message Structures for Settlement of Transactions

FUJ00079425

7.3.4 Transaction Reversal.

Transaction reversal enables the user to reverse transactions. E.g., If a stamp has been sold in error then the transaction id for the stamp transaction can be entered in the transaction reversal application, and the transaction will appear on the transaction stack and the user will be prompted to enter a settlement product, i.e., cash, and the transaction will be reversed with a printed reversal report. (see screens below).

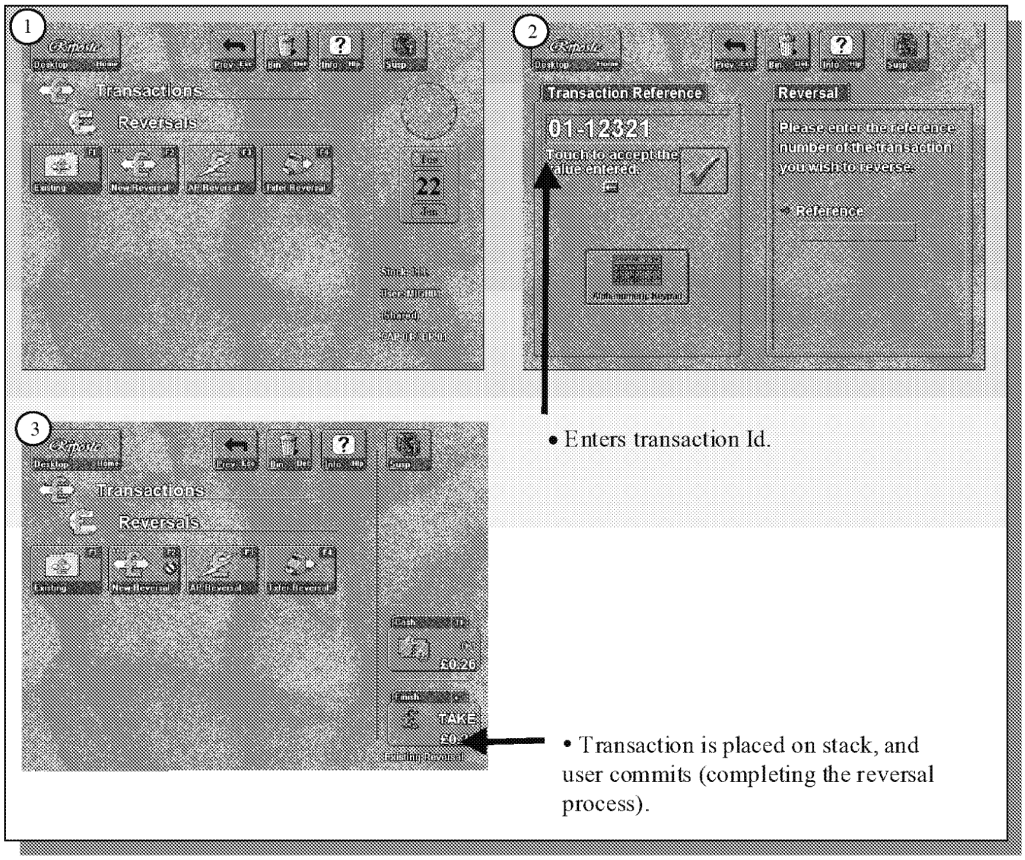


Figure 24 - The Reversal of Transactions

FUJ00079425

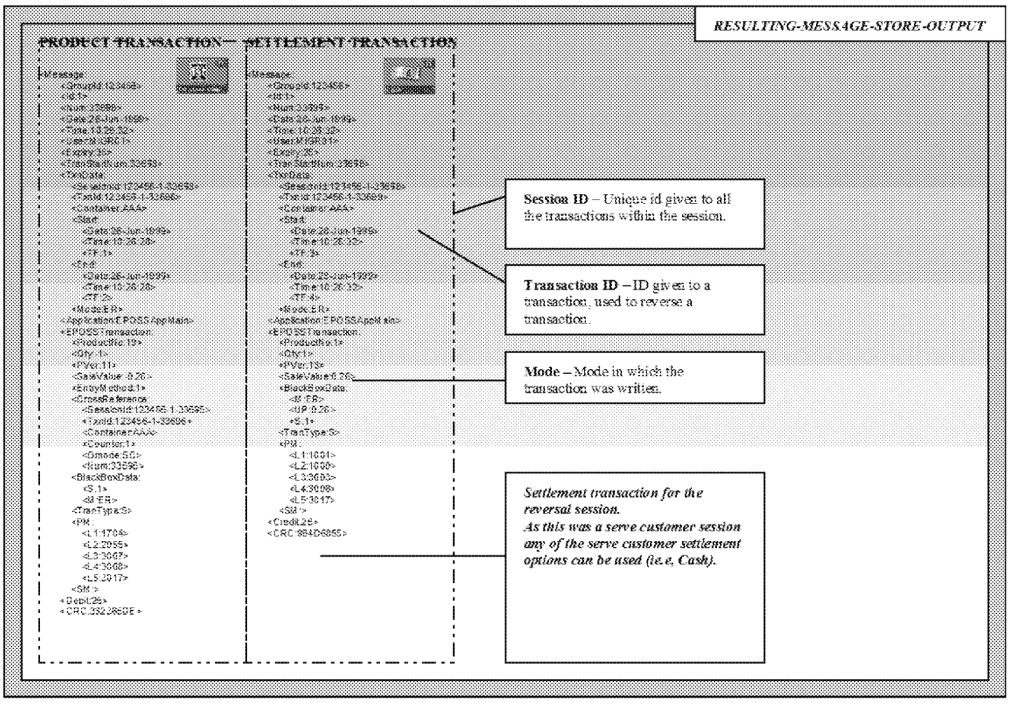


Figure 25 - Structure of the Transaction Message Showing the Transaction ID Used for Reversals

7.3.4.1 Obtaining the Transaction ID

The transaction log will provide all transactions based on criteria. E.g., to obtain the information required to reverse a transaction in CAP 1 for the stock unit "AAA", the following criteria would be entered in the transaction log report criteria screen.

a) Stock Unit set to "AAA".

b) CAP set to 1.

This will list all the transactions in CAP 1 for the stock unit "AAA".

FUJ00079425

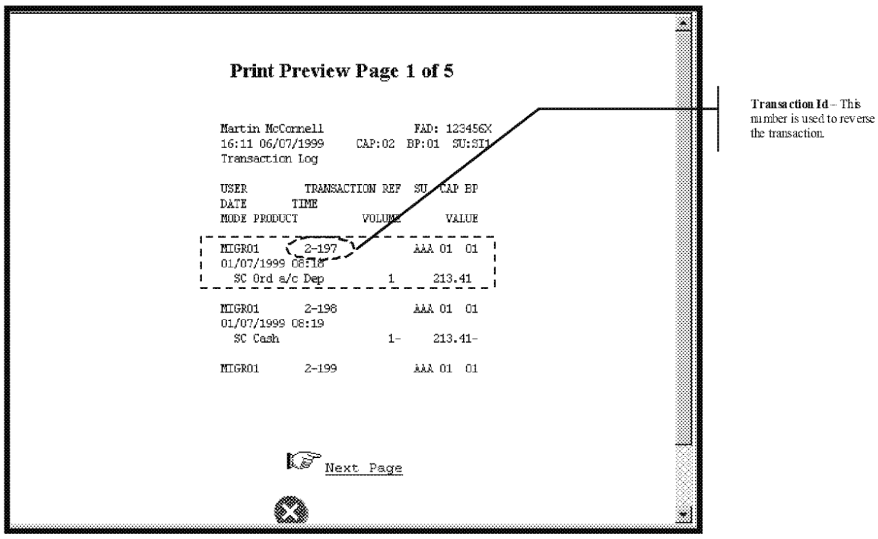


Figure 26 - Location of the Transaction ID to for Reversal of a Transaction

The transaction ID is then entered into the transaction reversal screen and the transaction can then be reversed.

7.3.5 Transfering (In/Out) Stock.

Transfers are used to transfer stock from one stock unit to another, and is a two phase process :-

1. Select the stock unit to transfer out to, and select the stock required to be transferred.
2. The above will then be transferred in using the receiving stock unit.

FUJ00079425

7.3.5.1 Transfer Out.

•User selects the stock unit to transfer to.

•User selects stock and commits transactions.

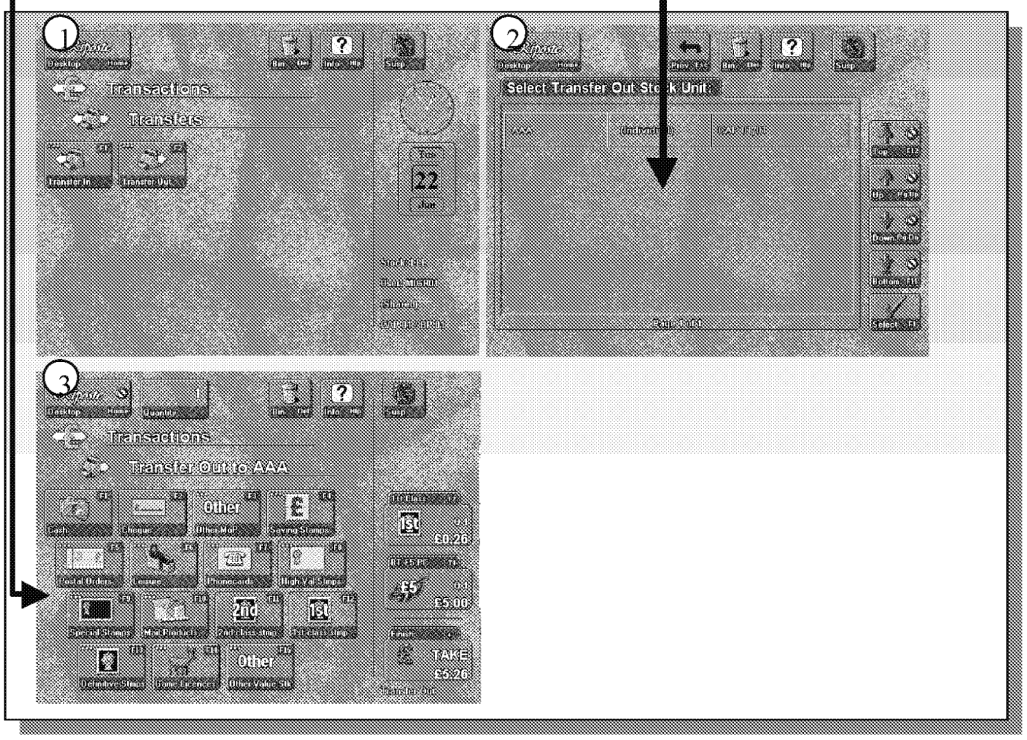


Figure 27 - Menus for the Transfer Out of Stock

The transfer works by:-

- Writing a marker to the message store.
- The transfer transactions (products to transfer).
- A Transfer trailer which identifies the transactions.
- A pending transfers object which identifies the transfer trailer (the objectname is made up of the transfer stock unit + the numeric value of the session id).

See next page for message store breakdown of the above process

FUJ00079425

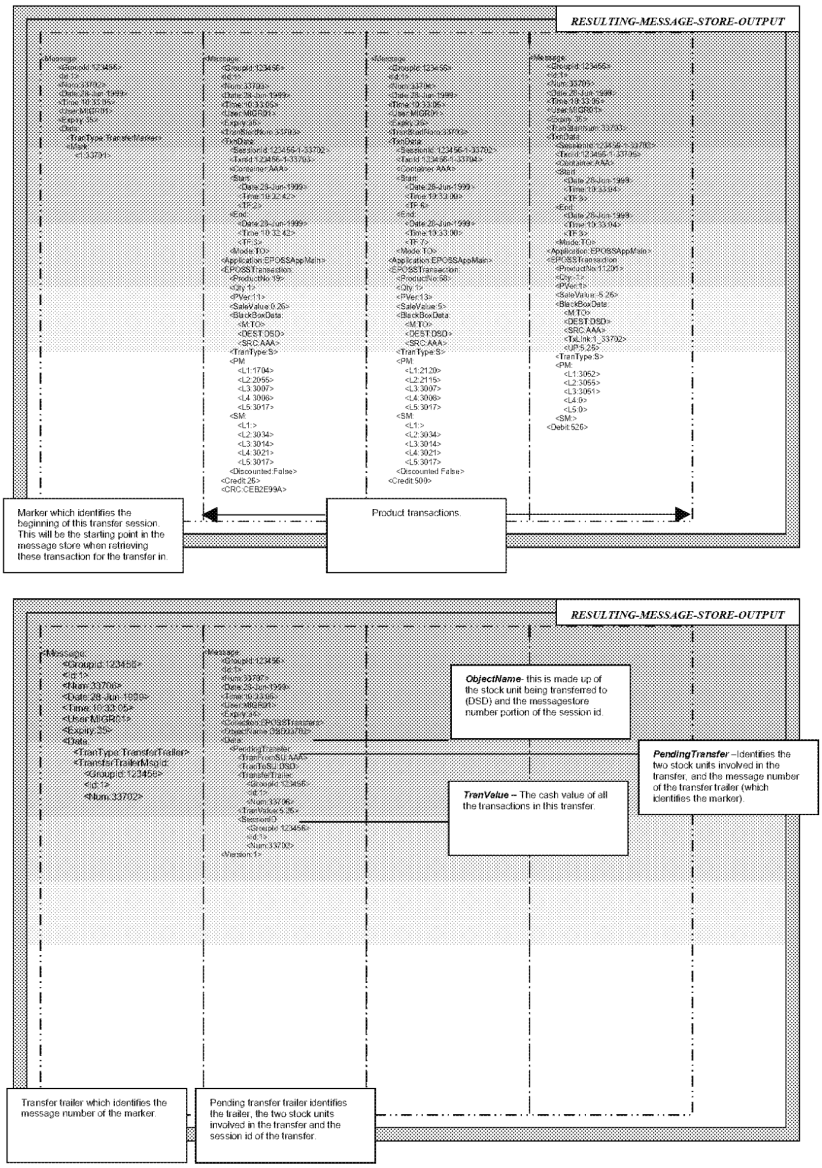


Figure 28 - Message Structures for the Transfer Out of Stock

7.3.5.2 Transfer In.

Transfer in requires a corresponding transfer out to a stock unit. Once the user has selected the transfer in session, and accepted the transfer the stock will become part of that stock units holding figures.

FUJ00079425

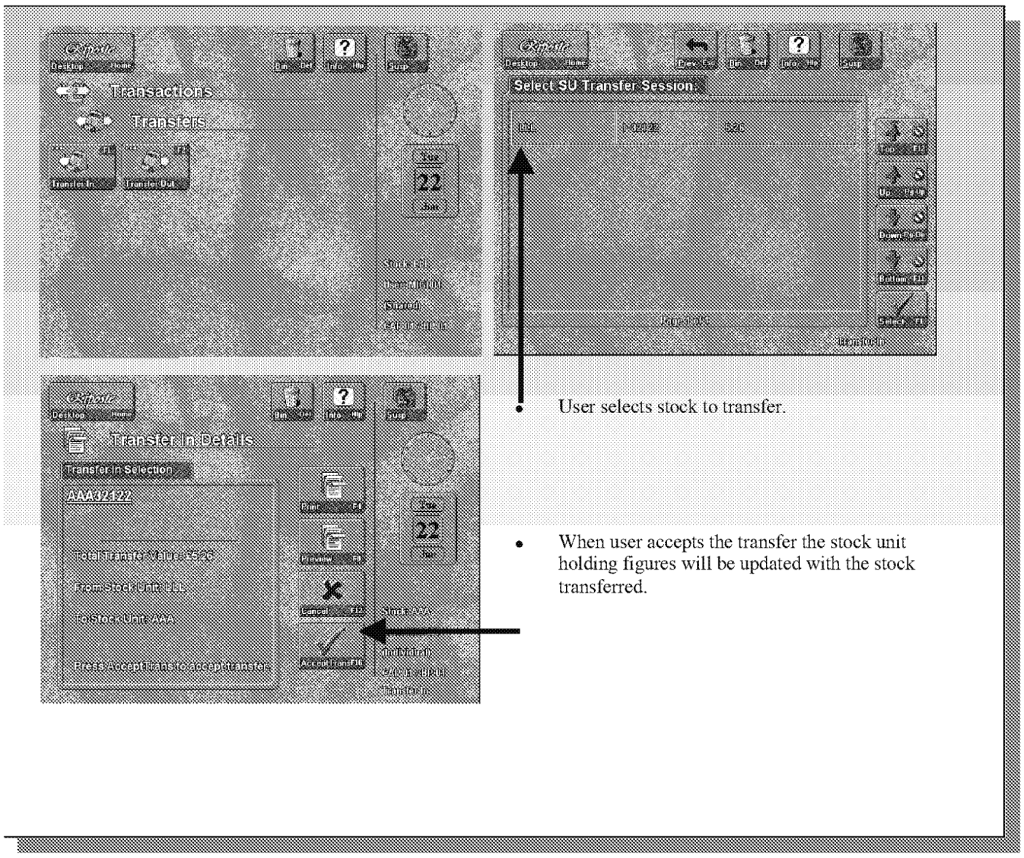


Figure 29 - Menu Structures for the Transfer In of Stock

How it works:

- The transfer list is built up of all the transfer sessions, which have an object name beginning with the current stock unit name (e.g., DSD).
- When the user selects a transfer session, the transfer object is used to retrieve the transfer products.
- These are then written as '*stock*' transactions for this stock unit.

7.4 End Of Day Activities

At the end of each day the user is supposed to make an over night cash holding declaration. If one is not made then on logon on the next day one of the first things the user will be asked to do is enter the previous days overnight cash holding declaration.

FUJ00079425

7.4.1 Overnight Cash Holding Declaration.

Figure 30 - Menu and Message Structures for the Daily Cash On Hand (Onch) Declarations

7.5 Balancing a Stock Unit.

Before a stock unit can be balanced the following processes have to be performed.

7.5.1 Declaring Stock (Shared Stock Units Only), Stamps, Cash.

Declarations are used to allow the user to tell the system what stock, stamps, and cash that is present in their stock unit. The system then compares these declared figures

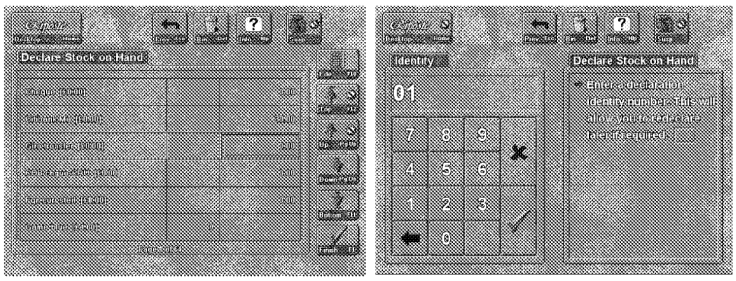
FUJ00079425

with the system figures, if there are any differences these will result in *discrepancy*[1] transactions.

- **Stock**

  Stock is only declared for shared stock units. A user will declare stock by entering each stock items value or quantity on declare stock pick list. When completed the user will enter a stock declaration ID, this will identify the stock declaration to the system.

  If stock is declared again before balancing with a different declaration ID the declaration will be added to the previous declaration, using the same declaration id will mean the system will only use the latest declaration.



- *Declare stock pick list.*
- *Declaration id screen*

Figure 31 - Menu Structures for Daily Cash On Hand Declarations for Shared Stock Units

[1] Discrepancies are explained later in this section.

FUJ00079425

7.5.1.1 Stock Declaration Messages.

All declaration transactions are 'identified' by a persistent object which does the following:-

1) Identifies the starting point (marker) in the messagestore from where the declaration messages start.

2) Identifies the unique id shared by all of the declaration transactions.

3) The starting message number (the first declaration product).

4) *The last message number (the last declaration product).*

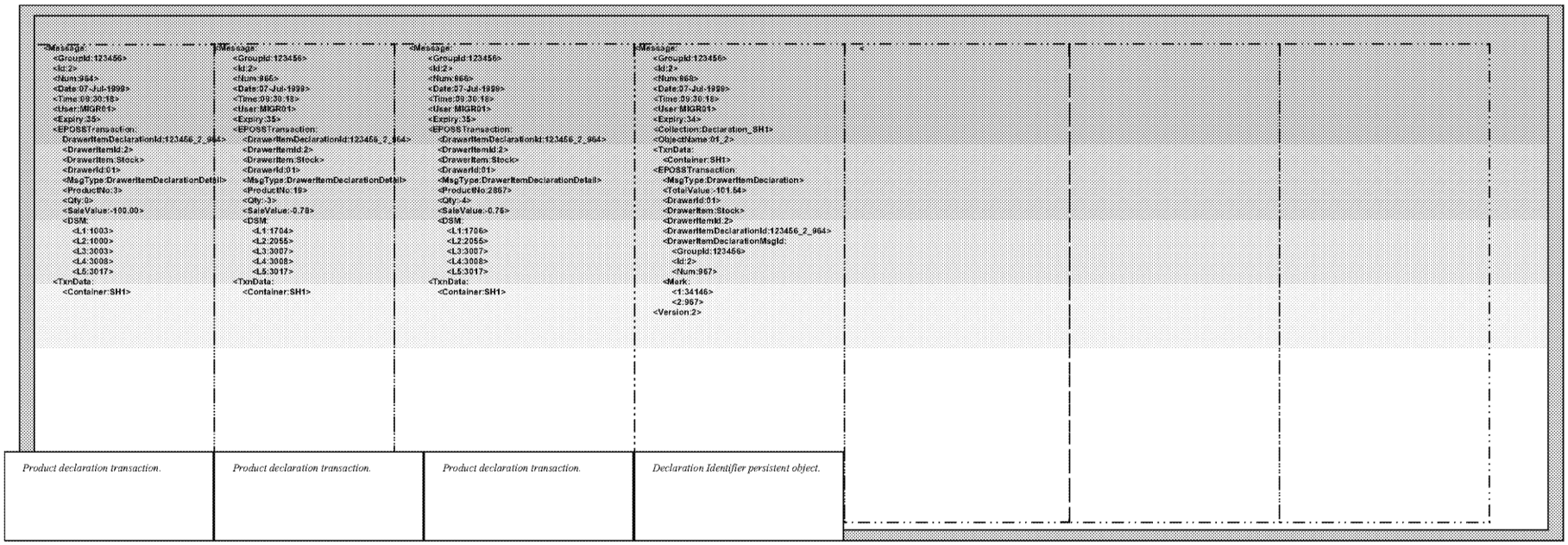
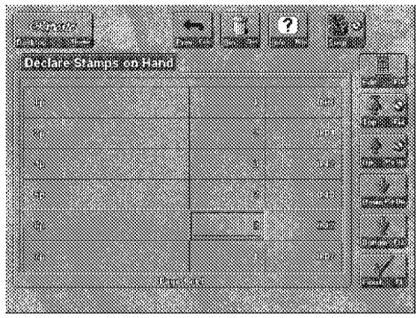


Figure 32 - Message Structures Associated with Stock Declaration

FUJ00079425

- **Stamps**

  Stamps are declared by entering the quantity for each denomination, and when complete, a declaration id. Unlike stock' stamps and cash must be declared by single stock units. No declaration id is required for single stock units.



•Declare stamp pick list.

Figure 33 - Stamp on Hand Declaration Menu

FUJ00079425

7.5.1.2 Stamp Declaration Messages

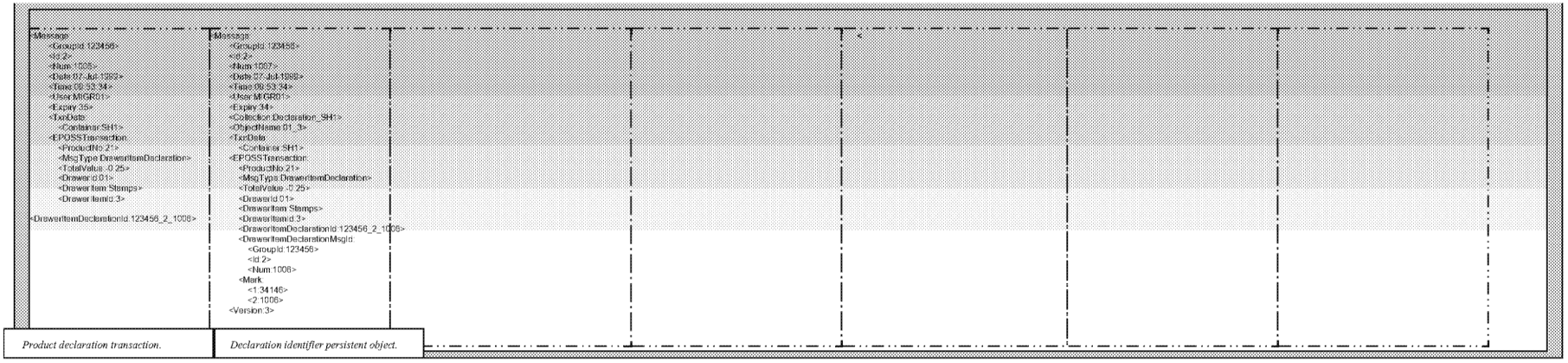
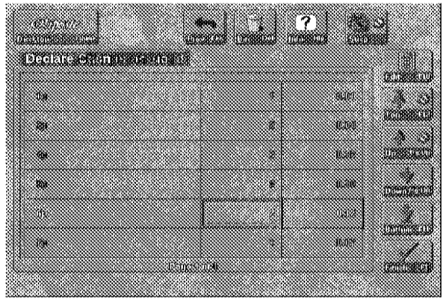


Figure 34- Message Structures Associated with Stamp Declaration

- **Cash**

  Cash is declared by entering the amount for each denomination, and when complete, a declaration id.

FUJ00079425



•*Declare cash pick list.*

Figure 35 -Menu Structure for Cash Declaration

7.5.1.3 Cash Declaration Messages.

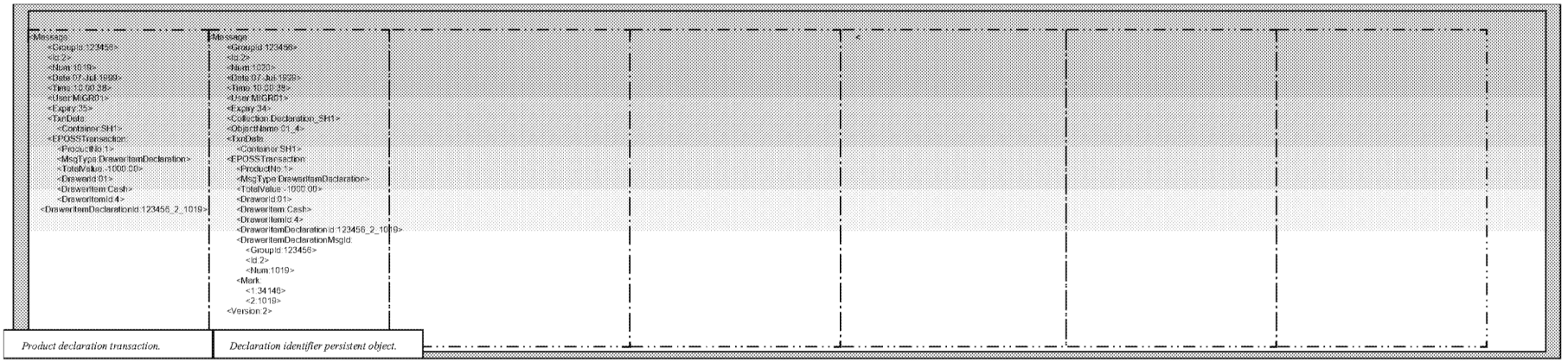


Figure 36 - Message Structures Associated with Cash Declaration

FUJ00079425

7.6 Mandatory Reports and Cut Offs

Before a stock unit can be rolled over certain reports must be printed.

7.6.1 Cut Off Reports

Once a report has been printed, it is then cut off. This cut off tells the system that the report has been run.

The cut off works by writing a marker to the message store. The marker is then compared to the last transaction number for a product of the type in the report, e.g., Giro Deposits. If the transaction number is greater then the report has not been cut off (see Diagram below).

All the transactions for node 110 are accumulated and the largest sequence number is stored (node number + message number). This is then checked against the sequence number in the cut off object (if there is one), and if the transaction sequence number is bigger then the report will need cutting off again (this would mean that there has been another transaction since the last cut off).

7.6.2 All Mandatory Reports

Mandatory reports include:

- Giro Deposit
- APS Transactions.
- BT Bills Daily.
- Giro Withdrawals
- NS Deposits
- P And A Summary
- P & O Encashed.
- DVL V10

7.6.3 Checking for Discrepancies

Discrepancies are system generated compensating transactions. E.g., if £100 is declared by the user, but the system calculates there is £200, a discrepancy '*loss*' transaction would be generated for £100.

Discrepancy transactions are generated immediately after you declare each declaration type, e.g., Cash, Stamps, and Stock, for Individual stock units. Shared Stock unit discrepancy checks don't take place until the balancing process is started.

FUJ00079425

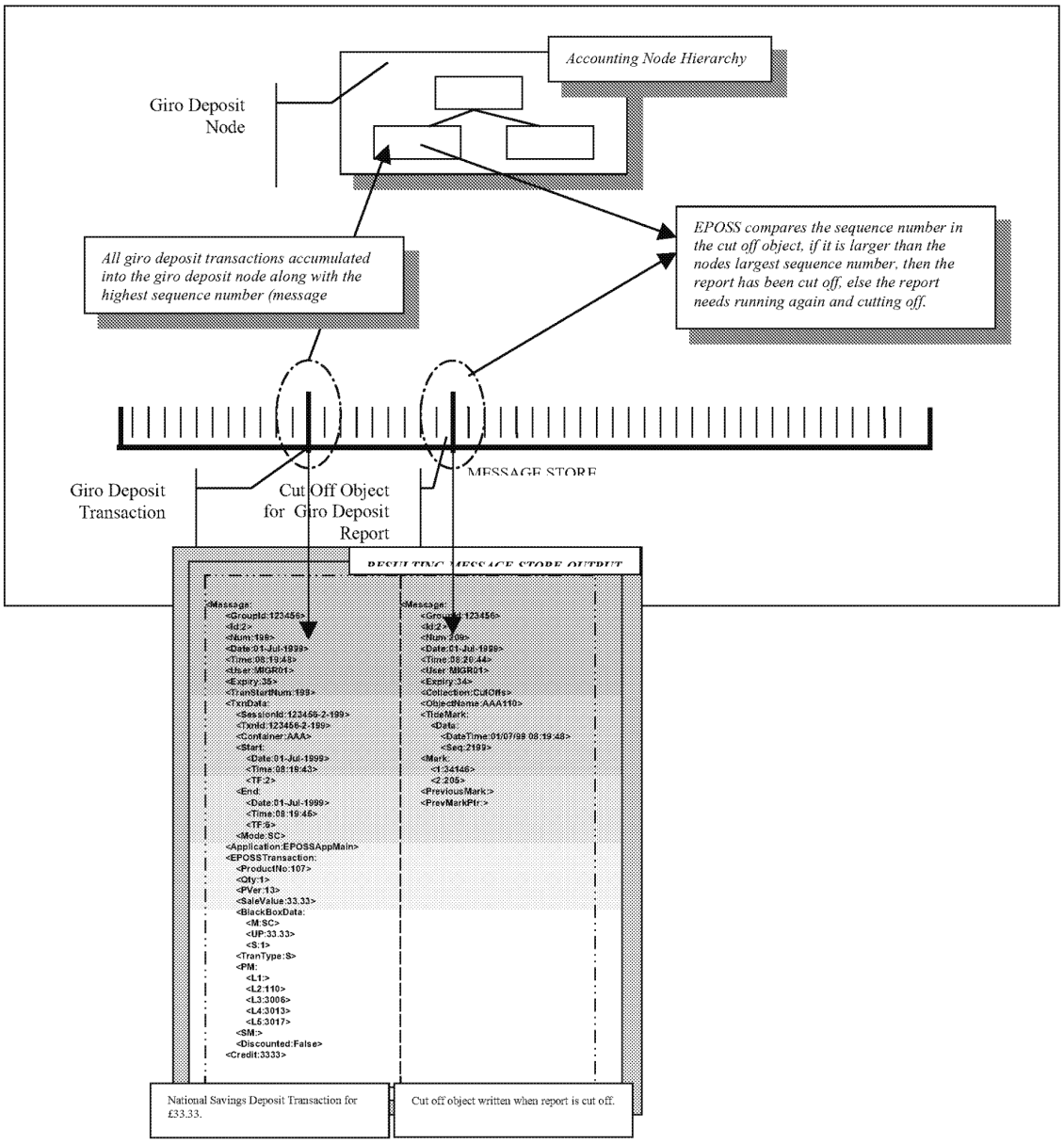


Figure 37 - Cutting Off Transactions

7.7 Rolling Over Stock Unit In to the Next Cash Account Period (or Next Balance Period).

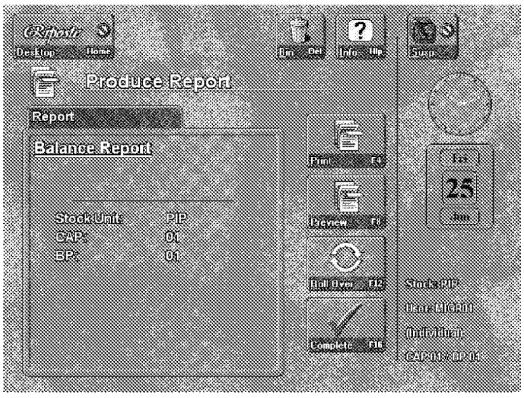
Once all the balancing activities have been completed, which includes printing the stock unit balance report, the stock unit can be rolled over. A stock unit can either be rolled over into the next CAP or BP.

FUJ00079425

7.7.1 Rolling Over Into the Next CAP.

When the user has selected to roll the stock unit over into the next CAP the system will retrieve all the stock and non stock items that have been sold / transacted from the beginning of the CAP.

The stock items are then written to the messagestore and become the stock holdings for the next CAP. None stock items (known as none inventory figures) are also written to the message store but are used for reporting, and suspense container purposes.



- Stock Unit Rollover Screen

- User selection of CAP or BP Rollover.

Figure 38 - Menu Structures for Rolling a Stock Unit Over

7.7.2 Stock Unit Rolling Over Processing Activities

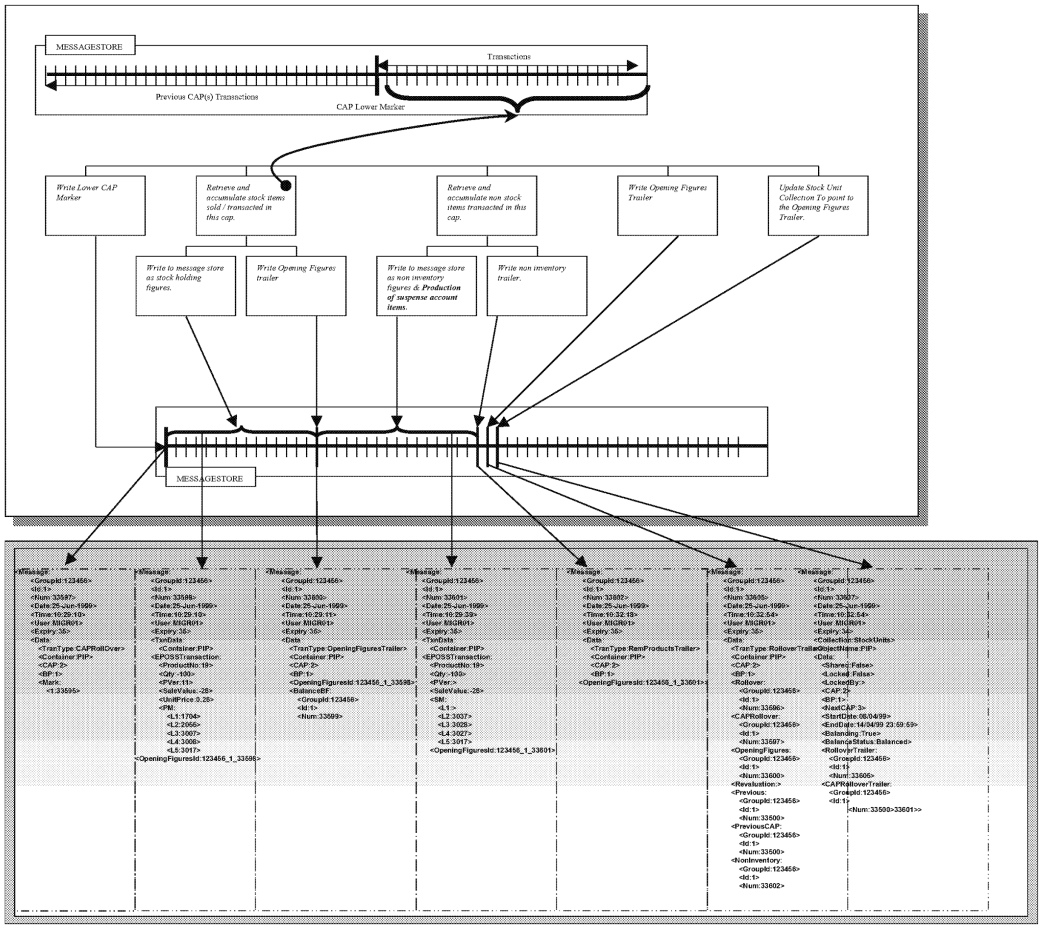


Figure 39 - Message Structures for Rolling a Stock Unit Over

7.7.3 Suspense Account Items

Suspense account transactions are transacted and stored at the stock unit level. All suspense account products are in the *housekeeping* menu, and are transacted in the mode HK.

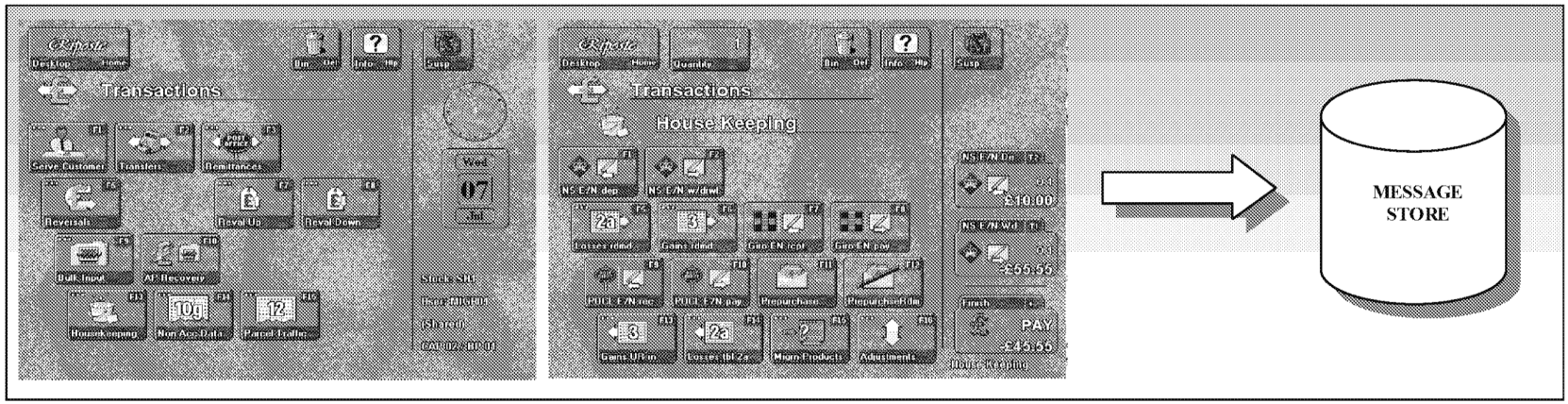


Figure 40 - Menu Structures for House Keeping Functions

7.7.3.1 Stock Unit Suspense Container Storage

When the none inventory figures are retrieved, each transaction is checked against the corresponding cash account mappings (for the suspense tables) for the product which will identify whether the product is a suspense account item. When a suspense account product is identified a SuspenseContainer:$$ attribute is added to the transaction, and written as part of the non inventory figures.

FUJ00079425

7.7.3.2 Suspense Account Process Overview at Stock Unit Rollover

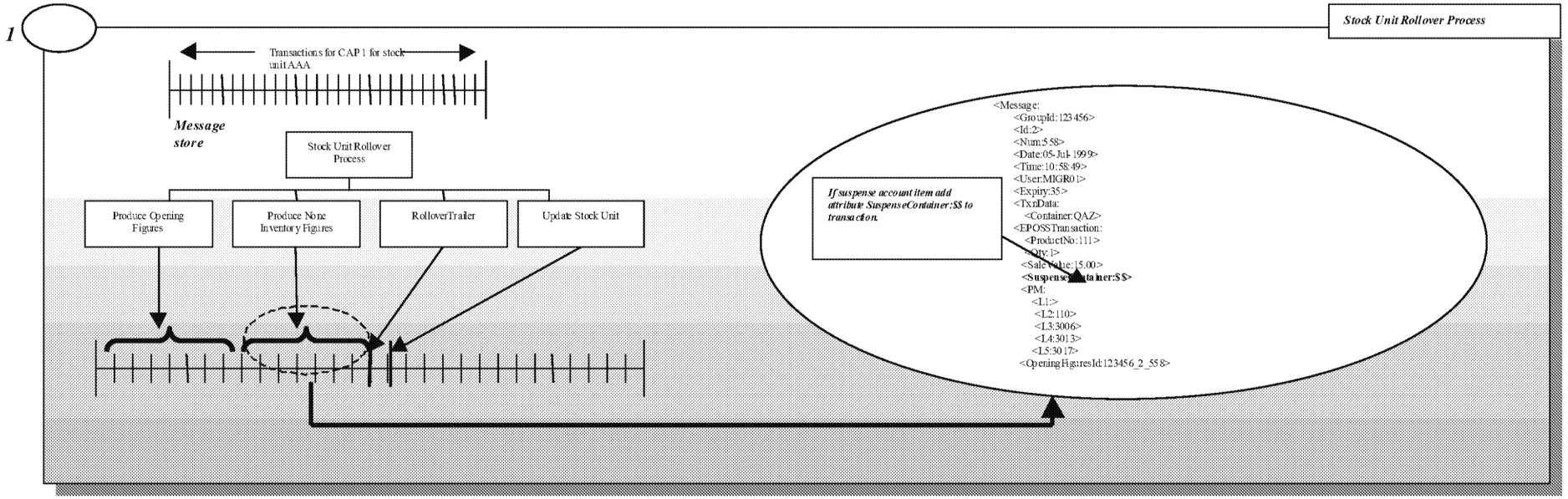


Figure 41- The Writing of Suspense Account Information

7.7.3.3 Suspense Account Process At Office Figures Production.

At office figure production time (Cash Account Report) the previous CAPs suspense account items are copied into the current CAPs office figures, to get a net suspense account.

FUJ00079425

7.8 Office Activities.

When all of the stock units have been rolled over, the office accounting activities can then take place.

Office activities include:

1. Production of office figures (## container) – System activity.
2. Production of the Trial Cash Account.
3. Office Rollover.
4. Product of the Final Cash Account.

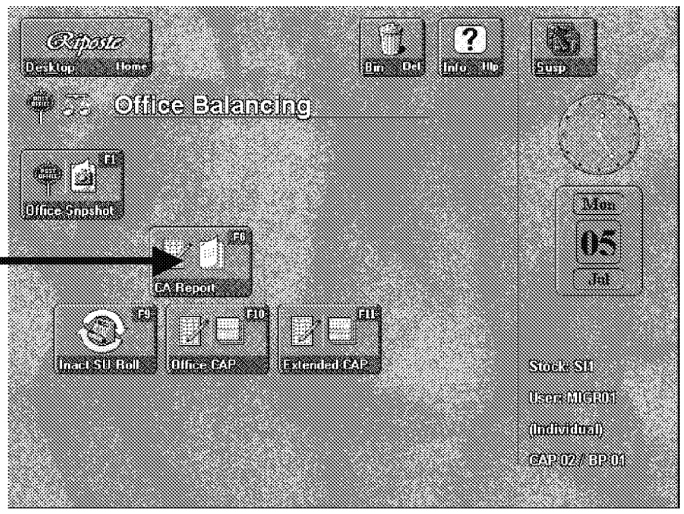


Figure 42 - Menu for Office Balancing

7.8.1 Office Figures

When the CA Report button is pressed the office container figures are produced by the system. These figures are the holding figures and the non-inventory figures for each of the stock units (diagram below shows the process).

FUJ00079425

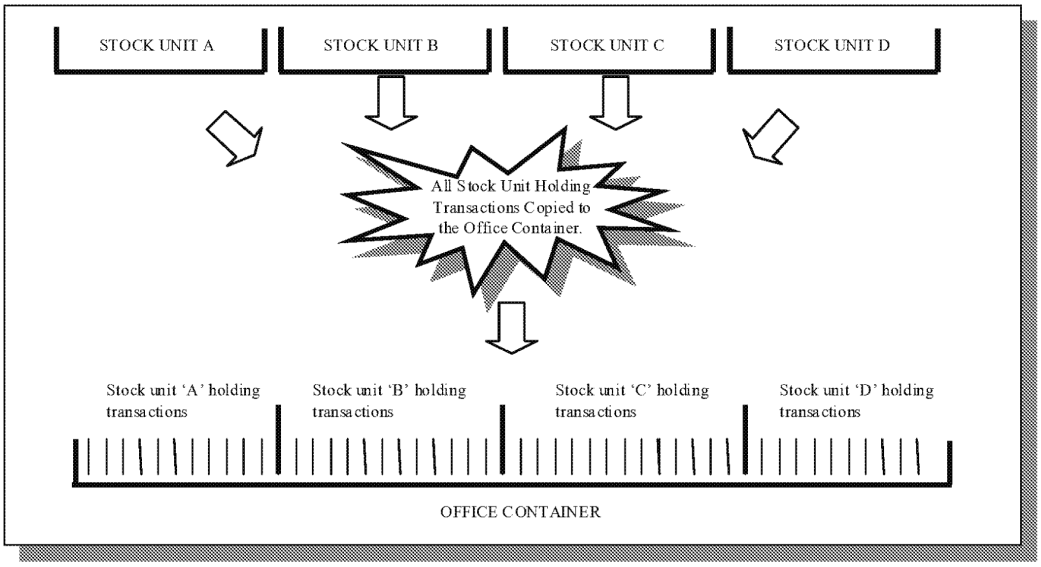


Figure 43- The Production of Office Container Figures

7.8.2 Cash Account Figures.

After the office figures have been produced they are accumulated by product and written as cash account transactions, ready for cash account production.

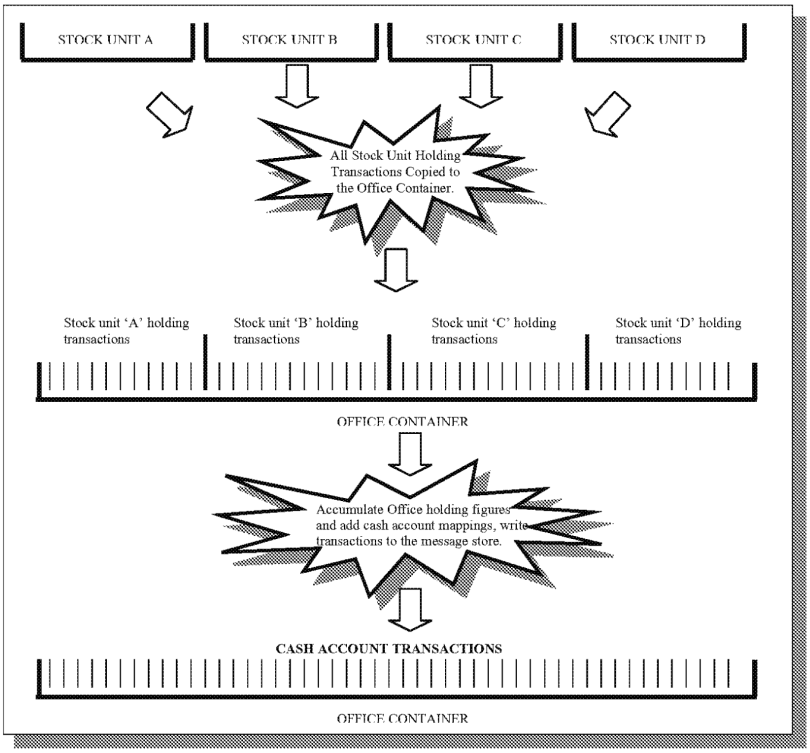


Figure 44 - The Production of Cash Account Figures

FUJ00079425

7.9 Cash Accounting Node Hierarchy.

The cash accounting node hierarchy consists of a seven level node structure, and is used only at cash account production time to produce the cash account.

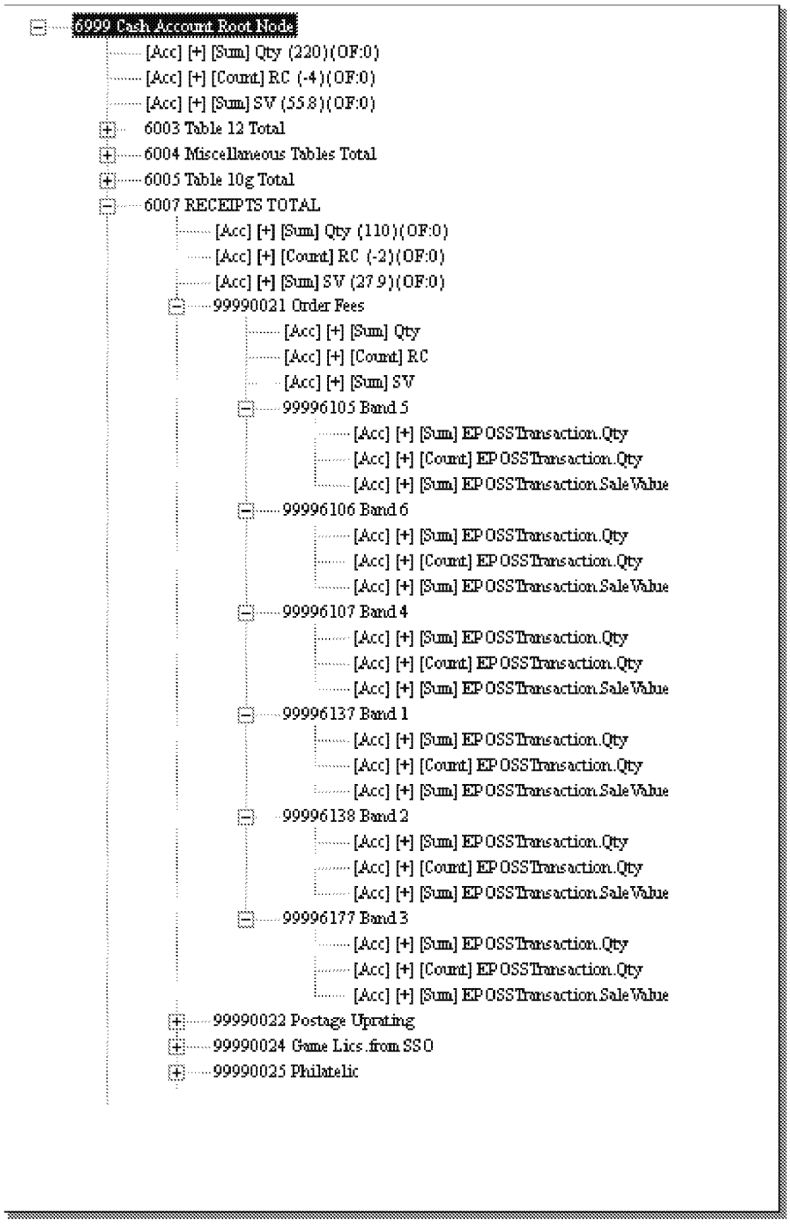


Figure 45- A Section of the EPOSS Accounting Node Hierarchy

FUJ00079425


```
<Message:
    <GroupId:123456>
    <Id:1>
    <Num:11728>
    <Date:29-Jul-1999>
    <Time:07:34:14>
    <Expiry:34>
    <TranStartNum:11523>
    <Collection:_CAMappings>
    <ObjectName:19_01>
    <StartDate:01-Jan-1996 00:00:01>
    <EndDate:>
    <RData:
      <Data:
        <ProductNo:19>
        <Leaf:
          <P:19>
          <Tab:CA-PROD-MODE>
          <N:99995011>
          <PN:0>
          <T:SC>
          <PNL:0>
          <M:
            <L1:99995011>
            <L2:99992057>
            <L3:6006>
            <L4:99992072>
            <L5:99991085>
            <L6:6008>
            <L7:6999>
          <MIMAN_SEQ:90009001>
        <Leaf:
          <P:19>
          <Tab:CA-PROD-MODE>
          <N:99998028>
          <PN:3038>
          <T:ROOP>
          <PNL:2>
          <M:
            <L1:>
            <L2:>
            <L3:99998028>
            <L4:6002>
            <L5:99991082>
            <L6:6008>
            <L7:6999>
          <MIMAN_SEQ:90009014>
        <Leaf:
          <P:19>
          <Tab:CA-PROD-MODE>
          <N:99997015>
          <PN:3030>
          <T:ROSD>
          <PNL:2>
          <M:
            <L1:>
            <L2:>
            <L3:99997015>
            <L4:6001>
            <L5:99991077>
            <L6:6008>
            <L7:6999>
          <MIMAN_SEQ:90009011>
        <Leaf:
          <P:19>
          <Tab:CA-PROD-MODE>
          <N:99996008>
          <PN:3037>
          <T:RIOP>
          <PNL:2>
          <M:
            <L1:>
            <L2:>
            <L3:99996008>
            <L4:6009>
            <L5:99990036>
            <L6:6007>
            <L7:6999>
          <MIMAN_SEQ:90009014>
        <Leaf:
          <P:19>
          <Tab:CA-PROD-MODE>
          <N:99996108>
          <PN:3031>
          <T:RISD>
          <PNL:2>
          <M:
            <L1:>
            <L2:>
            <L3:>
            <L4:99996108>
            <L5:99990028>
            <L6:6007>
            <L7:6999>
          <MIMAN_SEQ:90009001>
    <Version:1>
    <CRC:D92F7C5E>
```

Figure 46 - An Example of a Cash Accounting Node

FUJ00079425

7.10 EPOSS End of Day Service

The systems management of the Pathway Counter Services is described in the Counter Management Support Guide – see Reference 12. The reference describes the systems management of the counter as a whole in which the EPOSS end of Day applications are those supported by the EPOSS Development Team.

FUJ00079425

8 Using the Diagnostic Tools

8.1 Overview

This section lists and describes the tools and information available for investigating problems with the EPOSS product.

EPOSS uses a number of elements of the counter environment to support it providing information that may be used to diagnose faults. The use of these facilities in providing support information is based on the nature of the problem, the domain of operation in which the problem has occurred and the priority of needing support involvement urgently. These elements are represented diagrammatically as follows:

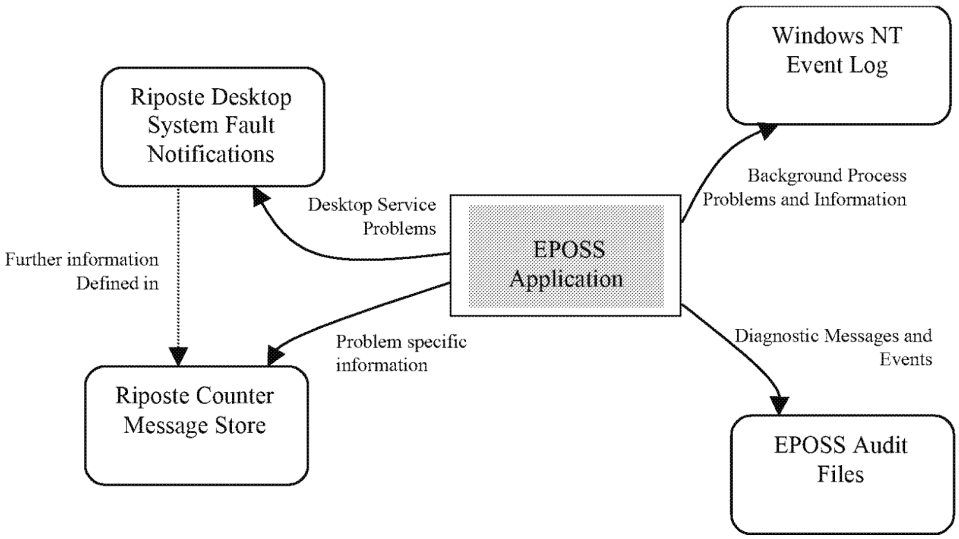


Figure 47 - Elements for Fault Diagnosis

8.1.1 Riposte Counter Message Store

The Riposte Message Store is the central "database" for each Counter's business operations. It provides the reference data used to drive the EPOSS Product and it is the physical data store for recording EPOSS Transactions and Business Events. Business Events are recorded to satisfy requirements for audit and user reporting and should not be confused with Diagnostic Events and Fault recording events (see below). The Riposte Message Store is an element of the Riposte Service in which EPOSS operates as a component of the counter platform architecture and is accessed by Riposte Tools. Riposte Tools are outside the scope of this document.

The Riposte Message Store is the primary vehicle for diagnosing faults in the EPOSS Product and should be a mandatory attachment to any PinICL raised. It is further appropriate that the full counter message store be available as the key components of reference data driving the EPOSS Product are necessary.

FUJ00079425

The Riposte Message Store is not normally used by EPOSS as the main register for writing diagnostic information required for fault investigation. However one of the key weaknesses of the message store alone can be the inability to relate user actions with system (EPOSS) events. On occasion therefore the system does record information to the Message Store in order to relate a particular action with some element of the business being conducted.

8.1.2 Windows NT Event Log

The Windows NT Event Log is a standard component of the Windows Operating System and is accessible through Windows 'Commands'. The NT Event Log is used by EPOSS primarily for reporting events, which demand Support intervention. In this context such events are primarily limited to processing operating in background, for example the End of Day Service Applications or the Riposte Desktop Start Up processing. When the user is using the desktop service events which demand support action are normally reported to the user to alert their attention to the fact (see System Fault Notifications below).

Serious NT Events can then be intercepted remotely and the appropriate support action taken.

8.1.3 Desktop System Fault Notifications

The EPOSS In-Day Service provides an online service to the user community. It is supported by a number of integrated EPOSS Applications driven largely by Reference Data. The service is responsible for ensuring consistency of its own business operation.

As a software product however the possibility of unexpected system faults has been identified and a standard policy for reporting such events to the user has been implemented.

This policy provides for an unexpected event being reported to the user so they may contact the Horizon Helpdesk for assistance. Such events are defined as problems that the user themselves can do nothing about and will need support assistance of some description.

To provide for any diagnosis of the problem being reported system fault notifications are recorded in the Riposte Counter Message Store with accompanying information that may aid support.

8.1.4 EPOSS Audit Files

EPOSS Audit Files are a component of the EPOSS solution and are used to provide a vehicle for logging diagnostic information for problem investigation. The files are managed automatically within the product.

There is no limitation on the information recorded in the audit files, their primary use being to record diagnostics, which aid difficult problem investigation. It is recognised that diagnostic information being written to files is an overhead on performance and

FUJ00079425

hence that information should ceased to be written once the specific problem is resolved.

One of the principle uses to which the audit files are put is the recording of the EPOSS Application component versions being executed in a particular desktop session.

The EPOSS Audit files tend not to be used to report a specific error to be investigated. Rather they record information about an error that is likely to have been previously reported but about which further information is required.

8.2 Architecture

8.2.1 Riposte Counter Message Store

The Riposte Counter Message Store is part of the Riposte Service running the counter. A message store is resident on each counter within the outlet and each message store is replicated across all counters in the outlet. The message stores are also replicated to the correspondence server with which the outlet is connected.

As such any message of any message store can be identified from the correspondence server, as identifiers are standard attributes of any message written. The attributes are:

- Group, identifying the outlet amongst all other outlets
- Id, identifying the counter (node) amongst all other counters within the outlet
- Num, identifying the message amongst all other messages within the message store.

An example Message Store Message might therefore be:

```
<Message:<GroupId:123456><Id:1><Num:27><Date:30-Sep-
1999><Time:10:42:05><Expiry:120><TranStartNum:26><SecurityEvent:<EventNa
me:UserGroupModified><UserGroup:SUPPORT>><CRC:E6928F>>
```

Access to the message store from within applications is via Riposte APIs. Access to the message store for support purposes is through Riposte Tools.

The Riposte Message Store is located in the directory C:\Riposte on the counter.

Riposte Tools are located in the directory C:\Counters\Bin on the counter.

The Riposte Counter Message Store is not private to EPOSS but services any Counter Application operating within the Riposte Service.

8.2.2 Windows NT Event Log

The NT Event Log is a standard feature of Windows and resides on the Counter. APIs are provided to record events in the Event Log which the EPOSS Product makes use of.

The NT Event Log is common to the NT installation and is accessible via NT utilities, primarily the Event Log Viewer.

FUJ00079425

The APIs into the event log provide for events to be categorised in a number of ways. Of most relevance is the fact that events can have a severity level, which EPOSS makes use of. However all events recorded by EPOSS are classed as Application, another category of record.

The Event Log may also be accessed remotely which allows for the different severity levels of event to be monitored and trapped allowing the appropriate support intervention to be invoked.

The NT Event Log is not private to EPOSS but is available to any application or service on the counter.

8.2.3 EPOSS System Fault Notifications

EPOSS System Fault Notifications are a software feature of the EPOSS Product. It is a design principle of the EPOSS Architecture that unexpected system faults in the use of the EPOSS In-Day Service shall be monitored, trapped and reported to the user. Such events should be at a minimum and only result from an unexpected error.

The list of faults are tabled in Appendix E.

What the feature does provide in addition to the notification to the user is additional information written to the Riposte Counter Message Store.

~~Details of the system fault being reported are provided to the user for reporting to the help desk.~~

An unexpected system defined fault is defined as a program detectable error that is reported to the user, but cannot be resolved by any direct user action. Explanation is given as to what the user may be expected to see when an unexpected system fault occurs. The primary purpose of this document is to define a catalogue of the unexpected events that the system will react to. The catalogue goes beyond this and explains the nature of each event that the appropriate corrective action can be put in place.

The intent of informing the user is to advised them that a requested action cannot be carried out, leaving the system in a tidy state for other actions to continue if possible, and provide details that they can report the problem to the help desk. The whole concept of an unexpected system fault defines an error whereby an action request from the user cannot be carried out by the system because the system information is not present or has become corrupt.

The policy for any system, detected fault is:

- To provide the user with a clear unambiguous message that such an event has occurred
- To inform the user of information necessary for them to report both the fault and that information to the help desk, from which identification of the fault can be investigated quickly

FUJ00079425

- To leave the user in a consistent state whereby they may carry on other actions that do not suffer from the fault.

Examples of system faults could be:

- missing mandatory reference data
- soft references to unknown applications
- unexpected Visual Basic errors

In all these cases the system determines that it has insufficient information to carry on, or that the information is incorrect with which it is to work, however there is nothing the user nor the system can do. In all respects there is a technical error in the system and responsibility for correction of that error rests with someone other than the outlet Post Master.

This type of error is contrary to a user defined error whereby the system reports an error to the user advising them of having incorrectly followed procedure. An example of a user error would be attempting to produce a Cash Account without having rolled the stock units into the next CAP first. In such cases correction of the error should be entirely down to the users' operation of the system and no involvement of system support should be necessary.

There are other types of error that fall between these two specific categories, for example Reference Data Errors. At CSR+ validation is being introduced so that if a product is not available in the appropriate transaction mode for the current desktop mode in which it is being transacted, then the user will not be able to transact that product and will have an error reported to them to advise.

Again in this category of error the user cannot do anything to resolve the situation themselves, and the problem rests with POCL reference data to investigate. It may be in this case that the error being presented to the user is correct and that indeed that particular outlet has not been granted permission to transact that particular product.

It is important therefore that a particular type of error is objectively identifiable and that responsibility for its investigation and resolution can therefore be determined simply. Such faults are defined as unexpected in the sense that they should never occur. The development process of testing at different stages should have established this in the first place. However any system should be robust and react to the unexpected.

The introduction of unexpected event handling in EPOSS is a recent addition that will be incrementally improved upon as development takes place further. At this stage however the catalogue is limited to the few situations that have been trapped.

~~The System Faults catalogued by the document are limited to those provided by the EPOSS Vertical Application, and not other counter products, specifically OBCS which is also maintained by the EPOSS Development Team.~~

FUJ00079425

To satisfy this policy EPOSS implements a standard architecture.

When an unexpected system fault is detected by the application a pop up tablet will be presented to the user advising them of the fault having occurred. The tablet provides three items of information:

1. The narrative on the tablet advises the user that this is a system fault.
2. The first of two coded values denotes a particular fault having been detected.
3. The second of the two coded values denotes a message number in the counter message store that records further information about the fault.

As a consequence of having been presented with this tablet the user is given a single option to acknowledge the error, as a result of which the system will tidily abandon the process currently being undertaken on the user's behalf and will return control to the desktop in a consistent state. Due to the nature of not being able to anticipate the seriousness of an error the position the user is returned to may not be an ideal, but is at least consistent.

The following diagram demonstrates the effect on the user:

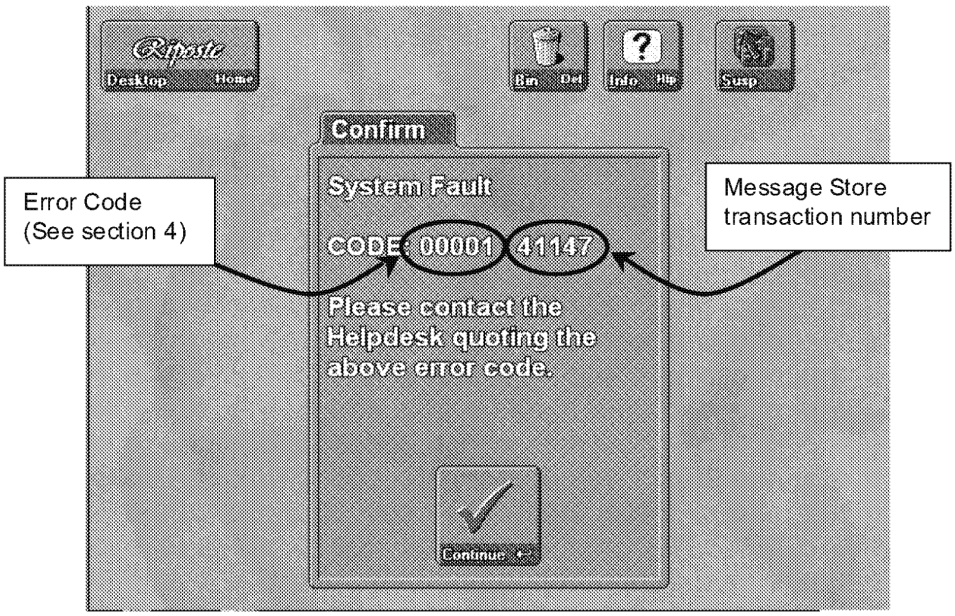


Figure 48- The Error Code System

Unfortunately there is currently a conformance exception to this standard within the system. This is documented in the following section.

~~The remainder of the document provides the current unexpected error catalogue.~~

It should be noted at this stage however that the implementation within the software of issuing the above tablet to the user is not supported by a Message defined in reference

FUJ00079425

data. A conscious decision was taken in architecting the solution to "hard code" the message rather than use the standard mechanism of a reference data message. The reason for this is that by its very nature a system fault indicates an event, which is indeterminate in its nature. As such relying on mechanisms to inform the user that themselves may well be failing would not be advisable. For example if the system fault being reported is that a mandatory item of reference data is missing then relying on a reference data message to be present, which itself is mandatory would be unwise.

8.2.3.1 An Exception to System Fault Reporting

Whilst the design standard for the reporting of system faults is in place it has been discovered that there is an exception to the standard already implemented in the system. This, exception is planned to be brought in line with other instances, however for the purposes of documentation completeness has been defined here pending it being brought into line.

In these cases the following tablet will be displayed to the user:

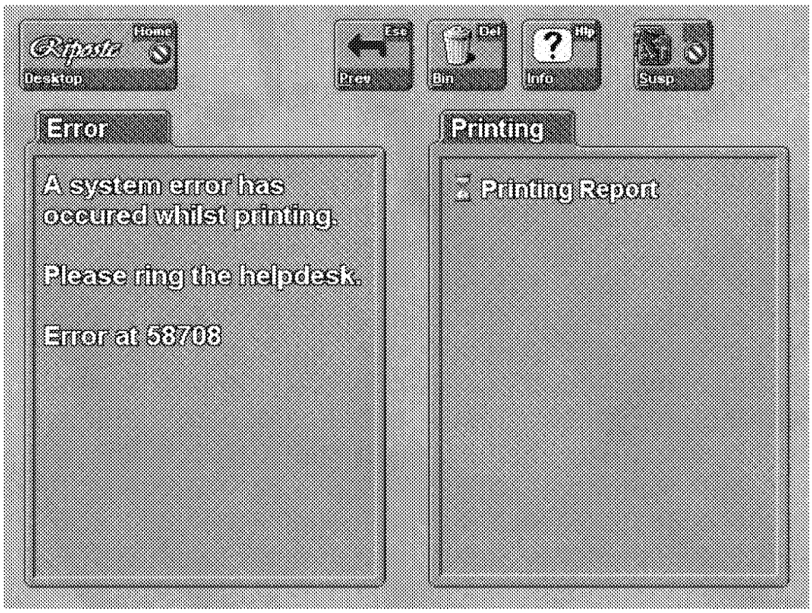


Figure 49 - An Exception to System Fault Reporting

The exception shown above concerns the reporting of unexpected errors in the Reporting Service of EPOSS. Errors there reported are displayed with a different Tablet not identifying a coded 'label' (Error Code) denoting the type of error that has occurred.

The displayed fault still however provides a message number indicating further information, which may be found about the error in the counter message store.

8.2.4 EPOSS Audit Files

EPOSS Audit Files are a software feature of the EPOSS Product. They are used to record diagnostic information primarily in the investigation of serious but

FUJ00079425

unexplainable errors. Largely they are used when an intermittent fault cannot be reproduced in development and requires evidence from another instance in the live estate.

EPOSS Audit Files are managed automatically by the EPOSS Product.

An audit file is created by the EPOSS Product when a request within the software is made to record some diagnostics. A file will be created named AUDIT_xxx.LOG in the directory C:\TEMP, where xxx is the day of the week, corresponding to the date on which the diagnostics are being written. So EPOSS will maintain a maximum of seven files for the week.

EPOSS also manages the fact that these files may become quite large and maintains a system of 'archiving' files. If therefore on opening the file of a particular day it is found that it is last week's file for the day, then the software will rename the file as an archive and create a new file for today. EPOSS therefore maintains files named AUDIT_xxx-1.LOG, AUDIT_xxx-2.LOG, AUDIT_xxx-3.LOG, together with AUDIT_xxx.LOG, being four week's worth of files in all. Files over four weeks old are deleted.

8.3 Which Domain is Propagating a Fault?

Support of the EPOSS Product, as has been described in the previous sections, requires an understanding of which domain is demonstrating a fault. The specific use of the Diagnostic Tools will depend largely on the domain.

8.3.1 Migration

As a one off operation the migration of an outlet to Horizon is a very specific bespoke exercise. In this context the support for the migration exercise is explicitly covered in the HFSO Guides (see References [13] and [14]).

However the Office Migration element of the exercise is undertaken within the Riposte Desktop. As such the Manual Migration process may utilise the Riposte Message Store, the NT Event Log, EPOSS System Fault Notifications and EPOSS Audit Files to record faults. At this time however no such usage exists.

8.3.2 EPOSS In-Day Service

The EPOSS In-Day Service utilises all of the elements of the diagnostic tooling provided for EPOSS Support, namely **Riposte Message Store, the NT Event Log, EPOSS System Fault Notifications and EPOSS Audit Files** to record faults.

8.3.3 EPOSS End of Day Service

The EPOSS End of Day Applications which operate within the end of day domain are background tasks operating outside of the EPOSS Desk top service. As such provision for fault reporting is limited. Faults being raised by the End of Day applications primarily use the **NT Event Log** to report errors. However the **Riposte Message Store** may also be used to record additional diagnostic information, specifically if the reconciliation process is run in 'debug' mode. The EPOSS Audit Files have so far not

FUJ00079425

been utilised for the end of day applications as no problem has been reported that cannot be diagnosed.

8.4 Diagnostic Tools used to Directly Report a Fault

8.4.1 NT Event Log

EPOSS may record an error to the Event log, described as a 'Red Dot' event. A Red Dot event is remotely managed by Tivoli to intercept their occurrences and notify support. The catalogue of Events issued by EPOSS is defined in Appendix A and defines those which are Red Dot Events as Type Error. Diagnosis of the fault may be possible from the defined explanation in appendix A. Red Dot Events are unexpected events and will require support activity.

8.4.2 System Fault Notifications

EPOSS may issue a system fault notification during the In-Day service as a result of an unexpected error. Such notifications are defined in Reference [10]. Diagnosis of the fault may be possible from examination of the associated information provided in the message store. System Fault Notifications are unexpected events and will require support activity.

8.5 Diagnostic Tools used as an aid to Investigate a Fault

Many problems raised do not result directly from a direct failure of the EPOSS Product but are revealed in unexpected results experienced by the user of the counter. In this case the Diagnostic Tools are used to trace any potential fault back to its source. Unexpected results in the balancing process are a primary instance of such problems.

8.5.1 Riposte Message Store

The Riposte Message Store for the counter from which a potential error has occurred is a mandatory requirement of any problem investigation of the EPOSS Product. Furthermore the message store should be a full message store as so much of the system is driven by reference data and uses results from earlier operations.

The Riposte Message Store is normally extracted from the correspondence server to which it will have been replicated and can be provided in text or binary formats. From this it is possible to textually trace the actions leading to the discovery of a potential fault, or indeed it is possible to run up the system from the message store to attempt to recreate the circumstances.

On occasion it may be necessary to supply the actual counter message store itself as problems have been experienced where the counter message store is not an exact replication of that on the CS.

FUJ00079425

A complete specification of the syntax of message store entries is provided in the EPOSS Attribute Grammar Catalogue (see Reference [15]) and the Riposte Attribute Grammar Catalogue – Messages (see Reference [16]).

8.5.2 Message Store Analysis Tools

Two analysis tools have been produced in order to make the process of investigation of Message Store evidence less labour intensive. These two tools are:

- RMsg2XL.exe A utility to search a text message store file for specified criteria and optionally place the output into a Microsoft Excel spreadsheet, sorted into date/time sequence
- CheckBalanceR216.exe A utility to check the accounting accuracy of the Stock Units and the Cash Account for any given CAP. This version operates using Riposte Build 216.
- CheckBalanceR223.exe A utility to check the accounting accuracy of the Stock Units and the Cash Account for any given CAP. This version operates using Riposte Build 223.

The tools are described in further detail in Section 7 of this document.

8.5.2.1 Text Message Extraction (RMsg2XL.exe)

8.5.2.1.1 General Description

This software is designed to enable a user to select specific Riposte Messages from a text version of a message store using multiple criteriaSearchCriteria in a single pass of the fileSourceFileName. The extracted messages are written to a fileDestinationFileName, optionally in CSVCSVOutput Format or in Microsoft Excel WorkSheet format sorted into date/timeSortDateTime sequence.

The Source File must be a text file containing Riposte Messages in attribute grammar format.

8.5.2.1.2 File Details

The 'File Details' section of the input form allows for the specification of the Source File and DestinationFileName Destination file SourceFilePath.

The SourceFileNameSource File may be specified with or without its file extension. If no file extension is specified the software will assume an extension of '.txt'.

The Destination FileDestinationFileName will default to the same name as the SourceFileNameSource File but with an 'ex' added. The default extension will be '.txt' unless the 'CSV Output' option has been chosen

8.5.2.1.2.1 Source File Name

The Source File Name field allows the user to specify the name of the file to be processed. The source file must be a text file of Riposte Messages (generally this will

FUJ00079425

have been created by using the Riposte Tools such as RiposteNextMessage or RiposteScanMessage) in full attribute grammar format.

The filename entered will be assumed to be a file with a '.txt' extension unless the user enters a filename which includes an extension.

8.5.2.1.2.2 Destination File Name

The Destination File Name field allows the user to specify the name of the file to be created. The Destination file will be a text file of the selected Riposte Messages. If the CSVOutput CSV option is selected then the output file will be created with each attribute separated from its following attributes by a ','. This file can then be read using Microsoft Excel which will format each attribute into a separate column in the spreadsheet. If the CSVCSVOutput option is not selected, then the Destination FileDestinationFileName will contain the selected messages without the comma separation between attributes.

The filename entered will be assumed to be a file with a '.txt' extension unless the user enters a filename which includes an extension.

8.5.2.1.3 Search Criteria

The 'Search Criteria' section of the input form allows the user to enter multiple (maximum of 10) SearchCriteria Criteria with which to search the text message store.

Each criteria is entered by specifying an attributeAttributeName, an operatorOperator and an expected value Value.

The values entered for the attribute nameAttributeName and the expected valueValue can be entered in Upper, Lower or Mixed case. The subsequent string comparisons will always convert the entered data and the data in the input file to a common case before determining whether the criterion has been met. The attribute nameAttributeName should be entered WITHOUT the terminating ':'.

The OperatorOperator may only be one of the list available from the 'Drop Down' list. If a Character Operator (e.g. 'DGE') is typed in the Operator field then it must be entered in the same case as shown in the drop-down list (or be selected from the list if preferred).

Where a date comparison is required, the Date Operators Operator ('DEQ', 'DGE' or 'DLE') must be used. If they are not then a string or numeric comparison will be made (assuming that the value being sought is a valid string or number).

If the Date OperatorsOperator are used then any valid date format may be entered into the Value field and will be converted by the software to the Medium Date format (dd-mmm-yyyy) used by Riposte to store dates.

If the numeric operators are used ('>', '<', '>=' or '<=') then the value being sought must be a valid numeric field.

The 'Like' OperatorOperator may only be used for string searches. If the 'Like' OperatorOperator is used then the value being sought may contain the character '*' at

FUJ00079425

the beginning or end of the string, or both. The '*' character may not appear in the middle of the string being searched.

8.5.2.1.3.1 Attribute Name

The Attribute Name field allows the user to specify any valid attribute in a Riposte Message which the user wishes to use to select messages from the Source File SourceFileName. The Attribute Name can be entered in any case but must be entered in its entirety and must be spelled correctly. The surrounding '<' and '>' characters and the terminating ':' character must not be entered, only the Attribute Name itself. The software does not use the 'nested' capabilities of the Riposte system itself and therefore does not use the '.' nomenclature for identifying sub-attributes. (e.g. the software will accept 'PM' as an attribute to identify the Primary Mappings attribute of a transaction message, rather than the 'EPOSSTransaction.PM' that would be expected in a Riposte Query).

8.5.2.1.3.2 Operator

The Operator field allows the user to select the comparison operator to be used when searching for a matching message.

The valid Operators are:

=	Equal (String or Numeric)
>	Greater than
<	Less Than
>=	Greater than or Equal to
<=	Less than or Equal to
DEQ	Date Equal to
DGE	Date Greater than or Equal to
DLE	Date Less than or Equal to
Like	Wild card '*' allowed for string searches only

The Operators DEQ, DGE and DLE are provided specifically for use with date format attributes. When these Operators are used, any expected valueValue being entered will be validated for date format compliance. Any valid date format can be entered and will be converted to Medium Date format (dd-mmm-yyyy).

The numeric Operators >, <, >= and <= are for use with numeric search values only. If these Operators are used then the Value entered will be validated as a numeric field.

The 'Like' Operator is provided for 'fuzzy' searches. If this operator is used then the Value being sought may contain an '*' character at the beginning or end of the string

FUJ00079425

being sought or at both ends. An '*' character may not appear in the middle of a string being sought. If an '*' is specified then the comparison will be deemed correct if:

- First Character = '*': If the message being read contains the required attribute and the value found in the message store ends in the remainder of the string specified in the Value field then the criteria is deemed to be met.
- Last Character = '*': If the message being read contains the required attribute and the value found in the message store starts with the string specified in the Value field then the criteria is deemed to be met.
- First and Last Character = '*': If the message being read contains the required attribute and the value found in the message store contains the string specified in the Value field between the '*' characters then the criteria is deemed to be met.

8.5.2.1.3.3 Value

The Value field allows the user to specify the expected value of the attributeAttributeName which is to be matched against the messages in the Source FileSourceFileName.

The value may be a date, a number or a string. The content of the Value field will be validated to ensure that its content is consistent with the type of Operator operator selected.

The value entered must be the entire attribute value being sought unless the 'Like' OperatorOperator is used. If the 'Like' OperatorOperator is used then the value being sought may contain the character '*' at the beginning or end of the string, or both. The '*' character may not appear in the middle of the string being searched.

8.5.2.1.4 Operands

Operands are used to link multiple search criteriaSearchCriteria. The Operators ANDOperandsAnd and OROr are provided.

If ANDAnd is selected then the search will require the criteria on BOTH sides of the And Operand to be True before the search criteria are deemed to have been metCurrentCriteria. If Or is selected then if either of the criteriaSearchCriteria on either side of the OROr is True then the search criteria will be deemed to have been metCurrentCriteria.

8.5.2.1.4.1 And

The And Operand is used to link criteriaSearchCriteria where BOTH the conditions on either side of the OperandOperands must be True for the current message in the Source File to be deemed to have met the criteriaCurrentCriteria.

Either the And or the OrOR Operand can be selected for the first criterion SearchCriteriaentered.

FUJ00079425

8.5.2.1.4.2 Or

The Or Operand is used to link criteriaSearchCriteria where EITHER of the conditions on either side of the OROperands may be True for the current message in the SourceFileName to be deemed to have met the CurrentCriteriacriteria.

Either the ANDAnd or the Or Operand can be selected for the first SearchCriteria entered criterion.

8.5.2.1.5 Current Criteria

The Current Criteria section of the form is updated each time a criterionSearchCriteria is entered.

The current criterion will be added to the Search Criteria with the selected Operand. For the first criterion entered it does not matter whether the 'And' or the 'Or' Operand is selected.

Once all required Search Criteria have been entered the user may edit the Current Criteria field to add any required Brackets for the purposes of 'Grouping' criteria.

E.g.

<Date:> DEQ "06-Jan-2000" And <ProductNo:> = "1" Or <ProductNo:> = "2"

Could be edited to read:

<Date:> DEQ "06-Jan-2000" And (<ProductNo:> = "1" Or <ProductNo:> = "2")

The original Current Criteria would match if <ProductNo:> was equal to "2" in ANY record found. The revised Current Criteria would find a match only if <ProductNo:> was equal to "1" or "2" AND the <Date:> was equal to "06-Jan-2000".

If brackets are edited into the Current Criteria field then the software will ensure that the number of open and close brackets are equally matched.

8.5.2.1.6 Search

This command button causes the software to commence searching the Source FileSourceFileName for messages matching the specified criteria CurrentCriteria.

8.5.2.1.7 Cancel

Pressing this command button at any time after the start of the search of the Source FileSourceFileName causes the search to be abandoned. All criteria and variables are cleared and the Source File and Destination FileDestinationFileName are closed.

8.5.2.1.8 Sort by Date and Time

This option allows the user to specify that the Output CSV fileDestinationFileName should be sorted into date and time sequence. If the option is selected then the output .CSV file is opened using Microsoft Excel. The columns containing the Date and Time attributes are edited to remove the attribute grammar content and the remaining values are sorted.

FUJ00079425

8.5.2.1.8.1 CSV Output

The CSV Output option allows the user to specify that the Destination FileDestinationFileName should be produced with each attribute in a message separated from its following attribute by a comma, thereby creating a comma separated variable file which can be read by Microsoft Excel.

CSV files read by Microsoft Excel result in the file being displayed with each attribute in a separate column on the spreadsheet. With the data displayed in this way it is possible to perform calculations on field such as 'SaleValue' with comparative ease.

8.5.2.1.9 Statistics

The Statistics section of the form is used to provide the user with visual feedback on the progress of the search of the SourceFileName for the specified CurrentCriteria.

The display shows:

1. Number of records Read	- Absolute number of messages from the start of the Source File
2. Number of Records Written	- Absolute number of messages written to the Destination File
3. Current Message Number	- The Riposte Node Id and Message Number of the current record being processed from the Source File
4. Percentage Complete	- The percentage of the Source File processed up to the current time
5. Progress Bar	- A graphical representation of the percentage progress

8.5.2.2 Accounting Checks (CheckBalanceR216.exe/CheckBalanceR223.exe)

8.5.2.2.1 General

The CheckBalance utility is dependant upon the version of the Riposte software being used on the PC being used for the analysis. The only difference between the two processes (CheckBalanceR216 and CheckBalanceR223) is the version of the Riposte.ocx used in the compilation of the program.

The purpose of the Check Balance Utility is to carry out the initial analysis of a target message store to determine whether there are any accounting inaccuracies in the Stock Unit Balances or the Cash Account for a given CAP. Investigation of a large number of accounting issues over the first 12 months of live operation has identified that there is a high percentage chance that any reported accounting or Cash Account mis-balance from a live outlet can be traced through the messages written to the message store.

The Check Balance Utility allows the user to either import a message store from a text file, copy an existing Message Store binary (.dat) file to the current Riposte directory or use the current Riposte Message Store Where the source file is either a text file or an existing message store to be copied, the Check Balance Utility makes the necessary changes to the GroupId and NodeId in the NT Registry.

FUJ00079425

8.5.2.2.2 Text File Input

Where the user selects a '.txt' input file, the input file must be a file of Riposte Attribute Grammar messages, preferably a complete message store extract for the given GroupId. The Check Balance Utility reads the text file and discards any records in the file which are longer than 2047 bytes since such messages cause the RiposteImport process to fail (this may be corrected in a later release of Riposte). The only messages which usually have this length are those written by Riposte during a 'session transfer' and therefore removal of these message does not compromise the integrity of the Horizon data in the file.

Once the file has been checked for overlength records, the text file is then imported into the Riposte MessageStore.dat file using the RiposteImport.exe utility provided by Escher Group Ltd. This process is 'shelled' from the Check Balance Utility program and runs in a separate command shell window which may require the user to respond to any warnings of gaps in the messages being imported. In most cases a single response of 'a' followed by a carriage return is all that is required. During this process (which may take a considerable time) the main Check Balance Utility monitors the shelled process and waits for it to complete.

On completion of the import the Check Balance Utility updates the NT registry settings to reflect the details of the GroupId of the imported messages and sets the NodeId to '1'.

8.5.2.2.3 Copy Existing Message Store File

If a complete Riposte Message Store file in binary format (.dat) is available then the Check Balance Utility can copy this file to the current Riposte directory and make the necessary changes to the NT Registry for the GroupId and NodeId. To achieve this it is necessary only for the user to select a '.dat' file as the input file and the Check Balance Utility will then assume that it is dealing with an existing message store file. The user will be required to enter the GroupId relevant to the message store file to be copied.

8.5.2.2.4 Use Existing Message Store

If the user wishes to use the current Riposte Message Store then an option exists to specify this before starting the checking process.

8.5.2.2.5 The Checking Process

The Check Balance Utility allows the user to specify the CAP number to be checked. If no CAP number is specified then the last complete CAP in the message store is assumed to be the one to be checked. On starting the checking process the user is given the option of specifying the date to which the PC should be set so that the Riposte system will not 'expire' messages which might be form part of the requested CAP. The system date will be re-set to it's previous value on completion.

The checking process consists of:

- Identifying the number and identities of the stock units in the message store

FUJ00079425

- For each stock unit identified:
- Retrieve the records containing the stock values at the beginning of the selected CAP and store the values against each Product Number;
- Retrieve the transaction records for the selected CAP and add the values for each record to the running total of the appropriate Product Number. As transactions are retrieved, check that the records contain Primary Mappings, a valid Mode, a valid transaction start/end date, that all transactions in the current session have the same mode and that the session balances to 0.00;
- Retrieve the records containing the rollover records at the end of the selected CAP;
- Compare the values accumulated for each product from the opening stock and the transaction records against the records recorded at rollover;
- Where the comparison identifies a difference, allow the user to print the retrieved and calculated Product Totals and list the transactions for that Stock Unit/Product in the selected CAP.
- If no errors have been detected in the Stock Unit Rollover records, retrieve the office-level rollover records (Container:##) for the selected CAP and compare the Office Rollover totals for each product with the Stock Unit rollover totals for each product;
- Where the comparison identifies a difference, allow the user to print the retrieved and calculated Product Totals and list the Stock Unit and Container:## records for the identified Product in the selected CAP.
- If no errors have been detected in the Container:## records, retrieve the CAP Summary records for the selected CAP and compare the values for each Cash Account Mapping Node with the value of the Container:## product values. In order to achieve this, each Container:## product total has to be converted into a value for the appropriate Cash Account Mapping Node;
- Where the comparison identifies a difference, allow the user to print the retrieved Cash Account Mapping Node value and list the Container:## records for the identified Node in the selected CAP;
- If no errors have been detected, retrieve the Cash Account Line records for the selected CAP and compare the value of each line with the values contained in the CAP Summary records. To achieve this, each of the CAP Summary records has to have its 'sign' adjusted to ensure that values are being correctly compared (as occurs when the Cash Account is printed).
- Where the comparison identifies a difference, allow the user to print the retrieved Cash Account Line value and list the CAP Summary records for the identified Node in the selected CAP.

The use of this automated approach to checking the accuracy of the records in the message store has been proven to identify approximately 99.9% of Cash Account imbalances reported from Live Outlets.

FUJ00079425

8.5.3 NT Event Log

The Event Log for the counter from which a potential error has occurred can provide important information in support of the message store, however it is in support of and does not replace the message store. A complete definition of EPOSS NT Events logged is defined in Appendix A.

8.5.4 System Fault Notifications

The notification of a system fault to the outlet operator is expected to result in a call to the Horizon Helpdesk. The relaying of all the information defined on the reported notification is all that is required to pursue the investigation further. A complete definition of all EPOSS System Fault Notifications is provided in the EPOSS System Fault Catalogue (see Reference [10]).

8.5.5 EPOSS Audit Files

The EPOSS Audit files for the counter from which a potential error has occurred can provide important information in support of the message store, however it is in support of and does not replace the message store. A complete definition of all potential EPOSS Audit File entries is provided in Appendix B, whilst an example of the file content is provided in Appendix C.

Of specific significance is the fact that the Audit Files provide a record of the software versions being executed by the EPOSS Product, though primarily the use of the audit files is to record diagnostics in support of particular PinICLs. The following sections provide an explanation of the formatting of Audit File entries.

8.5.5.1 Audit File Date

The first entry in any Audit File will be the date for which Audit entries are recorded in the file. The entry is used by the EPOSS Product to determine if a new audit file should be written for the day in question.

8.5.5.2 EPOSS Product Software Versions

The second set of entries in any Audit File will normally be a record for each EPOSS Product software application initialised at desktop start up. Each entry records the details about the software components being executed in the service.

Ordinarily the audit files are only used by the EPOSS In-Day Service, and being initialised as part of the Riposte desktop start up around 0300, each application that has initialised will be recorded. Should the desktop be rebooted in the course of a day then there will be a subsequent set of entries. Equally if the desktop runs over midnight of the following day then the first entries in the audit file for the subsequent day will not be the EPOSS Product Software Versions.

The auditing of EPOSS Product Application Versions in-service as recorded in the audit files allows definitively for the software on the counter to be reconciled against any work package dependencies that exist.

FUJ00079425

The following table defines the format of each entry recording the Product Software Versions:

Field	Description
<Time>	Time message recorded to log
<Application Name>	The desktop Application Name defined by the DLL and known to Riposte as the Desktop App
(<DLL Name>)	The name of the component DLL
Version <string>	The EPOSS Development Team unique identifying reference for the component DLL being executed. This should not be confused with the PVCS Version. The reference is associated with the properties of the DLL queried by 'right clicking the DLL.
[<work package details>]	EPOSS Development Work Package Details from which this component was sourced. If null the DLL was handed over prior to the mechanism being established.

An example message would be:

16:02:14 BusinessObject (businessobject) Version 14.1.320 [WP7727 Build Tue Mar 14 10:27:32 2000(GMT)]

It should be noted that not all EPOSS Product components currently register themselves in the Audit Files. A change will be raised to do so. It should also be noted that the EPOSS Audit Files mechanism is also used by the OBCS Counter Product.

8.5.5.3 Audit File Detail Entries

All other entries recorded in the audit files are of limited structured format and reflect bespoke needs to gather information. However the generic format of these messages is as follows:

Field	Description
<Time>	Time message recorded to log
<Application Code>: :	An internal mnemonic denoting the DLL recording the entry
<DLL Class>	The class and function name recording the entry, within the DLL
<Diagnostic String>	Bespoke diagnostic details

An example message would be:

```
18:27:59        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance -
Office Copy] the executed query returned [0] records.
```

It should be noted that a modification is being considered to this structure, so that the PinICL for which diagnostics are being recorded is identified. That way it may be explicitly identified that a PinICL instance has occurred again.

FUJ00079425

9 Troubleshooting EPOSS

Whilst the ability to support EPOSS generically is provided for by this document there are specific instances of use that may generate support needs. In all cases these problems are likely to have been caused by a misunderstanding of use and/or incorrect reference data definition.

During the life of the steady state Horizon system incidents will be raised on a daily basis describing issues perceived either at the Post Office Outlets, the Pathway Data Centres, the POCL Accounting HQ (including the POCL TIP system), the POCL AP Host Data Centre or the Pathway Customer Service Centres. Where investigation by 1st, 2nd and 3rd line support indicates that there may be a fault in the Horizon software, or where these support organisations have been unable to explain the cause of the incident, the call will be passed to the QFP 'stack' within the PinICL Fault Management database.

The QFP stack is monitored on a daily basis and the calls are assigned to members of the QFP group according to which area of the system appears to be at fault. Members of the group then conduct their analysis of the call and either return the call to 3rd line support with a description of why the call does not identify a software fault or issue, or pass the call to one of the Delivery Units for rectification of an identified issue or further analysis if the cause cannot be determined.

Analysis of all incidents raised in the Live system should be based on examination of the Riposte Message store data recorded at the outlet. Wherever possible the evidence provided by 3rd Line Support should consist of a complete extract of the message store for an outlet so that in the event of a software issue being identified, the Delivery Unit concerned shall have the opportunity to recreate the problem from the evidence supplied in order to better understand the technical cause of the incident.

In addition to message store the evidence should also include the NT Event log from the counter at which the incident arose and any 'log' files of diagnostic information that may be created from time to time by the counter software should also be supplied as evidence. Log files are normally created in the 'c:\Temp' directory on each of the counter systems at an outlet.

It should be noted that users may encounter a number of other log files in the c:\counters\bin\ directory. These log files are PSStandardLog, PSClient and PSServer. These log files are owned by Escher are out of the scope of EPOSS.

EPOSS Incidents raised in the Live System generally fall into the following categories:

- Operational problems (e.g. user 01sent to the POCL TIP system is rejected because the file contains data which fails validation)

Each of these categories is dealt with in further detail in the following sub-sections.

FUJ00079425

9.1 Operational Problems

9.1.1 Log On/Log Off Issues

9.1.1.1 General

These type of issues are generally caused by a failure of the software (either EPOSS or the underlying Riposte Desktop) to correctly update Persistent Objects (see Section nn) in the Message Store which define the current state of the user account.

At logon the Riposte software validates the user's ID and Password with the NT user account and either accepts or rejects the logon. If the logon is successful, the Riposte software updates the RiposteUserLogons collection, object name [userid] as indicated below:

If the logon attempt is unsuccessful then a failed logon attempt is record by the Riposte Software by updating the RiposteUserLogons collection, objectname [UserId] as indicated below:

<Message:<GroupId:465329><Id:1><Num:198805><Date:15-Jun-2000><Time:07:38:40><User:MHOW01><Expiry:34><TranStartNum:198804><Collection:RiposteUserLogons><ObjectName:MHOW01><LastLogon:<Id:1><Date:15-Jun-2000><Time:07:38:40><Mark:<1:198803><31:336><32:97839><33:15953><60:29><61:4><62:3668><63:10>>><Security:<ACL:Riposte>><Version:513><CRC:E4FF6055>>

Following a successful Riposte Logon the EPOSS application is notified of the logon event. At this point the EPOSS software updates the EPOSSUsers collection, ObjectName:[User ID] to record the date and time of the logon. If at this point the user was previously attached to a stock unit then the object will indicate the stock unit to which the user is attached. If the user was not previously attached to a stock unit then the EPOSS software will warn the user and then attach the user to the Default (DEF) stock unit. This will be recorded in the EPOSSUsers collection object. Examples of updated objects for a user attached to a stock unit and a user attached to the default are as follows:

<Message:<GroupId:465329><Id:1><Num:198810><Date:15-Jun-2000><Time:07:39:17><User:MHOW01><Expiry:34><Collection:EPOSSUsers><ObjectName:MHOW01><Data:<LoggedOnAt:15-Jun-2000 08:39:17><Container:AA>><Version:2022><CRC:8596975F>>

<Message:<GroupId:465329><Id:1><Num:198810><Date:15-Jun-2000><Time:07:39:17><User:MHOW01><Expiry:34><Collection:EPOSSUsers><ObjectName:MHOW01><Data:<LoggedOnAt:15-Jun-2000 08:39:17><Container:DEF>><Version:2022><CRC:8596975F>>

<Message:<GroupId:465329><Id:1><Num:198812><Date:15-Jun-2000><Time:07:39:18><User:MHOW01><Expiry:34><Collection:APSLogon><ObjectName:APSLogonState1465329><State:Logon><UserName:MHOW01><SU:AA><Mark:<Mark:<1:198811><31:336><32:97839><33:15953><60:29><61:4><62:3668><63:10>>><Version:566><CRC:473D4526>>

<Message:<GroupId:465329><Id:1><Num:198813><Date:15-Jun-2000><Time:07:39:19><User:MHOW01><Expiry:35><Application:OBCS><TranType:Admin><Tag:<v:6><C:4>><Data:<StartTime:07:39:19><STF:1><EndTime:07:39:19><ETF:1><Stat

FUJ00079425

e:5><IOP_ident:><ForeignIndicator:0><ForeignEnquiryTime:><ForeignEnquiryTimeOut:><Vouchers:0><EntryMethod:><Result:1><NC:0>><CRC:1B96E4B0>>

<Message:<GroupId:465329><Id:1><Num:198814><Date:15-Jun-2000><Time:07:39:19><User:MHOW01><Expiry:34><Collection:OBCSStartup><ObjectName:OBCSNull><Data:<Today:06152000>><Version:781><CRC:7373B3C8>>

<Message:<GroupId:465329><Id:1><Num:198815><Date:15-Jun-2000><Time:07:40:24><User:MHOW01><Expiry:34><Collection:StockUnits><ObjectName:AA><Data:<Shared:True><Locked:False><LockedBy:><CAP:13><BP:1><NextCAP:14><StartDate:15/06/2000><EndDate:21/06/2000 23:59:59><Balancing:False><BalanceStatus:Dirty><RolloverTrailer:<GroupId:465329><Id:1><Num:198239>><CAPRolloverTrailer:<GroupId:465329><Id:1><Num:198239>>><Version:711><CRC:20DD354C>>

9.1.1.2 Probable Causes

Failure to be able to log on to the system is usually associated with:

- The user is logged on already at another terminal and the software on that terminal is refusing to allow the session to be transferred because there is a critical process currently underway.

- The user is attached to a shared stock unit and another user is in the process of balancing the stock unit

Assuming that the actions described in any existing Known Error Log(s) (KELs) have been followed, the message store should be examined to determine whether the user was genuinely logged on already at another terminal and whether the system still had an extant StopDeskTransfer object in force at the time of the logon or whether another user was in the process of balancing at the time the logon was attempted. If neither of these apply then the call should be routed to the POCL Infrastructure Delivery Unit (EPOSS Development team) to investigate why the problem occurred.

It should be noted that the StopDeskTransfer events are no longer indicated by the presence of the USERLOCKREQUEST objects and these messages are not logged into the message store.

Following the creation of the first of these objects the user MHOW01 would not be allowed to log on at any other terminal in the outlet until the second (unlock) update is written to the message store. If a log on was attempted between the two updates, the user would be shown the message contained in the 'TransferLock' object (i.e. "Cannot perform session transfer during logon checks").

9.1.2 Stock Unit Attachment Issues

9.1.2.1 General

Attachment of a user to a stock unit is a pre-requisite for accessing many of the system functions, particularly those associated with serving customers and carrying out other actions which are associated with accounting activity.

FUJ00079425

When a user account is first created on the system the user is not associated with any stock unit. The first time that the user logs on a warning is given that the user is not attached to a stock unit and the system assigns the user to the 'Default' stock unit (DEF). While the user is assigned to the default stock unit, no accounting transactions can be performed. In order to attach the user to a stock unit either the current user or another user in the office must use the Administration functions to identify the user and the stock unit to which they are to be attached.

Stock Units which have been created as 'Individual' stock units may have only 1 user attached to the Stock Unit at any one time. If there is a need to attach another user to the stock unit then the current user of the stock unit must first be attached either to the Default stock unit or any other stock unit in the office. 'Shared' stock units may have any number of users attached at any one time.

9.1.2.2 Probable Causes

Most issues associated with failure to attach a User to a Stock Unit are related to:

- The Stock Unit is an individual Stock Unit and still has another user attached
- The stock Unit is currently being balanced by another user

In the case of the individual stock unit the existing user must be assigned to a different stock unit before the attachment can be made. This can only be done if the current assigned user is not currently logged on to the system. An individual stock unit object contains the attribute 'Shared' set to 'False', as in the following example:

<Message:<GroupId:465329><Id:1><Num:198815><Date:15-Jun-2000><Time:07:40:24><User:MHOW01><Expiry:34><Collection:StockUnits><ObjectName:AA><Data:<Shared:False><Locked:False><LockedBy:><CAP:13><BP:1><NextCAP:14><StartDate:15/06/2000><EndDate:21/06/2000 23:59:59><Balancing:True><BalanceStatus:Dirty><RolloverTrailer:<GroupId:465329><Id:1><Num:198239>><CAPRolloverTrailer:<GroupId:465329><Id:1><Num:198239>>><Version:711><CRC:20DD354C>>

In the case of the shared stock unit, a user can be attached to the stock unit at any time except when the stock unit is in the process of being balanced by another user. The fact that a stock unit is in the process of being balanced is signified by the 'Balancing:' attribute within the StockUnits Collection for the stock unit being set to 'True' (see above example). On occasions this state can be left set to true even though no user is actually carrying out the balance. This usually occurs if the user who was balancing has exited from the system abnormally (e.g. the PC has been powered off). In these circumstances the application has not been able to update the StockUnits object to reflect the fact that balancing is no longer taking place. While in this state, no user of the stock unit is allowed to logon at any of the other terminals in the outlet until a user of the stock unit has logged back on at the terminal which was originally carrying out the balancing activity.

In the event that there is confusion in the office as to who was balancing where, it is possible for the 3rd Line Support staff to update the StockUnits object in the office to change the balancing status to 'False', allowing users to then carry on with any logon

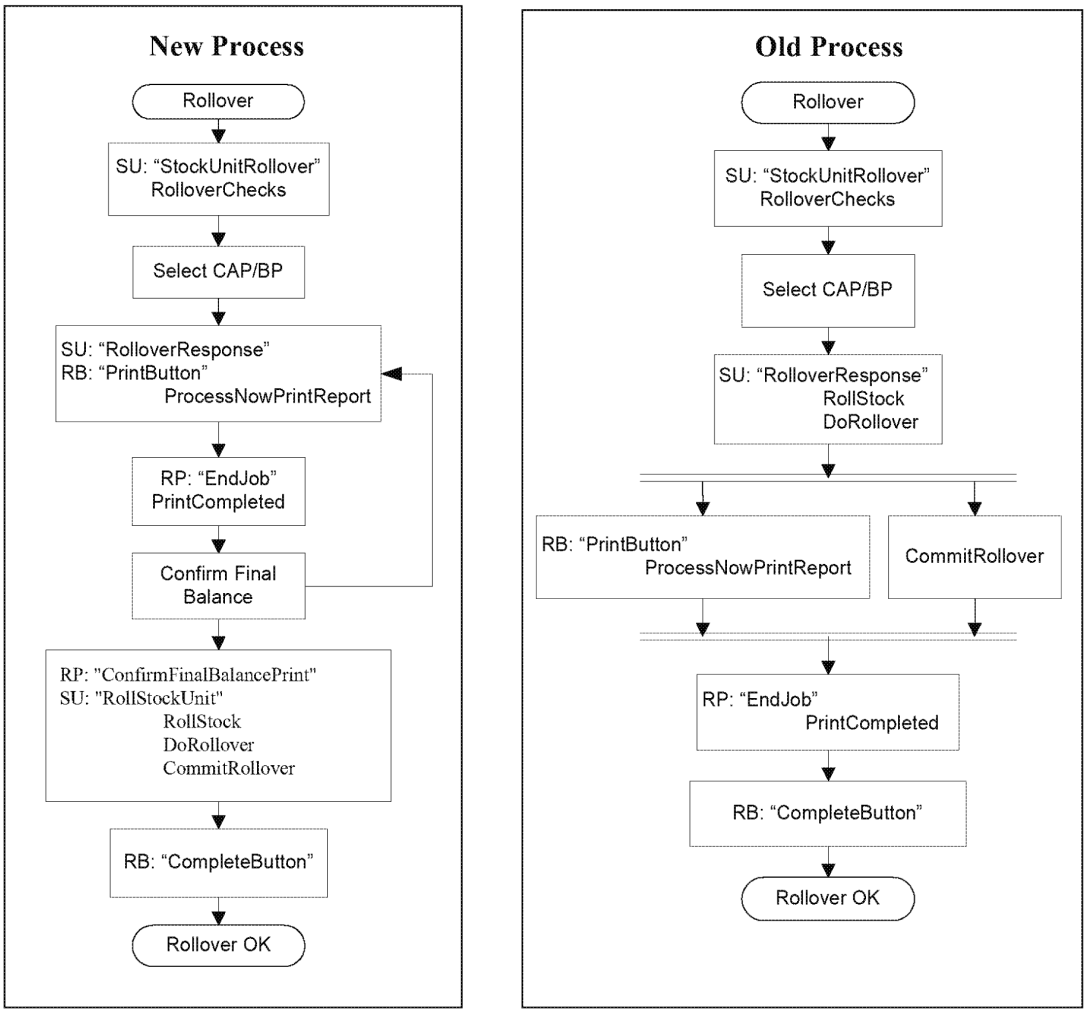
FUJ00079425

or attachment activity. This must, however, be done with care to ensure that the lock is not removed when balancing is genuinely taking place.

9.1.3 Stock Unit Rollover Process Confusion

9.1.3.1 Process Outline

The outline of the rollover processes is shown in the figure below, showing the enhancement at CI4. The major difference was that in the old process, the printing of the report was carried out in the background.



Abbreviations
RB: Report Broker
SU: Stock Unit

Figure 50 - A Comparison Between

FUJ00079425

9.1.3.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

At CI4 the EPOSS Product has been enhanced to change the Stock Unit Rollover Process. It is anticipated that this change may result in some operator confusion until familiarity with the new process is achieved. The change is in two parts:

- Additional confirmation stage has been applied to the rollover process, equivalent to that already present in the cash account rollover.
- The performance of the Stock Unit Rollover Process has been improved.

9.1.3.3 System Recovery Process

No recovery action is required however the operator may need 'coaching' through the revised process.

9.1.4 Product Mode Version not Populated Correctly

9.1.4.1 Failure Symptoms

- TIP raise an incident identifying that a product mode version has not been received
- TIP raise an incident identifying that an incorrect product mode version has been received

9.1.4.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

At CI4 the EPOSS Product has been enhanced so that given permission has been given to transact a product in a mode, then the Product Mode Version defined for that reference data is recorded on the transaction.

Situations arising where this has not been undertaken correctly do not affect the outlets day to day business and only affect TIP.

Diagnosis of the fault requires a number of factors to be considered:

- Establish if the transaction was recorded before the outlet migrated to CI4, as before CI4 no Product Mode Version will have been recorded. It is possible for a transaction to be recorded prior to CI4 but not be harvested until after the outlet migrated. As a result TIP will not received the product mode version. This is not an error in EPOSS and must be catered for by TIP.
- Establish if the Product Mode Version with which the transaction has been transacted reflects the reference data delivered to the counter. If it does then the fault lies in the reference data.
- Otherwise the fault may lie in the software and a fault should be raised.

A reference data fault may be confirmed by examining Reference Data relationships as follows:

FUJ00079425

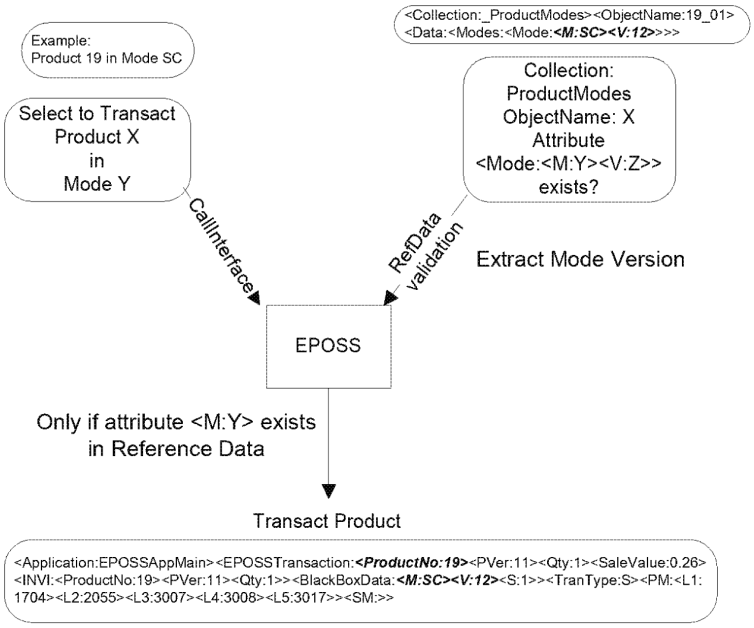


Figure 51 - The Validation of Product Mode: Incorrect Product Mode Version

9.1.4.3 System Recovery Process

There is no system recovery action on the counter as a result of this problem.

9.1.5 Product unavailable in Mode

9.1.5.1 Failure Symptoms

User of EPOSS In-Day Service received a tablet of the following nature:

FUJ00079425

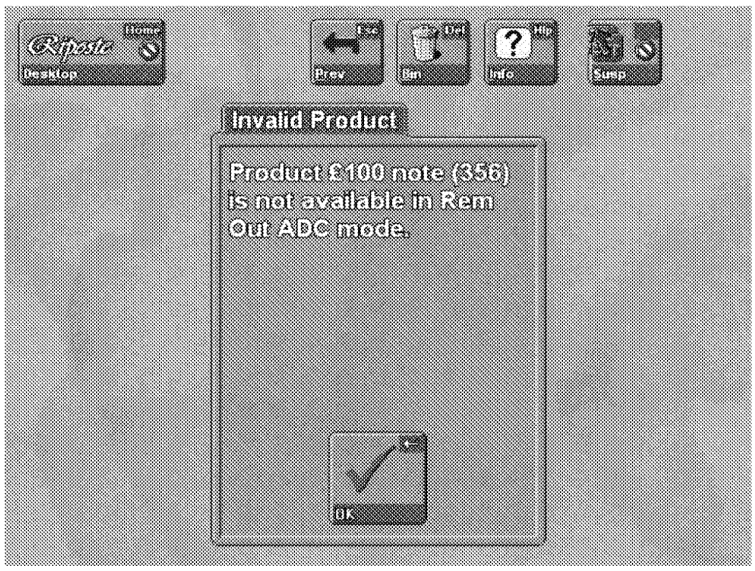


Figure 52 - Error Tablet showing Invalid Product-Mode Combination

9.1.5.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

At CI4 the EPOSS Product has been enhanced so that permission to transact a product in a particular mode within an outlet is governed by Reference Data provided by POCL.

The Reference Data dictates that each product available to be transacted at an outlet, the modes in which it can be transacted are defined. This data is checked as part of the EPOSS Transaction Service and the transaction only allowed if the product contains the mode in which the transaction is being attempted.

If the mode is not available then the impact is such that the product requested cannot be transacted in the defined mode. There is no circumvention to this problem other than to undertake an adjustment to the outlet accounting figures by another means, for example adjusting the relevant stock accordingly (if available).

The fault may be diagnosed and confirmed by examining Reference Data relationships as follows:

FUJ00079425

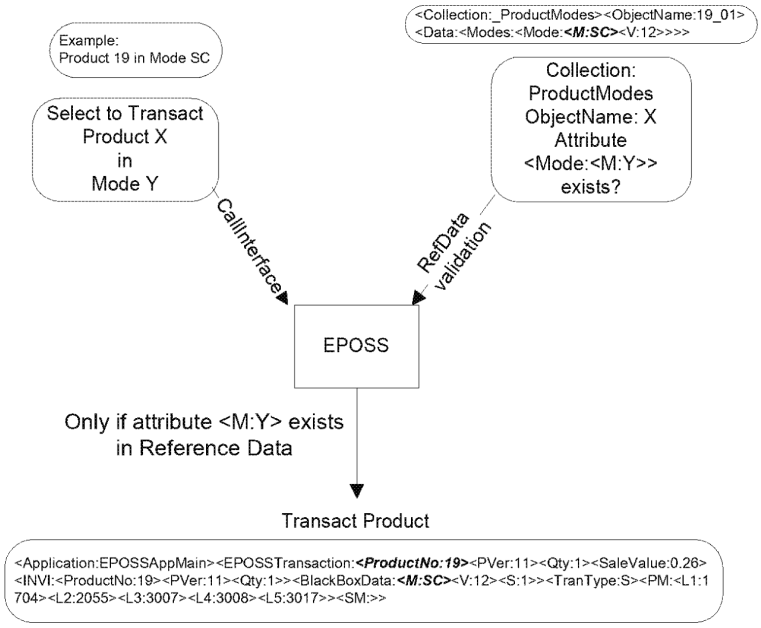


Figure 53 - The Validation of Product Mode: Unavailable Product

9.1.5.3 System Recovery Process

Referral of this problem must be made to POCL for correction of the reference data or contact to be made with the office as to why they are attempting to transact a product to which they are not entitled.

9.1.6 LFS Inventory Item not Recorded in Transaction

9.1.6.1 Failure Symptoms

SAPADS report that they have not received transaction information about products (Inventory Products) which they control.

9.1.6.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

At CI4 the EPOSS Product has been enhanced so that Products identified as Inventory Items in Reference Data, when transacted, have additional LFS attributes recorded in the transaction.

Situations arising where this has not been undertaken correctly do not affect the outlets day to day business and only affect TIP and SAPADS.

Diagnosis of the fault requires a number of factors to be considered:

- Establish if the transaction was recorded before the outlet migrated to CI4, as before CI4 no LFS attributes will have been recorded. It is possible for a transaction to be recorded prior to CI4 but not be harvested until after the outlet

migrated. As a result TIP and SAPADS will not received the additional information. This is not an error in EPOSS and must be catered for by SAPADS.

- Establish if the Inventory Item with which the transaction has been transacted reflects the reference data delivered to the counter. If it does then the fault lies in the reference data.
- Otherwise the fault may lie in the software and a fault should be raised.

A reference data fault may be confirmed by examining Reference Data relationships as follows:

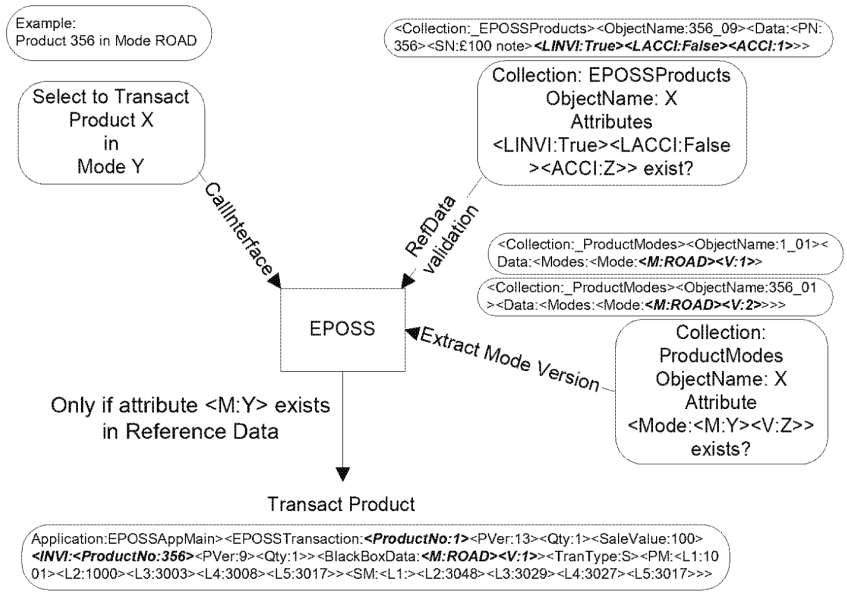


Figure 54 - Interactions for LFS Inventory Items

9.1.6.3 System Recovery Process

There is no system recovery action on the counter as a result of this problem. Further support information about LFS may be found in [20].

9.1.7 Non-Value Stock Problems

9.1.7.1 Failure Symptoms

Whilst the EPOSS Stock Unit Rollover and Office Rollover Processes have been changed to engage checks of LFS Non Value Stock Declarations and :Confirmations having taken place respectively, the actual processes to declare non value stock and confirm non value stock declarations, themselves are processes supported by the LFS Counter Product.

Any problems performing the Declaration and Confirmation of Declarations of Non Value Stock may require support intervention.

FUJ00079425

9.1.7.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

The inability to declare non value stock will prohibit the ability balance the stock unit and hence prohibit the ability to rollover the office. Fault Diagnosis of problems therein should be addressed to LFS Product Support.

The inability to confirm the declarations does not prohibit the office being rolled.

9.1.7.3 System Recovery Process

The inability to roll a stock unit cannot be avoided. It may be possible to amend the reference data that forces the stock unit rollover to be a mandatory process, so that it becomes optional, but this will require a change to data sourced from RDMC.

9.1.8 Non-Value Stock Not Declared

9.1.8.1 Failure Symptoms

User of EPOSS In-Day Service received tablets of the following nature:

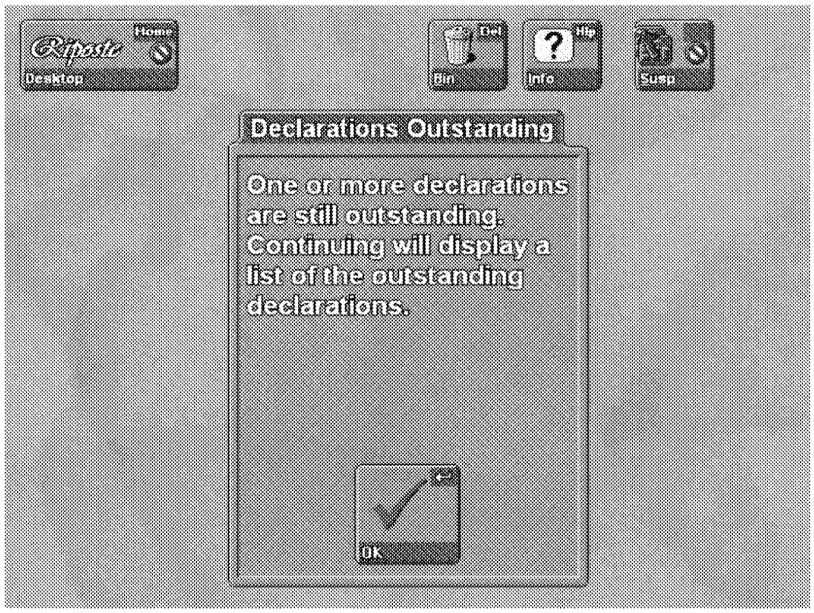


Figure 55 - Message Tablet Alerting User to Outstanding Declarations

FUJ00079425

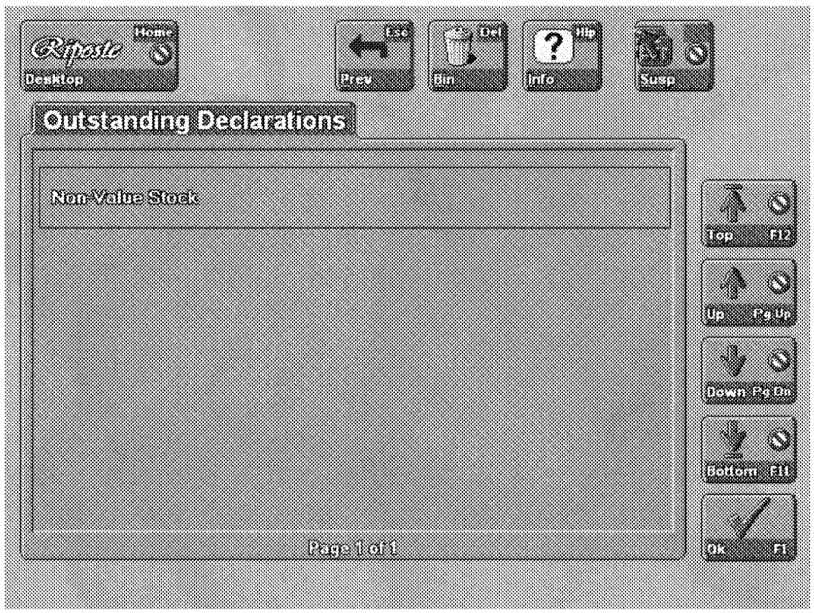


Figure 56 - List of Outstanding Declarations

9.1.8.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

At CI4 the EPOSS Product has been enhanced to change the Stock Unit Rollover Procedure. It is anticipated that this change may result in some operator confusion. The change results from the introduction of LFS at CI4 and introduces an additional mandatory declaration stage – the declaration of Non Value Stock. Failure of the operator to carry out this declaration for each stock unit prior to commencing the stock unit rollover process will result in the error described.

As a result the outlet will not be able to rollover the stock unit.

9.1.8.3 System Recovery Process

No recovery action is required however the operator may need 'coaching' through the revised process.

It should be noted that the mandatory nature of this declaration is driven by reference data. Should it be decided that the mandatory aspect be removed then this can be achieved by a change to reference data.

9.1.9 Non-Value Stock Not Confirmed

9.1.9.1 Failure Symptoms

User of EPOSS In-Day Service received a tablet of the following nature:

FUJ00079425

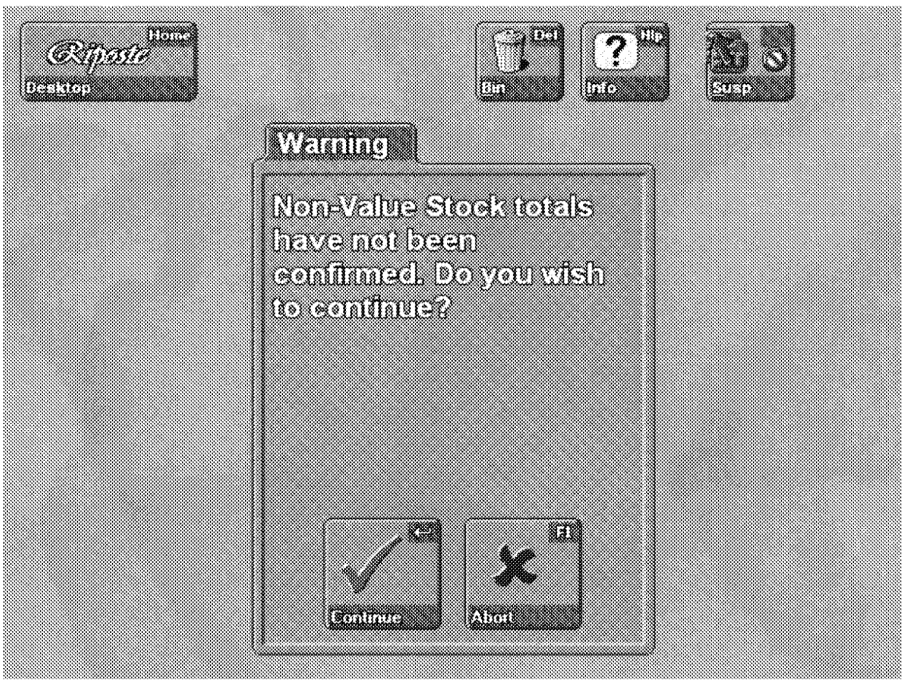


Figure 57 - Confirmation of No-Declarations for Non-Value Stock

9.1.9.2 Operational Impact of Failure and Fault Diagnosis and Confirmation

In addition to the change to the Stock Unit Rollover process at CI4, the EPOSS Product has also been enhanced to change the Office Rollover Procedure. It is anticipated that this change may also result in some operator confusion. The change results from the introduction of LFS at CI4 and introduces an additional confirmation stage – the confirmation of Non Value Stock Declarations. Failure of the operator to carry out this confirmation prior to commencing the cash account production stage, will result in the warning described.

The warning does not prohibit the office being rolled over.

9.1.9.3 System Recovery Process

If the operator so wishes, they can abandon the cash account production and exit, in order to confirm the declarations.

9.2 Reporting Issues

9.2.1 Report Cut-Off Issues

9.2.1.1 General

The full set of reports for the Horizon system are detailed in the Horizon Reports and Receipts Specification (SD/DES/005). Reports fall into categories of either 'Cut-Off'

FUJ00079425

reports or period reports. Cut-Off reports are outputs which ensure that once produced any further run of the report will exclude previously reported transactions. These are typically used to allow the user to produce a daily client report of transactions undertaken since the last report was sent to the client. Reporting issues are usually associated with the accuracy of the objects recorded in the message store to control the number of transaction messages which are to be included in the report, or with the way in which values are accumulated on the report.

9.2.1.2 Probable Causes

Where a report is clearly providing inaccurate volume or value information, the issue should be passed to the POCL Infrastructure Delivery Unit (EPOSS Development Team) to establish the precise cause of the failure.

In the case of 'Cut-Off' reports, the cut-off markers for the report at issue should be examined to determine whether the values reflected in the report have been accurately represented. To establish this the support analyst should:

- Identify the Object Name used for the report in question. The Object name will consist of the Stock Unit identifier of the current stock unit followed by the EPOSS Report number (e.g. <Collection:CutOffs><ObjectName:AA20> refers to the last Giro Deposit report 'cutoff' for stock unit AA).
- Locate in the message store the update to this object on the counter and at the time reported in the incident. The message should be similar to the following example:

```
<Message:<GroupId:465329><Id:1><Num:199002><Date:15-Jun-
2000><Time:15:58:12><User:MHOW01><Expiry:34><Collection:CutOffs><ObjectName:AA2
0><TideMark:<Data:<DateTime:15/06/2000
15:57:07><Seq:1198993>>><Mark:<1:198996><31:337><32:97839><33:15953><60:29><6
1:4><62:3668><63:10>><PreviousMark:<Mark:<1:197945><31:336><32:97839><33:15783>
<60:29><61:4><62:3668><63:10>>><PrevMarkPtr:197951><Version:322><CRC:5971E888
>>
```

- Identify from the object the Riposte Message Server 'Mark' that indicates the last message number on each node to be included in the report
- Identify from the object the content of the previous cut-off marker (attribute 'PreviousMark', if there was one) and obtain for each node from the previous marker the last message numbers to have been included in the previous report. If there was no 'Previous' cut-off then the start marker for the report will be the current stock unit balance period marker.
- Using the information from the markers, extract from the message store all the EPOSSTransaction records for the required Product Numbers (or more easily for the specified report node) between the start and end points. The reporting node can be determined from the cut-off object name that will relate to a node number recorded in the Primary Mappings attributes of transaction messages. The level of the required node is defined in the report definition in the Horizon Reports and Receipts Specification (SD/DES/005). For example, the Giro Deposit report is Report Number 20. All transactions contributing to the report will have a Level 2

FUJ00079425

node of 20, therefore all transactions between the lower and upper markers for the report and which have a level 2 Primary Mapping of 20 (EPOSSTransaction.PM.L2 = 20) should be included in the report.

9.3 Accounting Issues

Accounting issues are generally associated with a failure to balance either the Stock Unit or the Office. These incidents are generally reported as a difference between the total value of the Receipts Table and the Payments Table of the outlet Cash Account. In a balanced office, the Receipt Table total and the Payment Table total should always be the same value.

9.3.1 General

A 'Balance' (either at Stock Unit or Cash Account level) is struck through the following process:

The **'Receipt'** total is calculated from the sum of:

- The total value of Cash and Stock on hand at the start of the period plus the net value of any discrepancies in the previous period
- The total value of 'Receipt' (non Value Stock) transactions in the period
- The total value of Remittances In of Value Stock in the period
- The total value of Transfers In of Value Stock in the period
- The total value of any Revaluations 'Up'
- The **'Payment'** total is calculated from the sum of:
- The total value of Cash and Stock on hand at the end of the period
- The net value of any discrepancies in the stock unit
- The total value of 'Payment' (non Value Stock) transactions in the period
- The total value of Remittances Out of Value Stock in the period
- The total value of Transfers Out of Value Stock in the period
- The total value of any Revaluations 'Down'

9.3.2 Stock Unit Accounting

Discrepancies in the stock unit are calculated at the time that the stock unit is balanced. As part of the balancing process the user is required to 'Declare' to the system the amount of Cash held in the stock unit at that time. This declaration is a mandatory step in the balancing process. Once the declaration has been made, the system compares the value declared with the value of Cash that user should have had based on the transactions for the current period in the message store. Any difference between the system and declared values results in a discrepancy being generated (either as a loss or a gain) which is compensated for by an increase/decrease in the

FUJ00079425

value of Cash. For example, if the user declares that there is £100.00 of Cash physically in the drawer but the system calculates that there should be £110.00 from the recorded transactions then the current cash value in the message store is reduced by £10 to bring it into line with the declared value and a discrepancy is generated for £10 as a 'Loss'.

A similar process occurs with non-Cash declarations (Stamps - Mandatory or Stock – Optional). In these cases, when the system calculated stock levels of the declared items are compared with the declared values, any differences are recorded as an adjustment to the stock item (recorded with a mode of either 'DDN' (Declaration Discrepancy Negative) or 'DDP' (Declaration Discrepancy Positive). A compensating adjustment is then recorded against 'Cash' in order to maintain the balance. As a result of this, the net total of discrepancy values (including any discrepancy value rolled forward from the previous period) ends up as a net adjustment to 'Cash', which in turn results in a new net discrepancy total for the period when the Cash declaration is compared with the system calculated Cash value (see above). Only the final net value of any discrepancies in the stock unit is carried forward to the next balancing period as part of the Cash and Stock rollover records.

At Stock Unit level the values of Cash, Stock and Discrepancy items which are carried forward to the next balancing period are recorded in a set of 'rollover' records during the stock unit balancing process. These records are distinguished by having an <OpeningFiguresId:> attribute with a unique value which is then recorded in the OpeningFigures Trailer record written after the transactions. The message identity of the OpeningFigures Trailer record is written in the Rollover Trailer record, the latest of which is pointed to by an attribute in the Stock Unit object of the relevant stock unit. It is possible, therefore, to navigate back through the message store using these pointers to identify the appropriate <OpeningFiguresId:> attribute for any given Cash Account or Balance Period.

FUJ00079425

The Stock Unit 'Rollover' records also contain a summary, by Product Number, of the value of all non value stock transactions, remittances and transfers undertaken during the balancing period. These records are distinguished by having an <OpeningFiguresId:> attribute with a unique value which is then recorded in the RemProducts Trailer record written after the transactions. The message identitiy of the RemProducts Trailer record is written in the Rollover Trailer record, the latest of which is pointed to by an attribute in the Stock Unit object of the relevant stock unit. It is possible, therefore, to navigate back through the message store using these pointers to identify the appropriate <OpeningFiguresId:> attribute for any given Cash Account or Balance Period.

9.3.3 Office Accounting

At the Office level, the outlet accounts have to include not only the details of the individual stock unit balances and transactions but also what is referred to as the 'Suspense Account'. The Suspense Account is represented by the three tables on the front page of the Cash Account – Tables 2, 2a (known as the 'Unclaimed Payments' (UP) tables) and Table 3 (known as 'Uncharged Receipts' (UR) table).

The Suspense Account is used to record details of values which cannot, for some reason, be properly brought to account in the current period. This may be because of an error in an earlier accounting period which is awaiting resolution (e.g. a 'bounced' cheque) or because there is nowhere else available on the Cash Account to report the value (e.g. if the office is in possession of a 'POCL Cheque' for payment of benefits to a Nursing/Residential Home). Once a value has been placed in the 'Suspense Account' it ceases to be accounted for at stock unit level and appears only on the Cash Account in the appropriate Table.

In order to place a value in the Suspense Account, POCL have provide a number of 'Products' which the user can select from the menu hierarchy to achieve the desired result. These products are provided as 'pairs' of products – one to move a value INTO the Suspense Account and one to move a value OUT of the Suspense Account. These transactions to move values in and out are undertaken from any of the stock units in the office and, as in the case of discrepancies, are balanced by an equal and opposite adjustment to Cash. At the end of the balancing period the value of MOVEMENTS to and from the Suspense Account are reported as part of the current stock unit balance but the current running total of the Suspense Account lines is only ever reported on the Cash Account.

The way in which the system achieves the maintenance of the running total for the Suspense Account is that during the Cash Account preparation phase all the current stock unit balance records plus any Suspense Account records from the end of the last Cash Account Period are accumulated in a set of records identified as belonging to the '##' stock unit ('##' is the designation for the Office level stock unit). During this process any transaction records representing movements to and from the Suspense Account (in either this or any previous CAP) are accumulated by Product Number and are given an additional attribute '<SuspenseContainer:$$>'. These records are then included in the office level rollover records so that they are carried forward to the Cash Account preparation process in the next CAP.

FUJ00079425

For Office level accounting, at the time that the Cash Account is produced the system locates all the latest Stock Unit rollover records for the current Office CAP and copies them to a new set of messages recorded against a 'system' stock unit identified as Stock Unit ##. The system first copies all the Value Stock rollover records from each stock unit and then all the non Value Stock, Remittance and Transfer totals for each stock unit followed, finally by the Suspense Account values which are the net values of the Suspense Account totals from the Container:## records at the end of the previous Cash Account Period plus any new totals recorded in the stock unit rollover records for the current CAP. Each stock unit's set of both the Value Stock and non Value Stock totals recorded in the Container:## records are separately grouped and the OpeningFigures or RemProducts Trailer record identifies the stock unit to which the previous set of records relate. As with the stock unit rollover records, both the Value Stock and the non Value Stock sets of records are identified by having an <OpeningFiguresId:> attribute with a unique value which is then recorded in the OpeningFigures or RemProducts Trailer record written after the transactions. The message identity of the Opening Figures or RemProducts Trailer record is written in the Rollover Trailer record, the latest of which is pointed to by an attribute in the EPOSSCAP 'Office' object. It is possible, therefore, to navigate back through the message store using these pointers to identify the appropriate <OpeningFiguresId:> attribute for any given Cash Account.

Once the Container:## records have been written, the Cash Account process then takes each of the ## records and determines the Cash Account Mapping reference appropriate to each record, based on the values contained in the Primary Mapping or Secondary Mapping attributes of each record. These values are then written to the message store, again with a unique 'OpeningFiguresId' attribute which is written in the CAP Trailer record (as the 'OfficeUniqueId' attribute) at the end of the set of records. The identity of the Cash Account Trailer record is then recorded in the EPOSSCAP 'Office' object.

The next stage of the Cash Account production process takes the CAP Summary records (those identified by the 'OfficeUniqueId') and prints the Trial Cash Account report. As each line of the report is printed a record of the line is written to the message store, again with a unique reference recorded in the 'CashAccountId' attribute. The unique 'CashAccountId is then recorded in the Cash Account Trailer record written after the Cash Account Line records and the identity of this message is stored in the EPOSSCAP 'Office' object.

The final stage of the process follows the user's selection of the CAP Rollover option. At this point, the system takes all the Cash, Stock and Discrepancy records from the preceeding Container :## records and generates a new set of records in the message store against a system stock unit '#1'. These records are written as the first set of records in the new Office CAP and represent the opening values of the products in the new CAP for use by the Office Balance Snapshot report.

9.3.4 Probable Causes

9.3.4.1 Migration Discrepancies

FUJ00079425

On initial migration of an outlet (whether the office was previously a 'Manual' office or an ECCO+ automated office) it is possible for the migration data to be flawed. Checks at the point of migration identify whether the values entered would lead to a balanced Cash Account when compared with the value entered for the Office's total Brought Forward value from the previous CAP. Any detected discrepancy is recorded in the 'BBFwdCalc' object in the message store. Any discrepancy recorded in this object will definitely result in a difference between the Receipts and Payments totals on the first Cash Account of at least this amount (other errors may cause the value to be increased or decreased). The content of this object should always be checked first if an accounting error is reported in the first Cash Account after Migration.

9.3.4.2 ECCO+ Migration Issues

There are a number of known issues with the data migrated from the ECCO+ outlets. Data from ECCO+ outlets is read directly from the ECCO+ floppy disks and is then translated into equivalent Horizon data for input to the message store. There are two known issues which can cause a discrepancy in the first Cash Account (over and above the 'Brought Forward' value issue identified at Section 9.3.2 above):

- A stock holding of ECCO+ PLU Number 2 is migrated for one of the stock units. ECCO+ PLU 2 (Notes Held to Order) does not have an equivalent Product in the Horizon system, therefore when this situation occurs the MiECCO migration software migrates this as a value of the defined 'Contingency' Product (currently Cash - Horizon Product 1). For reasons that are not yet understood, this causes an imbalance in the outlet when the first Cash Account is produced. POCL have indicated that there should never be a value migrated for this product since it is no longer used in the ECCO+ system, however at least three instances of this have occurred since the initial rollout.
- Stock Transfer Imbalances. Since the ECCO+ system does not guarantee the integrity of stock transfers it is possible that the migrated data may contain only one half of a complete transfer (a Stock Transfer involves a transfer OUT from one stock unit and a corresponding transfer IN to another stock unit). In these circumstances the value of the transfer will affect the value of stock reported as being in the office at the end of the Cash Account Period and will result in a Cash Account imbalance.

With regard to the first issue, it should be noted that ECCO+ PLU2 relates to an ECCO Product "Notes HTO In". This relates to the cash "Held to Order", which is now defunct. The encounter of a value against this product should be first identified by the HFSO before migration. The usage of this product indicates two problems, firstly, the Office should have disabled this product in the ECCO system. Secondly, even if the product is still enabled, transactions against the product should not have been allowed.

FUJ00079425

9.3.4.3 Missing Primary or Secondary Mappings

If a transaction is recorded in the message store without any values assigned to either the Primary or Primary AND Secondary Mapping attributes then an imbalance at stock unit level (and ultimately Cash Account level) will result.

The Primary Mapping attributes of a transaction define the elements of the accounting node hierarchy to which the transaction value contributes (each transaction has a maximum of 5 levels of node mappings). This mapping will always represent the main 'branch' of the node hierarchy to which a product relates (e.g. the 'Stock' values, the 'Receipt' values or the 'Payment' values). Depending on the mode in which the transaction was undertaken, there may also be Secondary Mappings required. Secondary mappings are generally associated with Value Stock transactions undertaken in a mode other than normal Sale or Adjustment (e.g. a Remittance or Transfer). The Secondary Mapping attributes define the additional elements of the accounting hierarchy to which the transaction value should be accumulated. This is necessary because stock transactions such as a Remittance or Transfer has to affect the current stock holding (achieved through the Primary Mapping) and be balanced by an equivalent value being added to the total value of the Transfers or Remittances (achieved through the Secondary Mapping).

Lack of a Primary Mapping will result in the transaction value not being accumulated at all during the Stock Unit Balancing process. Lack of a Secondary Mapping for a transaction in a mode which requires it will result in the transaction being accumulated for only one side of the Stock Unit Balance.

Primary or Secondary Mappings are also recorded in the stock unit and ## rollover records (See General Description above) in order to satisfy the need for correct accumulation at the Cash Account level. It is clear, therefore, that in the event of a stock unit or ## rollover record being written without the necessary Primary or Secondary mappings a balancing error will occur at the office (Cash Account) level.

9.3.4.4 Mixed Mode Sessions

Transactions at the counter are recorded with a 'Desktop Mode' which was current at the time of the transaction. Whilst the presence or absence of the 'Mode:' attribute does not in itself cause any accounting issue in the outlet (since the subsequent balancing and Cash Account production rely on the Primary and Secondary Mapping attributes), the recording of transactions within a single 'session' with different 'Mode:' attributes may result in an accounting imbalance.

This imbalance would be caused by the 'settlement' product being used for the session being inappropriate for one or more of the transactions in the session. For example, if a session contained a 1st Class Stamp sale in 'Serve Customer' mode (Mode:SC) and a Remittance out of Cash to Other Post Offices (Mode:ROOP) then the settlement transaction for this session would probably use the mode of the last transaction (ROOP). Since the settlement product for Mode:ROOP is a non-POCL product it has no 'balancing' effect on the value in the stock unit, therefore the sale of the 1st Class Stamp would not be balanced by an equal and opposite value of Cash (or other

FUJ00079425

Product) being received. As a result, the stock unit and the Cash Account will mis-balance by this value.

9.3.4.5 Missing Rollover Records

The records created at the various rollover points in the system (Stock Unit Balance, Office Balance, Cash Account Production, and Cash Account Rollover) are generated as a result of the accumulation of the values for each product at the rollover point. These accumulations are carried out following a 'Query' passed to the Riposte Message Server which returns all the required records which satisfy the conditions of the query. The returned records are then used to calculate the required value for the rollover.

On occasions the Riposte Message Server query may fail and the error goes undetected by the application software. The result of such a failure is usually manifested in the failure to accumulate any values for a particular Primary or Secondary Mapping node in the rollover records (e.g. all Stock - Primary Mapping Level 4 = 3008 or all Receipts - Primary Mapping Level 4 = 3013). Such a failure at any of the rollover points will cause the Cash Account to mis-balance.

In order to identify whether this has happened, the appropriate sets of rollover records need to be examined to ensure that they contain records for all the major Level 4 nodes (Stock, Receipts, Payments, and all the Rem In and Out locations as a minimum, there may be other Level 4 values for Transfers, Revaluations etc.).

9.3.4.6 Missing Cash Account Mappings

During the Cash Account preparation process, the system reads the office-level Container:## records and assigns each record a set of Cash Account Mappings. These Cash Account Mappings identify the line numbers on the Cash Account into which the transaction value should be added according to the mode in which the transaction took place (the hierarchy consists of up to 7 levels for each valid mode of operation for the product). In the event that the system is unable to locate the Cash Account Mapping record for a particular product in a particular mode then the value will not be added to the Cash Account Lines for which it was intended and the result will be a mis-balance of the Cash Account.

In order to identify whether this has occurred, the Container:## record values need to be compared with the CAP Summary transaction values to determine whether there is a one-to-one match.

9.3.4.7 'Knock On' Effects

Where an office has produced a mis-balanced Cash Account and the root cause of the mis-balance is an issue which involves the mis-reporting of any of the values that report to Line 1085 of the Cash Account (the 'Balance Due to Post Office' line), then there is a probability that this will cause a further imbalance to occur in the following Cash Account Period. The values that report to Line 1085 are primarily the Cash and Stock (including Suspense Account) values from Table 5 and the Discrepancies from lines 0701 and 0702.

FUJ00079425

The value on Line 1085 is used as the Brought Forward value at the start of the next Cash Account Period and appears on line 0001 of the next Cash Account. If the value was mis-calculated in the previous CAP then the following CAP will be starting with an incorrect value and will certainly misbalance.

9.3.4.8 Missing/Duplicated Stock Transfers

Stock Transfers in the Horizon system should be self-balancing. Transfers Out from one stock unit to another are required to be transferred in before either of the stock units involved in the transfer is allowed to balance. However, should anything occur that allows a stock unit to balance while there is an outstanding transfer or should a transfer be allowed to be accepted into a stock unit more than once then the result will be that the value of Cash and Stock in the office will have changed without there being an equal and opposite effect elsewhere in the office.

Since Transfers are not separately reported on the Cash Account (they are considered an in-office activity that does not need to be reported to Chesterfield), the Horizon system deals with them by adding the values of the transfers to the appropriate Cash or Stock line of Table 5 of the Cash Account. Under normal circumstances that has a nil effect on the value of the line but where an error has occurred the value of the line will be adversely affected, causing a mis-balance of the Cash Account.

9.3.4.9 Mis-calculated Product Value

The process of accumulating values for the stock unit, office and Cash Account relies on the use of Markers in the message store to identify the points between which records are to be accumulated. Markers consist of messages, which record the exact message number on each node in the office at the time that the marker was requested and recorded. The Marker values are then used to bound the lower and upper points in the message store to be used by a scan or query in order to improve performance.

In a number of places within the system processes use only a lower marker and accumulate all values up to the current position in the message store. Whilst this is generally an acceptable practice, there can be occasions where process interlocks in the office have failed and the assumption that the interlocks are active and have prevented other users from committing further data while a critical process is taking place may prove to be incorrect.

As an example, in normal circumstances only one user of a stock unit may be logged on to the system and operating while the stock unit is balanced. Should this interlock fail, the balancing process may well accumulate the current stock position and record the stock rollover records and then accumulate the Receipt or Payment transaction values. Between these activities, if another is logged on and recording transactions for the stock unit, it is possible that the accumulation of the Receipt/Payment transactions may pick up a new transaction but the stock position has already been recorded and therefore the settlement of the new transaction will not be taken into account. It is also possible that on occasions a relevant transaction record may be recorded as the message immediately before or after a Marker message and that the application processing may exclude such a message (in error).

FUJ00079425

9.4 Reconciliation Issues

The counter reconciliation software runs as part of the Counter EOD Service on a daily basis. The High Level Design for the EOD Service is documented in EP\DES\025, with the Low Level Design for Daily Reconciliation contained in EP\DES\003 and the Low Level Design for the Weekly Reconciliation contained in EP\DES\004. The EOD Service runs 30 minutes after the time specified in reference data as the closing time for the office on the current day, or at 19:00 if no closing time is specified for the current day.

After the EOD Marker is written the reconciliation software performs two reconciliation functions:

- Daily Reconciliation Checks
- Weekly (Cash Account) Reconciliation Checks

The daily checks consist of calculating the total number of transactions recorded in the message store since the previous EOD Marker and accumulating the absolute Quantities and Sale Values in these records. These values are written to a message in the message store which is later compared with a similar calculation carried out within the TPS Host system. Any errors detected in the transaction records (e.g. missing Primary Mappings, null Mode: attribute) are written as error messages to the message store for later reporting through the Reconciliation reports run at the Data Centre(s). The counter software then writes a series of messages to the message store recording the total value of transactions during the period which are to contribute to each Cash Account Table at the end of the CAP - this is referred to as the Mini Cash Account. On the day in which the Cash Account is produced the reconciliation software will write two sets of these mini Cash Account records, one relating to the transactions for the CAP prior to rollover and one for transactions relating to the CAP after rollover.

When the reconciliation software detects that a Cash Account rollover has taken place in the current period, a further check is carried out to determine whether the values written to the Cash Account Lines agree with the total of the values in the mini Cash Account records for the CAP. If differences are detected these are written as error messages in the message store for later reporting through the Reconciliation Reports run at the Data Centre(s).

FUJ00079425

9.4.1 General

Reconciliation issues are generally reported as a result of the Reconciliation Report outputs from the Data Centre(s) on a daily basis. These issues will either be related to 'pure' reconciliation issues where the values reported from the counter software differ from the equivalent calculations from the TPS Host software or where there is the detection of some error in the data laid down in the message store by the counter software.

9.4.2 Probable Causes

9.4.2.1 Differences in the number of Transactions between Counter and Host

These are usually the result either of missing attributes (e.g. Start Date/Time, Mode, Omode (on a reversal)) or where one or other of the systems has misinterpreted the start and end points of the period.

In either case, examination of the records in the message store between the two EOD Harvest Trailer (EODHT) markers should reveal the cause - either through the recording of error messages at the time of the reconciliation processing at the counter or through physically checking the accuracy of the record counts recorded in the message store.

In order to check the record counts it should be borne in mind that the following transaction records are excluded from the message store count:

- Transfers In
- Transfers Out
- Transactions for Product Numbers > 10,000

9.4.2.2 Null Mode, Null Primary Mappings or Null Start Date/Time Attributes

These errors are usually recorded by the reconciliation software in the message store. The reconciliation error message will identify the message Id and Number of the offending message. These issues need to be passed to the appropriate counter development team to analyse why the error occurred.

FUJ00079425

9.4.2.3 Nil Count received from the Counter

This is usually caused by either a malfunction of the Counter Reconciliation software or by the misinterpretation of the location of the reconciliation total records by the Agent Harvester.

The message store should be checked to ensure that a normal set of daily reconciliation messages were written and that the appropriate EOD and EODHT Markers were written. If this is the case then the issue should be passed to the Agent development team for further analysis.

9.5 Interface Issues

9.5.1 General

Interface issue generally arise from values exceeding the size of the field allowed on the interface for the item or from a failure of the validation checks carried out by the target system.

As the system matures, the majority of these issues have been identified and removed, however there remains the possibility that these errors can occur.

9.5.2 Probable Causes

9.5.2.1 Invalid Mode

The POCL TIP system rigidly enforces validation of the Item-Transaction Mode relationships identified in Reference Data. At CSR the Horizon counter software does not enforce these relationships and will allow a product to be transacted in any of the modes in which either a product menu button is available or where the product PLUImpulse is valid. At CSR+, the Horizon software enforces the Item-Transaction Mode relationships supplied in reference data.

It is possible, therefore, in CSR to record a message with an 'invalid' mode which will subsequently be rejected by POCL TIP. One particular way in which this can occur is on the migration of ECCO+ data. On occasions a transaction will have been carried out in ECCO+ for a product which has no equivalent Horizon Product Number. These values are migrated against the defined 'Contingency' product (currently 'Cash'). It is possible that the un-mapped product may have been transacted in a mode for which 'Cash' is not valid according to the Horizon Item-Transaction Mode Reference Data (e.g. a Remittance in from Supplies Division (RISD)). In these circumstances the Horizon Balancing and Cash Account processes will correctly assign the value and bring it to account but the POCL TIP system will reject the file in which the transaction is sent.

9.5.2.2 Quantity or Value too Large

There are defined limitations on the TIP Interface for the size of the Transaction Quantity and Sale Value fields. Where transactions are initiated by the user, validation

FUJ00079425

checks at the counter prevent users from recording transactions which would exceed these field sizes.

However, there are types of transactions recorded at the counter which are the result of the system calculating values (such as discrepancies and stock adjustments). These transactions are not currently the subject of the same level of validation applied to user transactions. It is possible, therefore, for the system to generate a transaction with a large quantity or large sale value, which will then be rejected at the TIP Interface.

There is little that can be done about these transactions since they represent the actual transaction quantities and values that were recorded at the outlet and which have/will be used for the calculation of the balances and the Cash Account at the outlet.

FUJ00079425

10 APPENDIX A – EPOSS NT EVENT LOG USAGE CATALOGUE

All Events recorded by the EPOSS Product are from the Counter Platform.

All Events recorded by the EPOSS Product are NT Log Type Application.

All Events recorded by the EPOSS Product have a Severity Type of either Error, Warning or Information.

Source Application	Event Message	Type	Support Notes
EPOSSDataServer	Unexpected method call clsSession.Notify received	Error	The .Notify method was called in class Session
EPOSSDataServer	Unexpected method call clsSession.CallInterface received	Error	The .CallInterface method was called in class Session
EPOSSDataServer	Invalid additional criteria. VarType is <*VarType*>	Error	Called from Function CompareToAdditionalCriteria in class InternalSession. An additional criteria was not of type String
EPOSSDataServer	Invalid comparison operator in additional criteria. Comparison operator is <*Operator*>. Criteria is <*Criteria*>.	Error	Called from Function CompareToAdditionalCriteria in class InternalSession. The comparison operator of an additional criteria was not in ['EQ','eq' or 'DEQ'].
EPOSSDataServer	Error in DestroyMessagePort: < *ResultError*>	Error	A RIPOSTE function. Called from CompletePopulateTree in class InternalSession.
EPOSSDataServer	Error in DeleteQuery: < *ResultError*>	Error	A RIPOSTE function. Called from CompletePopulateTree or TidyUpAfterCompletePopulateTree in class InternalSession.
EPOSSDataServer	Error in CreateMessagePort: <*ResultError*>	Error	A RIPOSTE function. Called from CompletePopulateTree in class InternalSession.
EPOSSDataServer	[<*DesktopCurrentApp*>] Error in CreateQuery(<*Criteria*>) <*ResultError*>	Warning	Called from CompletePopulateTree in class InternalSession.
EPOSSDataServer	[<*DesktopCurrentApp*>] Error in QueryStatus(<*Criteria*>) <*ResultError*>	Warning	Called from CompletePopulateTree in class InternalSession.

FUJ00079425

Source Application	Event Message	Type	Support Notes
EPOSSStockUnit	Error connecting to OLE Server <*ClassName*>	Error	1.Called from BrokerHandle in modMain. Attempted to create a handle to an object which returned nothing. 2. Called from Connect in forDesktop class. Attempted to get a Message object handle or the Message's Event object handle.
EPOSSStockUnit	Error connecting to OLE Server <*ClassName*>. [0x<*HexErrNo*>] : <*ErrMessage*>	Error	Called from Connect in forDesktop class. Used in procedure error handler.
EPOSSStockUnit	Unexpected error in Main. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from sub Main in modMain. Used in error handler and probably caused when EventLogOpen fails.
EPOSSReportProcessor	Could not get Config Info for Cash Account ID - <*CurrentDate&Time*>	Error	Called from sub GetUniqueID in modDesktopHelpers in STOCKUNIT (NOT actually from ReportProcessor), also from GetUniqueId_ErrHandle in modDesktopHelpers in ReportProcessor. Caused when 3rd attempt to get Config Info fails.
EPOSSStockUnit	StockUnit Rollover Error EPOSSProduct data missing for Product No. <*ProdNo*> (Rolling <*CurrentStockUnit*> from CAP <*CurrentCAP*>	Error	Called from GetNonInventoryPMs in modDesktopHelpers
EPOSSStockUnit	Unable to get current marker. Error: <*ResultError*>	Error	Called from GetGetDrawerItemDeclarations in modDesktopHelpers and GetONCHDrawerItemDeclarations in class Outlet.
EPOSSStockUnit	Unexpected error in SetDeclarationTrailer.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from SetDeclarationTrailer in modDesktopHelpers Used in procedure error handler.
EPOSSStockUnit	Unexpected error in InitialiseNew.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from InitialiseNew in class Rollover. Used in procedure error handler.
EPOSSStockUnit	Unexpected error in Rollover.[0x<HexErrNo>] <*ErrMessage*>	Error	Called from Rollover in class Rollover. Used in procedure error handler.
EPOSSStockUnit	Unexpected error in SetRolloverTrailer.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from SetRolloverTrailer in classes Rollover and Outlet. Used in procedure error handler.

Source Application	Event Message	Type	Support Notes
EPOSSStockUnit	Unexpected error in SetRevaluationTrailer.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from SetRevaluationTrailer in class Rollover. Used in procedure error handler.
EPOSSStockUnit	Unexpected error in SetOpeningFiguresTrailer.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from SetOpeningFiguresTrailer in classes Rollover and Outlet. Used in procedure error handler.
EPOSSStockUnit	Unable to get Revaluation up/down rating product id for <*ProductNo*>	Error	Called from CreateRevaluationFigures in class Rollover. Used in procedure error handler.
EPOSSStockUnit	Unexpected error in SetOpeningFiguresTrailer.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from SetOpeningFiguresTrailer in class Rollover. Used in procedure error handler. Also called from SetNonInventoryStockTrailer (coding error is presumed)
EPOSSStockUnit	Unable to get connection status to node <*NodeID*>	Error	Called from CheckNeighboursConnected in class Outlet.
EPOSSStockUnit	Lost Cash Account ID - <*CurrentDate&Time*>	Error	Called from CallInterface in Outlet class.
EPOSSStockUnit	Unexpected error, Cannot Rollover Default StockUnit	Error	Called from CallInterface in Outlet class.Happens if vars *OfficeCAPRecord* and *sDEFSU* are both equal to "".
EPOSSStockUnit	Error creating balance declarations. Error: <*ResultError*>	Error	Called from ProcessFinalBalance in Outlet class. Happens when a call to CreateBalanceDeclarations does not return a successful result.
EPOSSStockUnit	Error checking for discrepancies.Error: <*ResultError*>	Error	Called from Discrepancies in Outlet class. Happens when a calls NonStockDiscrepancies or StockDiscrepancies do not return a successful result.
EPOSSStockUnit	Unable to get unique identifier for <*CashAccountFiguresId*>.Error: <*ResultError*>	Error	Called from ProcessNewCashAccount in Outlet class. Happens when a call to GetUniqueID does not return a successful result.
EPOSSStockUnit	NON INVENTORY PM'S (Recovery) <*Index*>	Info	Called from GetNonInventoryPMs in modDesktopHelpers
EPOSSStockUnit	NON INVENTORY PM'S Search Failure TALOP_CREATE_QUERY_EX <Index>	Warning	Called from GetNonInventoryPMs in modDesktopHelpers.Happens when an attempt to create a query does not return a successful result.

FUJ00079425

Source Application	Event Message	Type	Support Notes
EPOSSStockUnit	NON INVENTORY PM'S TALOP_QUERY_STATUS <strIndex>	Warning	Called from GetNonInventoryPMs in modDesktopHelpers.Happens when an attempt to get a query status does not return a successful result.
EPOSSStockUnit	ClsRollover::DeleteAllDecalartionsForStock Unit - Failed to delete object, collection: <*Collection*> ObjectName: <*ObjectName*> <*ReturnValue*>	Warning	Called from DeleteAllDeclarationsForStockUnit in Rollover class. Typo in message is as seen. Happens when the ReturnValue from a call to deletePersistentObject is not equal to "".
EPOSSStockUnit	Warning: An attempt was made to invoke a public interface before the application has been initialised.	Warning	Called from CallInterface and GetObjectHandle from forDesktop class. Happens when the flag fAppStarted=false.
EPOSSReport	Unable to get connection status to node <*NodeID*>	Error	Called from CheckNeighboursConnected in forDesktop class. (Report Broker object, not Report Processor)
EPOSSReport	GlobalObjects.dat is Missing or Corrupt	Error	Called from CheckGOVersion in forDesktop class. Happens when the string length of the "Data.Version" attribute value is less than 6 characters. (Report Broker object, not Report Processor)
ReportProcessor	GlobalObjects.dat is Missing or Corrupt	Error	Called from CheckGOVersion in forDesktop class. Happens when the string length of the "Data.Version" attribute value is less than 6 characters.
EPOSSAppMain	Unexpected error in Main. Error: <*HexErrNo*>] <*ErrMessage*>	Error	Called from Initialize in EPOSS class. Used in general procedure error handler.
EPOSSAppMain	Unable to BEGIN transaction. <*ResultError*>	Error	Called from ProcessStockTransferTransactions in EPOSS class. Happens when an attempt to start a transaction does not return a successful result.
EPOSSAppMain	Unable to UNDO transaction. <*ResultError*>	Error	Called from ProcessStockTransferTransactions in EPOSS class. Happens when an attempt to undo a transaction does not return a successful result.
EPOSSAppMain	Unable to END transaction. <*ResultError*>	Error	Called from ProcessStockTransferTransactions in EPOSS class. Happens when an attempt to end a transaction does not return a successful result.

Source Application	Event Message	Type	Support Notes
EPOSSAppMain	Class initialisation completed	Info	Called from Initialise in EPOSS class.
EPOSSAppMain	Temporary ProductGroup code. Elapsed Time: <*ElapsedTime*> Lines Written: <*LinesWritten*>	Info	Called at end of CreateProductGroupsInMsgStore in EPOSS class.
EPOSSDeclare	Unexpected error in Main. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from sub Main in modMain. Used in procedure error handler and probably caused when EventLogOpen fails.
EPOSSDeclare	Unable to get unique identifier for DrawerItemDeclarationId. Error: <*ResultError*>	Error	Called fromCommitList in EPOSSDeclare class. Happens when a call to GetUniqueID does not return a successful result.
EPOSSDeclare	Unable to create drawer item declaration detail. Error: <*ResultError*>	Error	Called fromCommitList in EPOSSDeclare class. Happens when calls to CreateDrawerItemDeclarationDetail, CreateDrawerItemDeclaration or CreateONCHDeclaration do not return a successful result.
EPOSSDeclare	Error connecting to OLE Server <*ClassName*>. [0x<*HexErrNo*>] : <*ErrMessage*>	Error	Called from general procedure error handler in Connect in EPOSSDeclare class.
EPOSSDeclare	Error committing declarations. Error: <*ResultError*>	Error	Called from HandleCmdStr in EPOSSDeclare class. Happens when a call to CommitList does not return a successful result.
EPOSSDeclare	DrawerItemDeclaration entry not found. Error <*ResultError*>	Error	Called from NextDrawerItemDeclaration in EPOSSDeclare class. Happens when an attempt to get an entry does not return a successful result.
EPOSSDeclare	Unexpected error in "Initialize". Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from Initialize in EPOSSDeclare class. Used in general procedure error handler.
EPOSSDeclare	DrawerONCHItemDeclaration entry not found. Error <*ResultError*>	Error	Called from NextONCHDrawerItemDeclaration in EPOSSDeclare class. Happens when an attempt to get an entry does not return a successful result.
EPOSSDeclare	Unable to get current marker. Error: <*ResultError*>	Error	Called from GetGetDrawerItemDeclarations in EPOSSDeclare class. Happens when an attempt to get a marker does not return a successful result.
EPOSSDeclare	Warning: An attempt was made to invoke a	Warning	Called from CallInterface from EPOSSDeclare class. Happens when

Source Application	Event Message	Type	Support Notes
	public interface before the application has been initialised.		the flag fAppStarted=false.
EPOSSScales	Unexpected error in Main. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from Initialize in Scales class NOT from Main. Used in general procedure error handler.
EPOSSScales	Class initialisation complete	Info	Called from Initialize in Scales class.
BESEPOSSFallback	Unexpected error in Main. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from Initialize in BESSFallbackDesktop class NOT from Main. Used in general procedure error handler.
BESEPOSSFallback	Class initialisation complete	Info	Called from Initialize in BESSFallbackDesktop class.
EPOSSWatchDog	Unexpected error in Main. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from Initialize in ConnectionWatch class NOT from Main. Used in general procedure error handler.
EPOSSWatchDog	Unexpected error in Timer. Error: [0x<*HexErrNo*>] <*ErrMessage*> The node connection timer has NOT been restarted.	Error	Called from Timer event of TimerNodeConnected object on form Timers. Used in general event error handler.
EPOSSWatchDog	Unexpected error in Timer. Error: [0x<*HexErrNo*>] <*ErrMessage*> The server connection timer has NOT been restarted.	Error	Called from Timer event of TimerServerConnected object on form Timers. Used in general event error handler.
EPOSSWatchDog	Class initialisation completed	Info	Called from Initialize in ConnectionWatch class.
EPOSSWatchDog	Node and server watch timers NOT started due to missing definition.	Info	Called from ProcessLogon in ConnectionWatch class. Happens if there's no persistent object called NodeWatch
EPOSSWatchDog	Failed to initialise the message object	Warning	Called from Initialize in ConnectionWatch class. Happens when an attempt to get a EPOSSMessage.clsForDesktop object handle does not return a successful result.
CASEPOSSEvProducts	Unable to initialise scheduler object	Error	Called from Load event of form frmEOD. Happens if the function funMain=false.
CASEPOSSEvProducts	Error No : <*ErrNo*>, Error Description : <*ErrMessage*>	Error	Called from Load event of form frmEOD. Used in general event error handler.
CASEPOSSEvProducts	A Counter Application Scheduled task could not been started due to missing	Warning	Called from funMain in modMain. Happens if the identifier key passed by the scheduler is an empty string.

FUJ00079425

Source Application	Event Message	Type	Support Notes
	environmental parameters		
CASEPOSSDailyRecon	The EPOSSReconciliation Txn Selection Reference Data is missing.	Error	Called from Main in modMain. Happens when the Rdata attribute in an EPOSSTxnSelection object is an empty string.
CASEPOSSDailyRecon	An error has occurred in modMain.Main. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from Main in modMain. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred within modMain.InitSchedulerLink. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from InitSchedulerLink in modMain. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred whilst extracting the command line parameter <*ParamName*> from the command line <*CmdLine*>. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from CmdLineGetValue in modMain. Used in general procedure error handler.
CASEPOSSDailyRecon	The clsEPOSSReconciliation CallInterface function has failed. The following error was reported: Num: <*ErrNo*> Desc: <*ErrMessage*>	Error	Called from CallInterface in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSDailyRecon	Unable to successfully complete EOD EPOSS Private Reconciliation.	Error	Called from PerformEPOSSEODRecon in EPOSSReconciliation class. Happens if a call to PerformPrivateReconciliation returns false.
CASEPOSSDailyRecon	Performing Private Reconciliation on date <*Date*>	Error	Called from PerformPrivateReconciliation in EPOSSReconciliation class. Only occurs when a global debugging flag is set to True and not normally seen.
CASEPOSSDailyRecon	Performing Public Reconciliation on date <*Date*>	Error	Called from PerformPublicReconciliation in EPOSSReconciliation class. Only occurs when a global debugging flag is set to True and not normally seen.
CASEPOSSDailyRecon	Reading in accumulations between <*LowerMarker*> and <*UpperMarker*>	Error	Called from ReadInPrivateSubTotals in EPOSSReconciliation class. Only occurs when a global debugging flag is set to True and not normally seen.

Source Application	Event Message	Type	Support Notes
CASEPOSSDailyRecon	<*ReconErrReason9999*> | 9999	Error	Called from ValidateTxn in EPOSSReconciliation class. Happens if we are validating the transactions in the EPOSSTxnValidation collection and the collection is empty. If the persistent object ReconErrReason(no. 9999) does not have the attribute Data.Reason, then just the value 9999 is logged.
CASEPOSSDailyRecon	End of Day Criteria Not Valid <*Criteria*>	Error	Called from ValidateTxn in EPOSSReconciliation class. Happens if the operator in the criteria is not in this list [EQ,NE,GT,GE,LT,LE,LIKE]
CASEPOSSDailyRecon	An error has occurred within clsEPOSSReconciliation.EliminateDuplicate CAPs. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from EliminateDuplicateCAPs in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred within clsEPOSSReconciliation.EliminateSubsequ entDuplicateCAP. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from EliminateSubsequentDuplicateCAPs in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred within clsEPOSSReconciliation.FirstMarkerLarger. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from FirstMarkerLarger in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred in clsRiposteAPI.Init. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from Init in RiposteAPI class. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred in clsRiposteAPI.AllNodesConnected. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from AllNodesConnected in RiposteAPI class. Used in general procedure error handler.
CASEPOSSDailyRecon	An error has occurred in clsRiposteAPI.GetNeighbourNodeIds. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from GetNeighbourNodeIds in RiposteAPI class. Used in general procedure error handler.

Source Application	Event Message	Type	Support Notes
CASEPOSSDailyRecon	Attempting to connect to Riposte...	Info	Called from Init in RiposteAPI class.
CASEPOSSDailyRecon	Attempt to connect to Riposte has succeeded	Info	Called from Init in RiposteAPI class.
CASEPOSSWeeklyRecon	The EPOSSReconciliation Txn Selection Reference Data is missing.	Error	Called from Main in modMain. Happens when the Rdata attribute in an EPOSSTxnSelection object is an empty string.
CASEPOSSWeeklyRecon	An error has occurred in modMain.Main. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from Main in modMain. Used in general procedure error handler.
CASEPOSSWeeklyRecon	An error has occurred within modMain.InitSchedulerLink. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from InitSchedulerLink in modMain. Used in general procedure error handler.
CASEPOSSWeeklyRecon	An error has occurred whilst extracting the command line parameter <*ParamName*> from the command line <*CmdLine*>. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from CmdLineGetValue in modMain. Used in general procedure error handler.
CASEPOSSWeeklyRecon	The clsEPOSSCAPReconciliation CallInterface function has failed. The following error was reported: Num: <*ErrNo*> Desc: <*ErrMessage*>	Error	Called from CallInterface in EPOSSCAPReconciliation class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	Duplicate Node <*Node*>	Error	Called from AccumulateCALines in EPOSSCAPReconciliation class.
CASEPOSSWeeklyRecon	No Data For <*Node*>	Error	Called from AccumulateCALines in EPOSSCAPReconciliation class.
CASEPOSSWeeklyRecon	No Cash Account Lines Found for Last Rollover (CAP <*CAP*>) (CA Year <*CAYear*>)	Error	Called from AccumulateCALines in EPOSSCAPReconciliation class. Happens when the collection object colCALines=nothing.
CASEPOSSWeeklyRecon	Duplicate Data in BF <*Node*>	Error	Called from FetchBFCALines in EPOSSCAPReconciliation class.
CASEPOSSWeeklyRecon	No Data present For <*Node*>	Error	Called from FetchBFCALines in EPOSSCAPReconciliation class.

Source Application	Event Message	Type	Support Notes
CASEPOSSWeeklyRecon	An error has occurred in clsRiposteAPI.Init. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from Init in RiposteAPI class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	An error has occurred in clsRiposteAPI.AllNodesConnected. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from AllNodesConnected in RiposteAPI class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	An error has occurred in clsRiposteAPI.GetNeighbourNodeIds. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from GetNeighbourNodeIds in RiposteAPI class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	The clsEPOSSReconciliation CallInterface function has failed. The following error was reported: Num: <*ErrNo*> Desc: <*ErrMessage*>	Error	Called from CallInterface in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	Unable to successfully complete EOD EPOSS Private Reconciliation.	Error	Called from PerformEPOSSEODRecon in EPOSSReconciliation class. Happens if a call to PerformPrivateReconciliation returns false.
CASEPOSSWeeklyRecon	Performing Private Reconciliation on date <*Date*>	Error	Called from PerformPrivateReconciliation in EPOSSReconciliation class. Only occurs when a global debugging flag is set to True and not normally seen.
CASEPOSSWeeklyRecon	Performing Public Reconciliation on date <*Date*>	Error	Called from PerformPublicReconciliation in EPOSSReconciliation class. Only occurs when a global debugging flag is set to True and not normally seen.
CASEPOSSWeeklyRecon	Reading in accumulations between <*LowerMarker*> and <*UpperMarker*>	Error	Called from ReadInPrivateSubTotals in EPOSSReconciliation class. Only occurs when a global debugging flag is set to True and not normally seen.
CASEPOSSWeeklyRecon	<*ReconErrReason9999*> | 9999	Error	Called from ValidateTxn in EPOSSReconciliation class. Happens if we are validating the transactions in the EPOSSTxnValidation collection and the collection is empty. If the persistent object ReconErrReason(no. 9999) does not have the attribute Data.Reason,

Source Application	Event Message	Type	Support Notes
			then just the value 9999 is logged.
CASEPOSSWeeklyRecon	End of Day Criteria Not Valid <*Criteria*>	Error	Called from ValidateTxn in EPOSSReconciliation class. Happens if the operator in the criteria is not in this list [EQ,NE,GT,GE,LT,LE,LIKE]
CASEPOSSWeeklyRecon	An error has occurred within clsEPOSSReconciliation.EliminateDuplicateCAPs. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from EliminateDuplicateCAPs in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	An error has occurred within clsEPOSSReconciliation.EliminateSubsequentDuplicateCAP. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from EliminateSubsequentDuplicateCAPs in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	An error has occurred within clsEPOSSReconciliation.FirstMarkerLarger. Error No: <*ErrNo*> Error Desc: <*ErrMessage*> Source:<*ErrSource*>	Error	Called from FirstMarkerLarger in EPOSSReconciliation class. Used in general procedure error handler.
CASEPOSSWeeklyRecon	Attempting to connect to Riposte...	Info	Called from Init in RiposteAPI class.
CASEPOSSWeeklyRecon	Attempt to connect to Riposte has succeeded	Info	Called from Init in RiposteAPI class.
EPOSSStatus	Invalid attribute. Function: CreateDataTree Collection: [Data.BalanceRootNode | Data.StockRootNode] ObjectName: <*ObjName*> Argument:<*Arg*>	Error	Called from CreateDataTreeWithRolloverId, CreateNonInventoryFigures, StockUnitNonInventory, CreateDataTreeWithRolloverIdForCashAccount and CreateaSpecialTree in CurrentFigures class. Happens if the vars lBalanceRootNode or lstockrootnode = 0. The call to log this event message never provides values for ObjName or Arg so these two are expected to be always blank.

FUJ00079425

Source Application	Event Message	Type	Support Notes
EPOSSStatus	Error creating data tree. PopulateTree failed.	Error	Called from CreateDataTreeWithRolloverId, CreateaSpecialTree, StockUnitNonInventory, CreateDataTreeWithRolloverIdForCashAccount and CreateNonInventoryFigures in CurrentFigures class. Happens when a call to the DataServer object to populate the tree returns false.
EPOSSStatus	Invalid attribute. Function: CreateDataTree Collection: [Data.BalanceRootNode | Data.StockRootNode] ObjectName: <*ObjName*> Argument:<*Arg*>	Error	Called from CreateNonInventoryFigures in CurrentFigures class. Happens if the vars lBalanceRootNode or lstockrootnode = 0. The call to log this event message never provides values for ObjName or Arg so these two are expected to be always blank.
EPOSSStatus	Unable to create object Desktop.dtpDesktopClass	Error	Called from Initialize in forDesktop class. Happens when the CreateObject function returns nothing.
EPOSSStatus	Unable to create object clsCurrentFigures	Error	Called from Initialize in forDesktop class. Happens when the New method returns nothing.
EPOSSStatus	Error creating data tree. BuildTree failed.	Error	Called from CreateDataTreeWithRolloverId, CreateaSpecialTree, StockUnitNonInventory, CreateDataTreeWithRolloverIdForCashAccount and CreateNonInventoryFigures in CurrentFigures class.

FUJ00079425

11 Appendix B – EPOSS Audit Files Usage Catalogue

The following events are recorded by EPOSS Applications in addition to the auditing of the application version being executed by the desk top service.

Source Application	Audit Message	Support Notes
EPOSSDataServer	DS:clsNode:IncreaseDtMaxIfNecessary <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in IncreaseDtMaxIfNecessary in Node class.
EPOSSDataServer	DS:: clsInternalSession.CompletePopulateTree: Failed to retrieve OpeningFiguresId for SUReprint Report [CAP:<*CAP*>:BP:<*BP*>]	Called from CompletePopulateTree in InternalSession class. Happens when then OpeningFiguresID attribute value is equal to ""
EPOSSDataServer	DS:: clsInternalSession.CompletePopulateTree: Report [<*ReportTitle*>] executed the transaction query [<*TransactionQuery*>].	Called from CompletePopulateTree in InternalSession class.
EPOSSDataServer	DS:: clsInternalSession.CompletePopulateTree: Report [<*ReportTitle*>] the executed query returned [<*RecordCount*>] records.	Called from CompletePopulateTree in InternalSession class.
EPOSSDataServer	DS:clsInternalSession:TransactionHasAReversal <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in TransactionHasAReversal in InternalSession class.
EPOSSDataServer	DS:clsInternalSession:NotAReversal <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in AReversal in InternalSession class (note-message text says NotAReversal).
EPOSSDataServer	DS:clsInternalSession:ReturnTransaction <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in ReturnTransaction in InternalSession class
EPOSSDataServer	DS:clsInternalSession <TALOP_ACCESS_QUERY failed to return record number <*RecordNum*>>	Called from ResultSuccessForRecordRetrieval in InternalSession class
EPOSSStockUnit	SU::clsOutlet.RolloverChecks - ERROR [<*ErrNo*> :	Called from procedure error handler in RolloverChecks in Outlet

FUJ00079425

Source Application	Audit Message	Support Notes
	<*ErrMessage*>].	class
EPOSSStockUnit	SU:clsOutlet.RemoveRolloverButton [ERROR:<*ErrNo*>:<*ErrMessage*>]	Called from procedure error handler in RemoveRolloverButton in Outlet class
EPOSSStockUnit	SU:clsOutlet.SaveMenuPositionWhenAppropriate [ERROR:<*ErrNo*>:<*ErrMessage*>]	Called from procedure error handler in SaveMenuPositionWhenAppropriate in Outlet class
EPOSSStockUnit	SU:clsOutlet.NavigateBack [ERROR:<*ErrNo*>:<*ErrMessage*>]	Called from procedure error handler in NavigateBack in Outlet class
EPOSSStockUnit	SU:: clsOutlet:FinalBalanceReportConfirmation <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in FinalBalanceReportConfirmation in Outlet class
EPOSSStockUnit	SU:clsStockUnit.Delete [ERROR:<*ErrNo*>:<*ErrMessage*>]	Called from procedure error handler in Delete in StockUnit class
EPOSSReportBroker	RB.modEPOSSHelpers.IsThisMigrationPeriod <<*ErrNo*>::<*ErrMessage*>>	Called from procedure error handler in IsThisMigrationPeriod in modEPOSSHelpers
EPOSSReportBroker	RB.modEPOSSHelpers.GetCAPS <<*ErrNo*>::<*ErrMessage*>>	Called from procedure error handler in GetCAPS in modEPOSSHelpers
EPOSSReportBroker	RB.modEPOSSHelpers.IsCurrentPeriod <<*ErrNo*>::<*ErrMessage*>>	Called from procedure error handler in IsCurrentPeriod in modEPOSSHelpers
EPOSSReportBroker	RB.modEPOSSHelpers.SetCriteria <<*ErrNo*>::<*ErrMessage*>>	Called from procedure error handler in SetCriteria in modEPOSSHelpers
EPOSSReportBroker	RB.clsForDesktop.Initialise <<*ErrNo*>::<*ErrMessage*>>	Called from procedure error handler in Initialize in forDesktop class.
EPOSSReportProcessor	[RP.clsReport.PrintCancelRequest] Error trying to delete old print jobs.	Called from PrintCancelRequest_ErrHandle in Report class. Happens when a call to objPServer.CancelJob sets Err.Number to a non-zero value.
EPOSSReportProcessor	<DeviceError:Error found in ReportProcessor.clsReport.DeviceErrorHandler><Data:<*Data*>>	Called from DeviceErrorHandler_ErrHandle in Report class. Happens when a call to WriteMessage with this same message text as a parameter does not return with a positive success code.

FUJ00079425

Source Application	Audit Message	Support Notes
EPOSSReportProcessor	<Message:An error has occurred in [ReportProcessor <*SourceFunction*>].><Error:(<*ErrNo*>)<*ErrMessage*>>	Called from subPrintErrorReport in Report class. Happens when a call to WriteMessage with this same message text as a parameter does not return with a positive success code.
EPOSSReportProcessor	ERROR: [<*SourceFunction*>] (<*ErrNo*>)<*ErrMessage*>	Called from subPrintErrorReport in Report class.
EPOSSReportProcessor	RP:clsReport:Reversals <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in Reversals in Report class
EPOSSReportProcessor	RP:clsReport:NoReversals <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in NoReversals in Report class
EPOSSReportProcessor	RP:clsReport:BuildTheTree <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in NoReversals in Report class
EPOSSReportProcessor	RP:: GlobalGetObject failed - Check Event Log	Called from GlobalGetTemporalObject_ErrHandle in frmHelpers form. Happens when a call to GlobalGetObject returns the result code "FATAL"
EPOSSReportProcessor	RP:: GlobalGetObject(<*Collection*>, <*Object*>) Error(<*ErrNo*>), <*ErrMessage*>	Called from GlobalGetTemporalObject_ErrHandle in frmHelpers form. Happens when a call to GlobalGetObject sets Err.Number to a non-zero value.
EPOSSReportProcessor	RP::clsSection.Output - Section ID - <*SectionID*>	Called from Output_ErrHandle in Section class. Happens when objReport.lLastReportID = 25
EPOSSReportProcessor	<Error:An error has occurred in [ReportProcessor. <*SourceFunction*>].><Code:<*ErrNo*>><Description:<*ErrMessage*>>	Called from funWriteErrorMessage in modDesktopHelpers. Happens when a call to WriteMessage with this same message text as a parameter does not return with a positive success code.
EPOSSReportProcessor	GlobalObjects.DAT Version [<*Version*>]	Called from CheckGOVersion in forDesktop class.
EPOSSReportProcessor	RP:: ERROR: GlobalObjects.dat is Missing or Corrupt	Called from CheckGOVersion in forDesktop class. Happens when the length of the <*Version*> string is less than 6 characters.
EPOSSReportProcessor	RP:: AutoRestart Data = [<*Data*>]	Called from CheckGOVersion in forDesktop class. Happens when the length of the <*Version*> string is less than 6 characters.
EPOSSReportProcessor	RP:: AutoRestart already done today	Called from CheckGOVersion in forDesktop class. Happens when the length of the <*Version*> string is less than 6 characters and the date of the last restart is equal to today.

Source Application	Audit Message	Support Notes
EPOSSReportProcessor	RP:: AutoRestart data not present	Called from CheckGOVersion in forDesktop class. Happens when the length of the <*Version*> string is less than 6 characters and the start and end times for the AutoRestart object are equal to ""
EPOSSReportProcessor	RP:: Can not AutoRestart at this time	Called from CheckGOVersion in forDesktop class. Happens when the length of the <*Version*> string is less than 6 characters and the current time is outside the start and end times for the AutoRestart
EPOSSReportProcessor	RP:: RESTARTING DESKTOP	Called from CheckGOVersion in forDesktop class. Happens when the length of the <Version> string is less than 6 characters and everything is OK for the AutoRestart
BESReports	BR.clsBESReports.MiscellaneousTransactionsReport <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in MiscellaneousTransactionsReport in BESReports class
BESReports	BR.clsBESReports.BuildMiscTxnsQuery <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in BuildMiscTxnsQuery in BESReports class
BESReports	BR.clsBESReports.RetrieveMiscTxns <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in RetrieveMiscTxns in BESReports class
BESReports	BR.clsBESReports.PrintMiscTxns <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in PrintMiscTxns in BESReports class
BESReports	BR.clsBESReports.InsertSuperGroups <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in InsertSuperGroups in BESReports class
BESReports	BR.clsBESReports.PrintCompleted <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in PrintCompleted in BESReports class
BESReports	BR.utilsBES.CutOffBESReports <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in CutOffBESReports in utilsBES module.
EPOSSApp	EC:Err Null Mode <*Command*>	Called from CallInterface in EPOSS class. Happens when the length of objDesktop.CurrentMode is zero.
EPOSSCommon	LockDesktopRequest Error : <*ErrorMessage*>	Called from LockDesktopRequest in Txn class. Happens when a call to LockUser returns with a non-zero result code, or when a subsequent call to UnlockUser returns the same.

FUJ00079425

Source Application	Audit Message	Support Notes
EPOSSCommon	EPOSSCommon: clsCommon.LockDesktopRequest <<*ErrNo*>:<*ErrMessage*>>	Called from procedure error handler in LockDesktopRequest in Common class.
EPOSSCommon	Loading DLL - <*DLLHandle*>	Called from LoadDLL in forDesktop class.
PCDFEndOfDay	PCDFEndOfDay:: clsPCDFEndOfDayScript.CallInterface: call received <Cmd:<*Cmd*>>	Called from CallInterface in PCDFEndOfDayScript class.
Testperipherals	<DeviceError:Error found in TestPeripheral.clsEngineerApp.DeviceErrorHandler><Data:<*Data*>>	Called from DeviceErrorHandler in EngineerApp class. Happens when a call to WriteMessage with this same message text as a parameter does not return with a positive success code.

FUJ00079425

12 Appendix C – Example EPOSS Audit File Entries

```
05/05/2000
10:44:34        EPOSSCommon (eposscommon) Version 14.3.341 [WP8035 Build Mon Apr 10 14:51:57 2000(GMT)]
10:46:04        BESReports (BESReports) Version 2.0.871 [provides the functionality for the counter reporting (BES)]
10:46:04        BusinessObject (businessobject) Version 14.1.320 [WP7727 Build Tue Mar 14 10:27:32 2000(GMT)]
10:46:04        EngineerApp (testperipherals) Version 14.2.328 [WP7819 Build Thu Mar 30 10:26:15 2000(GMT)]
10:46:07        MiMAN (MiMAN) Version 2.2.93 []
10:46:40        EPOSSDataServer (EPOSSDataServer) Version 14.3.339 [WP7729 Build Mon Apr 17 13:45:47 2000(GMT)]
10:46:47        ReportProcessor (REPORTPROCESSOR) Version 14.3.348 [WP8039 Build Fri Apr 28 07:15:22 2000(GMT)]
10:46:49        GlobalObjects.DAT Version [WP8039.RDMCv45.RDMC08v11.RDMC09v08.RDMC10v07]
10:46:51        EPOSSScales (EPOSSScales) Version 3.0.32 []
10:47:01        EPOSSReport (EPOSSReportBroker) Version 3.0.32 []
10:47:02        EPOSSStockUnit (EPOSSStockUnit) Version 4.0.19 []
10:47:02        EPOSSSetMain (EPOSSSettlementObject) Version 14.3.347 [WP8037 Build Tue Apr 25 13:21:52 2000(GMT)]
10:47:39        EPOSSAppMain (EPOSSCore) Version 3.0.59 [CSR+]
10:47:39        EPOSSReport (EPOSSReportBroker) Version 3.0.32 []
10:48:10        EPOSSDeclare (EPOSSDeclare) Version 14.3.348 [WP8039 Build Tue May  2 10:08:50 2000(GMT)]
10:49:00        OBCS (OBCS) Version 14.2.336 [WP7820 Build Mon Apr  3 16:36:18 2000(GMT)]
10:57:55        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] executed the transaction query
[<Name:EPOSSDataServer1><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58321>>>].
10:57:55        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] the executed query returned [0] records.
11:00:20        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] executed the transaction query
[<Name:EPOSSDataServer2><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58325>>>].
11:00:20        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] the executed query returned [0] records.
11:01:07        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] executed the transaction query
[<Name:EPOSSDataServer4><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58329>>>].
11:01:07        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] the executed query returned [0] records.
15:46:31        LockDesktopRequest Error : UnlockUser called when not locked.
16:00:47        EPOSSCommon (eposscommon) Version 14.3.341 [WP8035 Build Mon Apr 10 14:51:57 2000(GMT)]
16:02:14        BESReports (BESReports) Version 2.0.871 [provides the functionality for the counter reporting (BES)]
16:02:14        BusinessObject (businessobject) Version 14.1.320 [WP7727 Build Tue Mar 14 10:27:32 2000(GMT)]
16:02:14        EngineerApp (testperipherals) Version 14.2.328 [WP7819 Build Thu Mar 30 10:26:15 2000(GMT)]
16:02:17        MiMAN (MiMAN) Version 2.2.93 []
16:02:23        EPOSSDataServer (EPOSSDataServer) Version 14.3.339 [WP7729 Build Mon Apr 17 13:45:47 2000(GMT)]
16:02:24        ReportProcessor (REPORTPROCESSOR) Version 14.3.348 [WP8039 Build Fri Apr 28 07:15:22 2000(GMT)]
16:02:24        GlobalObjects.DAT Version [WP8039.RDMCv45.RDMC08v11.RDMC09v08.RDMC10v07]
```

FUJ00079425


```
16:02:24        EPOSSScales (EPOSSScales) Version 3.0.32 []
16:02:29        EPOSSReport (EPOSSReportBroker) Version 3.0.32 []
16:02:29        EPOSSStockUnit (EPOSSStockUnit) Version 4.0.19 []
16:02:29        EPOSSSetMain (EPOSSSettlementObject) Version 14.3.347 [WP8037 Build Tue Apr 25 13:21:52 2000(GMT)]
16:03:08        EPOSSAppMain (EPOSSCore) Version 3.0.59 [CSR+]
16:03:09        EPOSSReport (EPOSSReportBroker) Version 3.0.32 []
16:03:39        EPOSSDeclare (EPOSSDeclare) Version 14.3.348 [WP8039 Build Tue May  2 10:08:50 2000(GMT)]
16:04:30        OBCS (OBCS) Version 14.2.336 [WP7820 Build Mon Apr  3 16:36:18 2000(GMT)]
18:10:36        EPOSSCommon (eposscommon) Version 14.3.341 [WP8035 Build Mon Apr 10 14:51:57 2000(GMT)]
18:12:06        BESReports (BESReports) Version 2.0.871 [provides the functionality for the counter reporting (BES)]
18:12:06        BusinessObject (businessobject) Version 14.1.320 [WP7727 Build Tue Mar 14 10:27:32 2000(GMT)]
18:12:06        EngineerApp (testperipherals) Version 14.2.328 [WP7819 Build Thu Mar 30 10:26:15 2000(GMT)]
18:12:09        MiMAN (MiMAN) Version 2.2.93 []
18:12:15        EPOSSDataServer (EPOSSDataServer) Version 14.3.339 [WP7729 Build Mon Apr 17 13:45:47 2000(GMT)]
18:12:16        ReportProcessor (REPORTPROCESSOR) Version 3.0.47 []
18:12:16        GlobalObjects.DAT Version [WP8039.RDMCv45.RDMC08v11.RDMC09v08.RDMC10v07]
18:12:17        EPOSSScales (EPOSSScales) Version 3.0.32 []
18:12:21        EPOSSReport (EPOSSReportBroker) Version 3.0.32 []
18:12:21        EPOSSStockUnit (EPOSSStockUnit) Version 4.0.19 []
18:12:22        EPOSSSetMain (EPOSSSettlementObject) Version 14.3.347 [WP8037 Build Tue Apr 25 13:21:52 2000(GMT)]
18:13:00        EPOSSAppMain (EPOSSCore) Version 3.0.58 [CSR+]
18:13:00        EPOSSReport (EPOSSReportBroker) Version 3.0.32 []
18:13:31        EPOSSDeclare (EPOSSDeclare) Version 14.3.348 [WP8039 Build Tue May  2 10:08:50 2000(GMT)]
18:14:20        OBCS (OBCS) Version 14.2.336 [WP7820 Build Mon Apr  3 16:36:18 2000(GMT)]
18:15:43        DS:: clsInternalSession.CompletePopulateTree: Report [] executed the transaction query
[<Name:EPOSSDataServer1><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58447>>>].
18:15:43        DS:: clsInternalSession.CompletePopulateTree: Report [] the executed query returned [79] records.
18:16:21        DS:: clsInternalSession.CompletePopulateTree: Report [DVLA V10 - Office Copy] executed the transaction query
[<Name:EPOSSDataServer2><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.SaleValue><Type
:Number><Order:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND (NOT Exists(EPOSSTransaction.OpeningFiguresId)) AND
EPOSSTransaction.PM.L2 EQ "145") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ
""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58456>>>].
18:16:21        DS:: clsInternalSession.CompletePopulateTree: Report [DVLA V10 - Office Copy] the executed query returned [2] records.
18:25:48        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer3><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58588>>>].
18:25:48        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [151] records.
18:26:33        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer4><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
```

FUJ00079425

r:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3031" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:26:33 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:26:39 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer5><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3037" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:26:39 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:26:46 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer6><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3030" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:26:46 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:26:52 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer7><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3045" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:26:52 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:26:59 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer8><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3038" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:26:59 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:05 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer9><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Typ
e:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number>
<Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Orde
r:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3042" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:27:05 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:11 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer10><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord
er:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "2493" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:27:11 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:17 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer11><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord

FUJ00079425

er:Forward>>><SelectExpression:(EPOSSTransaction.PM.L4 EQ "3013" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58599>>>].
18:27:17 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [14] records.
18:27:25 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer12><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.PM.L4 EQ "3016" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58613>>>].
18:27:25 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [2] records.
18:27:31 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer13><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.SM.L3 EQ "3015" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:27:31 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:37 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer14><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.SM.L3 EQ "3011" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:27:37 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:42 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer15><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3046" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:27:42 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:48 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer16><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3033" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:27:48 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:53 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer17><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3034" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:27:53 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:56 DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer18><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Order:Forward>>><SelectExpression:(EPOSSTransaction.PM.L2 EQ "2495" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L2 EQ "2495") AND ((NOT Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].

FUJ00079425


```
18:27:56        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:27:59        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer19><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord
er:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3047" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:27:59        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:28:04        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer20><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord
er:Forward>>><SelectExpression:(EPOSSTransaction.SM.L2 EQ "3048" AND TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58615>>>].
18:28:04        DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query returned [0] records.
18:28:10        DS:: clsInternalSession.CompletePopulateTree: Report [Final Balance - Office Copy] executed the transaction query
[<Name:EPOSSDataServer21><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord
er:Forward>>><SelectExpression:(TxnData.Container EQ "S1" AND EPOSSTransaction.PM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ ""))><Lower:<Mark:<1:58283>>><Higher:<Mark:<1:58616>>>].
18:28:10        DS:: clsInternalSession.CompletePopulateTree: Report [Final Balance - Office Copy] the executed query returned [151] records.
18:31:32        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Report] executed the transaction query
[<Name:EPOSSDataServer22><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord
er:Forward>>><SelectExpression:(TxnData.Container EQ "##" AND EPOSSTransaction.PM.L5 EQ "3017" OR EPOSSTransaction.SM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ
"123456_1_58699"))><Lower:<Mark:<1:58697>>><Higher:<Mark:<1:58726>>>].
18:31:32        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Report] the executed query returned [19] records.
18:32:12        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Report] executed the transaction query
[<Name:EPOSSDataServer23><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.ProductNo><Type:Number><Ord
er:Forward>>><SelectExpression:(TxnData.Container EQ "##" AND EPOSSTransaction.PM.L5 EQ "3017" OR EPOSSTransaction.SM.L5 EQ "3017") AND ((NOT
Exists(EPOSSTransaction.OpeningFiguresId)) OR (EPOSSTransaction.OpeningFiguresId EQ
"123456_1_58699"))><Lower:<Mark:<1:58697>>><Higher:<Mark:<1:58758>>>].
18:32:12        DS:: clsInternalSession.CompletePopulateTree: Report [Balance Report] the executed query returned [19] records.
18:33:10        DS:: clsInternalSession.CompletePopulateTree: Report [Cash Account] executed the transaction query
[<Name:EPOSSDataServer24><SortKeys:<Key:<Attribute:EPOSSTransaction.PM.L7><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L6><Ty
pe:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L5><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L4><Type:Number
><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L3><Type:Number><Order:Forward>><Key:<Attribute:EPOSSTransaction.PM.L2><Type:Number><Order:F
orward>><Key:<Attribute:EPOSSTransaction.PM.L1><Type:Number><Order:Forward>>><SelectExpression:EPOSSTransaction.PM.L7 EQ "6999" AND
OpeningFiguresId EQ "123456_1_58759"><Lower:><Higher:<Mark:<1:58792>>>].
18:33:10        DS:: clsInternalSession.CompletePopulateTree: Report [Cash Account] the executed query returned [30] records.
18:33:10        RP::clsSection.Output - Section ID - 5000
18:33:11        RP::clsSection.Output - Section ID - 5001
18:33:16        RP::clsSection.Output - Section ID - 5002
18:33:27        RP::clsSection.Output - Section ID - 5003
18:33:42        RP::clsSection.Output - Section ID - 5004
```

FUJ00079425


```
18:33:54        RP::clsSection.Output - Section ID - 5005
18:34:05        RP::clsSection.Output - Section ID - 5006
```

.

FUJ00079425

13 Appendix D - Support Route

4th line support of the EPOSS Product defined within this document is provided by the EPOSS Development Team.

The support provided is available to Pathway Programme testing teams and Live Product Support.

For Live Support of the EPOSS Product any call should first be routed to the EPOSS Business Support Analyst for analysis of the problem. Support is provided 0800 until 2200.

For Testing problems raised weekdays (Monday to Friday) in office hours any support needs should be raised to the EPOSS Development Team Senior Team Leader.

For Testing problems raised at weekends, support should be raised to the EPOSS Development Team Leader responsible for Software Development. Support is provided 0800 until 2200.

FUJ00079425

14 Appendix E – EPOSS System Error Catalogue

The primary purpose of this appendix is to define a catalogue of the unexpected events that the system will react to. The catalogue goes beyond this and explains the nature of each event that the appropriate corrective action can be put in place.

Error Code	Explanation
00001	PropogateCommand has failed (tried to call a non-existent application). The application of this check is currently limited to the invocation of LFS as a consequence of settling a remit session in mode ROAD from CSR+ onward.
00002	PropogateCommand has failed (tried to call a non-existent command in an existing application). The application of this check is currently limited to the invocation of LFS as a consequence of settling a remit session in mode ROAD from CSR+ onward.
01001	A Visual Basic error has occurred whilst performing some action in EPOSSCore. The message in the messagestore will contain more details of what happened and where exactly.
01002	An error in loading mode information from the messagestore (or via supplied parameter) in LoadInfo has occurred. The supplied mode name does not match the objectname.
01003	A transaction was attempted where the current mode is set to be NULL. This is not allowed as the transaction would not be committed against any stock unit and would lead to an unbalanced Cash Account.