FUJ00091192

Document Title: S80 Impact Release 3 EPOSS Counter Operational Support Guide

Document Type: SPG

Release: S80

Abstract: This is the Operational Support Guide for the S80 EPOSS Counter.

Document Status: APPROVED

Originator & Dept: Jon Hulme, APDU Counter Development

Contributors: S80 Impact Release 3 Counter Development Team

Internal Distribution:

External Distribution:

Approval Authorities:

Name	Role	Signature	Date
Mik Peach	Operations & Support Service Manager		
Mark Scardifield	Development Manager		

FUJ00091192

Document Control

0.1 Document History

Version No.	Date	Reason for Issue	Associated CP/PEAK/PPRR Reference
0.1	28/04/2005	First draft for input from development team.	CP3716
0.2	10/05/2005	Includes contributions from the development team.	
1.0	20/05/2005	Issued for approval. Incorporates reviewers comments.	

0.2 Review Details

Review Comments by :	
Review Comments to :	Originator & Document Management

Mandatory Review	
Mik Peach*	Operations & Support Services Manager
Walter Wright*	Development Team Leader
Optional Review	
Mark Scardifield*	Development Manager
Phil Hemingway	Designer
Martin Nixon	Designer
Martin McConnell	Developer
Phil Orton	Developer
Gerald Barnes*	Developer
Mike Coon	Developer
Rob Dinnadge*	Developer
David Leatherdale	Developer
Anne Chambers*	SSC System Specialist
*Issued for Information – **Please restrict this distribution list to a minimum***	

(*) = Reviewers that returned comments

FUJ00091192

0.3 Associated Documents

Reference	Version	Date	Title	Source
PA/TEM/001			Fujitsu Services Document Template	PVCS
EP/MAN/001			EPOSS Operational Support Guide	PVCS
NB/SPG/001			NB Counter Operational Support Guide	PVCS
EA/DPR/004			This document is the Design Proposal for the Horizon aspects of Release Three of IMPACT (was known as End-to-End Re-architecting Release 2).	PVCS
EA/HLD/005			Impact Release 3 Counter Design for Balancing, Rollover and Stock Processing	PVCS
EA/HLD/006			Impact Release 3 Counter Design for Declaration, Correction and Revaluation	PVCS
EA/HLD/008			IMPACT Release 3 Migration High Level Design	PVCS
EA/IFS/011			Impact Release 3 Report Production User Interface	PVCS
EA/IFS/012			Impact Release 3 Declaration, Correction and Revaluation User Interface	PVCS
EA/IFS/013			Impact Release 3 Balancing and Trading Statement Production User Interface	PVCS
SD/DES/005			Horizon OPS Reports and Receipts - Post Office Account Horizon Office Platform Service	PVCS
SD/DOC/009			Horizon OPS Desktop Messages and Help Text	PVCS
SD/SPE/016			Horizon OPS Menu Hierarchy	PVCS
SD/SPE/022			Horizon OPS Menu Hierarchy:	PVCS

FUJ00091192

			Changes Supplement – Issue 235	
RD/DES/056			Reference Data End to End High Level Design for S80 (Impact, Track & Trace, +1 Sales)	PVCS
EA/LLD/003			Counter Message Retention Periods Low Level Design	PVCS
EA/LLD/004			Counter Balancing Functional Low Level Design for Impact Release 3	PVCS
EA/LLD/005			In Day Transactions Low Level Design	PVCS
EA/LLD/006			IMPACT Release 3 – Trading Period Rollovers Low Level Design	PVCS
EA/LLD/007			Maintain Office Variances Low Level Design	PVCS
EA/LLD/008			IMPACT Release 3 Transaction Correction Application	PVCS
EA/LLD/015			Impact Release 3 Receipts & Reports	PVCS
EA/LDD/019			Data Protection Low Level Design	PVCS
EP/LLD/008			EPOSS Stock Unit Low Level Design	PVCS
EP/LLD/009			EPOSS Report Broker Low Level Design	PVCS
EP/LLD/010			EPOSS Report Processor Low Level Design	PVCS
EP/LLD/011			EPOSS BES Reports Low Level Design	PVCS
EP/LLD/012			EPOSS Core Low Level Design	PVCS
EP/LLD/013			EPOSS Declare Low Level Design	PVCS
EP/LLD/014			EPOSS Common Low Level Design	PVCS

FUJ00091192

EP/LLD/015			EPOSS Settlement Object Low Level Design	PVCS
EP/LLD/017			EPOSS Dataserver Low Level Design	PVCS
EP/LLD/025			Low-Level Design for BI3S80 MiMAN	PVCS
NB/LLD/056			SoftLaunch Low Level Design	PVCS

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

N.B. Printed versions of this document are not under change control.

0.4 Abbreviations/Definitions

Abbreviation	Definition
BP	Balance Period
BTS	Branch Trading Statement
CAP	Cash Account Period
CBDB	Counters Business DataBase (POL's current accounting system)
EA	End-To-End Architecture
EP	EPOSS
EPOSS	Electronic Point Of Sale System
HLD	High Level Design
LLD	Low Level Design
POL	Post Office Limited
PM	Primary Mappings
SM	Secondary Mapping
TP	Trading Period
TM	Tertiary Mappings

0.5 Changes in this Version

Version	Changes
0.1	This is the first version.
0.2	Inclusion of contributions from development team.

FUJ00091192

1.0	Whole Document – Miscellaneous typos. Security classification set to consistent capitalisation of "COMPANY-in-CONFIDENCE". Section 0.3 – Added reference SD/DOC/009. Corrected SD/SPE/022 reference. Section 0.4 – Abbreviation added: BP, CDBD, PM, POL, SM, TM. Section 3.1.2 – Additions to impacted components table. Section 3.2.1 – Clarified that it is stock units and the office rather than the counter that operate in CAP or TP mode. Clarified that it is the S80 release that first operates in CAP mode. Section 3.2.2 – Reiterated that the accounting period moves from weekly CAP to four weekly TP. Explained that Transaction Corrections replace the previous manual error notice system and are part of the overall aim of Impact Release 3 to gain better accounting control over the branches. Section 3.2.3, 4.10.1, 4.10.3 – Made more visible the statement that DataProtection is run by the NT scheduler not the Counter Application Scheduler. Stated that the DailyCACT messages are no longer written Section 3.3 – Clarified that components that are not used at S80 are still loaded by the desktop. A few changes to the lists. Section 4.2.3.6.2 – Clarified that BP1 is implicit and the change described only applied to further BP rollovers. Section 4.2.5 – Clarified that AccountingPeriods is in subscription group 111111111. Section 4.2.6.3 – SoftLaunch enabling product number range specified. Section 4.3.4 – Clarified which trailers are written when. Section 4.3.6.2 – Fixed bad reference. Clarified that CashAccLines are not longer written in TP mode and diagnosis should use BTS trailers and BTSSourceData. Section 4.7.3 – BESReport and ReportProcessor/Global Objects can both print to the tally roll and the back office printer. Section 4.7.6.3.1 – Expanded to describe the BTS section in more detail and provides an example BTS. Section 4.9.1 – Explanation as to why MaintainOfficeVariances still run. Section 7.0 – Added further explanation re BTS CP 4002 and updated example BTS to be one with a trading position of zero.

0.6 Changes Expected

Changes
None.

FUJ00091192

0.7 Table of Contents

FUJ00091192

FUJ00091192

1.0 Introduction

The overall aim of S80 Impact Release 3 is to improve accounting control of branches. This is addressed by two major new requirements:

- Introduction of stock handling by volume, rather than by value.
- Introduction of the (4 weekly) Trading Period rather than the (weekly) Cash Account Period.

These changes impact the EPOSS accounting model and many EPOSS reports.

This Support Guide provides an overview of the functional changes, cross-references associated documentation, and provides a component breakdown in order to assist fault location, diagnosis and resolution.

This document also lists the set of errors that can be produced and their meaning.

2.0 Scope

This document limits the scope to S80 Impact Release 3 changes on the counter.

The related Support Guides are:

Doc. Ref.	Title	Comment
EP/MAN/001	EPOSS Operational Support Guide	Written in October 2000 for the then current EPOSS product.
NB/SPG/001	NB Counter Operational Support Guide	Covers counter Network Banking. Updated as S80 to include Plus One.
AP/DOC/003	APS Support Guide	

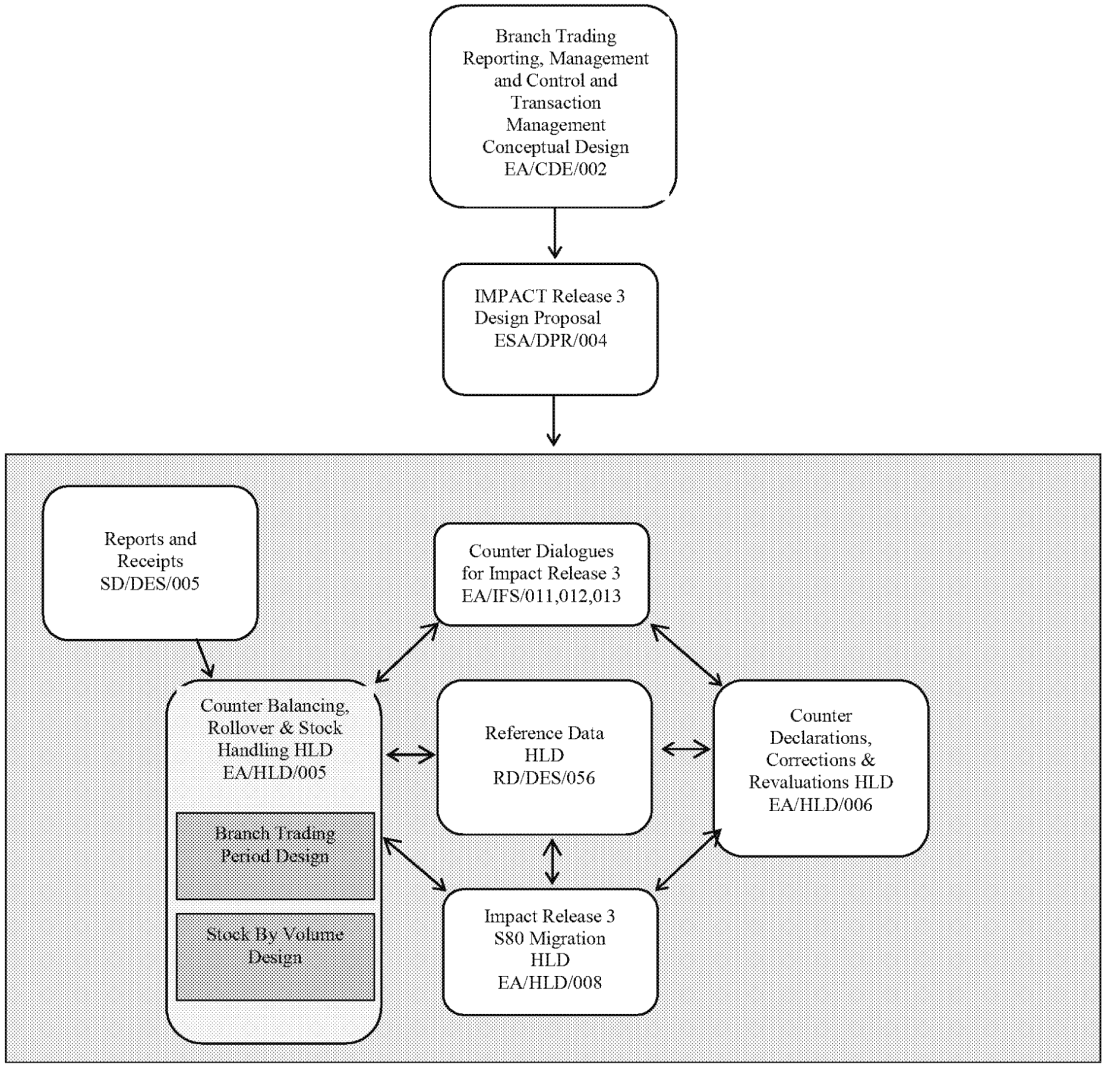
FUJ00091192

3.0 Overview

3.1 Documentation

3.1.1 High Level Design

The following diagram (taken from EA/HLD/005 section 1.0) summaries the High Level Design documents. These documents are cross-referenced in later sections.



FUJ00091192

3.1.2 Low Level Design

Since S80 Impact Release 3 addresses distinct functional areas across a wide range of components, a new set of LLDs were written, one per functional area rather than one per component (these documents start [EA/LLD/…]).

The original component LLDs are at a very low level (these documents start [EP/LLD/…]). Some of these have been updated where appropriate.

This full set of relevant LLDs is given in section 0.3. The following table lists all changed LLDs and the components they impact. These LLDs are cross-referenced in later sections where appropriate.

Doc. Ref.	Title	Affected Components
EA/LLD/003	Message Retention Periods	BESReports.dll EPOSSCommon.dll EPOSSCore.dll EPOSSDataServer.dll EPOSSDeclare.dll EPOSSMessage.dll EPOSSReportBroker.dll EPOSSSettlement.dll EPOSSStockUnit.dll EPOSSWatchDog.dll LFSConfirm.dll NBFinalise.dll OBCS.dll ReportProcessor.dll TestPeripherals.dll
EA/LLD/004	Counter Balancing	EPOSSDataServer.dll EPOSSStockUnit.dll GlobalObjects ReportProcessor.dll
EA/LLD/005	In Day Transactions	EPOSSCore.dll EPOSSSettlement.dll
EA/LLD/006	Trading Period Rollovers *(Addresses CAP to TP transition)*	BESReports.dll EPOSSCommon.dll EPOSSDataServer.dll EPOSSDeclaration.dll EPOSSMessage.dll EPOSSReportBroker.dll EPOSSStockUnit.dll EPOSSWatchdog.dll

FUJ00091192

EA/LLD/007	Maintain Office Variances	CASEPOSSImpact.exe
EA/LLD/008	Transaction Corrections	EPOSSCore.dll EPOSSDeclare.dll EPOSSStockUnit.dll EPOSSTxnCorrection.dll
EA/LLD/015	Receipts and Reports *(Lists which reports required change)*	BESReports.dll GlobalObjects ReportBroker.dll ReportProcessor.dll
EA/LLD/019	Data Protection	DataProtection.exe
EP/LLD/011	BESReports *(Appendix contains one section per S80 Changed Report)*	BESReports.dll
EP/LLD/025	MiMAN	MiMAN.dll

3.2 EPOSS Counter

The EPOSS counter breaks down into the following three services:

- Migration Service
- In-Day Service
- EOD (End-Of-Day) Service

The following sections give an overview of the changes to these services at S80 Impact Release 3, breaking them down by functional area. Section 4.0 gives more detail on each functional area and the components that were changed to implement that functional area.

3.2.1 Migration Service

The Migration Service handles the process of migrating a counter from a previous version of Horizon, or from a brand-new counter, to a state ready for operational use.

At S80 Impact Release 3 migration involves:

- MiMAN Migration

  When a new counter is installed, it must be "migrated" to an initial operation state.

  See section 4.1 for details.

- CAP (Cash Account) to TP (Trading Period) Migration

  An S80 stock unit can operate in CAP or TP mode. When all stock units have been rolled into TP mode then the office (aka branch) can be rolled into TP mode. When S80 is first released to the estate it will operate in CAP mode. In CAP mode, S80 mimics the

FUJ00091192

behaviour of S75 (with a few differences). S80 then goes through various points on a time line, ending up operating fully in TP mode.

See section 4.2 for details.

3.2.2 In-Day Service

The In-Day service changes at S80 Impact Release 3 in the following ways:

- Counter Balancing

  When transitioned into TP mode, accounting uses Stock by Volume, the accounting period moves from weekly CAP to four weekly TP, and the Branch Trading Statement replaces the Cash Account.

  In addition, the concept of "local suspense" is introduced. This allows a stock unit to rollover, transferring a net cash discrepancy to office wide local suspense. The last stock unit in an office Trading Period to rollover must clear local suspense (typically by making good with cash).

  See section 4.3 for details.

- In Day Transactions

  When transitioned into TP mode, most stock must be transacted by Volume. The main exceptions (which continue to be treated by Value) are currently:

  - Cash
  - Cheques
  - Other Stamps

  Various EPOSS components have been changed to handle transacting stock by volume.

  See section 4.4 for details.

- Transaction Corrections

  Transaction Corrections are messages that are distributed to branches if the central accounting body of POL wants to perform a "correction" at the counter to correct existing accounting values.

  These replace the previous manual Error Notice system. The introduction of Transaction Corrections is related to the overall aim of gaining better accounting control over the branches, and the introduction of the Trading Period. Because the correction of transactions is now automated, POL can ensure they are settled by the branch before a Trading Period rollover is allowed.

  See section 4.5 for details.

- Cash Declarations and Variances

  The change of balancing mechanism from a weekly to a monthly basis necessitates the ability to reconcile cash on a more regular basis. This impacts cash declarations,

FUJ00091192

discrepancy processing, and adjustments. A new Cash Variances report was introduced, but has subsequently been dropped due to a late Change Proposal.

See section 4.6 for details.

- Receipts and Reports

  A small number of reports change while still operating in CAP mode. A large number of reports and receipts change while operating in TP mode, notably the Cash Account being replaced by the Branch Trading Statement.

  New reprint technology is introduced for the Stock Unit Balance Report and Branch Trading Statement.

  See section 4.7 for details.

- Message Retention

  Messages needed for accounting purposes must be retained for at least a trading period, and in some cases longer. The Riposte Expiry message expiry value therefore needs increasing from the pre-S80 value of 35 days.

  This issue is not strictly limited to the In Day Service, but is covered here as it is mainly In-Day accounting transaction messages that are impacted.

  See section 4.8 for details.

3.2.3 EOD Service

The End Of Day service changes at S80 in the following ways:

- CAS EPOSS Daily Recon

  Simplified so that after point 50 (office in TP mode) the Mini Cash Account reconciliation process is no longer performed (i.e. the DailyCACT messages are no longer written).

- CAS EPOSS Weekly Recon

  No longer called after point 50 (office in TP mode).

- CAS EPOSS Impact (aka Maintain Office Variances)

  This is a new component.

  A set of office variance persistent objects that were used by the now defunct Cash Variances report (see 3.2.2 above) is maintained at end of day.

  See section 4.9 for details.

- Data Protection

  This is a new component that is different to the other EOD tasks in that it is run by the NT scheduler rather than the Counter Application Scheduler.

FUJ00091192

If the earliest stock unit to roll into the current office CAP/TP did so more than a set number of days ago, riposte archiving is disabled at end of day in order to prevent messages that are about to expire from being deleted.

Prior to this, the user is also warned at logon to roll the office in to the next CAP/TP (this is part of the In-Day Service, but is covered under this topic as it is closely related to Date Protection).

See section 4.10 for details.

3.3 Components

All components are written in Visual Basic 6, unless otherwise stated below.

New code conforms to the coding standards in [NB/STD/001], although for consistency code changes often adopt the style of their context that may pre-date the coding standards.

The following counter components are new or changed at S80:

- BESReports
- CABSProcess
- CASEPOSSDailyRecon
- CASEPOSSImpact
- DailyRecon
- DataProtection
- EPOSSCommon
- EPOSSCore
- EPOSSDataServer
- EPOSSDeclare
- EPOSSMessage
- EPOSSReportBroker
- EPOSSSessionPrompts
- EPOSSSettlement
- EPOSSStatus
- EPOSSStockUnit
- EPOSSTxnCorrection
- EPOSSTxnPrompts
- EPOSSWatchDog
- GenericPAF
- GlobalObjects
- LFSPServer
- MiMAN
- NBFinalise
- NBOperationLaunch
- NBOutput
- OBCS
- PropertyBag
- ReportProcessor
- SOAPWrapper
- TestPeripherals

FUJ00091192

The following counter components are not actually used at S80, but are still loaded at desktop boot:

- BESFallback
- Scales
- ScalesConfig
- ScanKeys
- TrainingMode (VB5)

The component CASEPOSSWeeklyRecon (VB5) is not used once the office is in TP mode, but is still loaded at desktop boot.

The following counter components carry forward unchanged from S75:

- BdCCommon
- BdCCore
- BdCDialogue
- BdCRateBoard
- BusinessObject (VB5)
- CASEvProducts (VB5)
- CPrintMonitor.ocx (C)
- DC_PAFCapture
- DVLACommon
- DVLAUtil
- DVLAVRM
- DVLAOnline
- EMVInputBanking
- EMVInputRetail
- EMVLaunch
- EMVValidate
- ExtendTLS (C)
- GenericHelpers
- GenRecovery
- Global.ocx (C)
- KillProcessByName (C)
- LFSEndOfDay (C)
- LFSInterface
- LFSRIPInterface (C)
- LFSLibrary
- LFSNVStock
- LFSView
- NBComLib
- NBCommon
- NBFramework
- NBHelpers
- NBManualInput
- NBRequestReply
- NBThirdParty
- NBValidate
- PAFCommon
- PCDFEndOfDay (VB5)
- Picklist
- RiposeteBlobAPI (C)
- RiposteAPI (C)
- SoftLaunch
- StackStatus

FUJ00091192

4.0 Detail

4.1 MiMAN Migration

4.1.1 Purpose

MiMAN has been re-written at S80 Impact Release 3 due to reduced functionality. It now:

- Only initialises new branches (it does not migrate from pre-Horizon systems).
- No longer provides capturing of initial assets. Initial assets in an office are entered using normal counter Remit In functionality.
- Migrates to CAP or TP mode as indicated by the accounting calendar.

4.1.2 High Level Design

The high level design is given in [EA/HLD/005] section 5.1.2.9.

4.1.3 Low Level Design

The low level design is given in [EP/LLD/025].

MiMAN is implemented in component MiMAN.dll.

4.1.4 Message Store Messages

Various messages are written to set up the initial office, as follows :

Writes a <TranType:RollOver> marker

Writes a <TranType:CAPRollover> marker

Writes a <TranType:OpeningFiguresTrailer> trailer

For the last CAP, or any TP, writes a <TranType:BTSCFFiguresTrailer> trailer

Writes a <TranType:RolloverTrailer> trailer

Updates the <StockUnits><DEF> object to include the RolloverTrailer

Writes a <StockUnitMarkers><DEF> object

Writes an Inactive SU Rollover Event for DEF

Creates the <EPOSSCAP><Office> object

Writes report Cutoff objects for all menu items in the Office Daily and Office Weekly menu hierarchies

FUJ00091192

4.1.5 Reference Data

The CAP Calendar from collection CAWeeks and the TP Calendar from collection AccountingPeriods are read as part of the logic to determine the initial CAP or TP to migrate into. During the S80 transition from CAP accounting to TP accounting, MiMAN will also take into account any setting of the SoftLaunch Point 50 FINAL_CAP attribute (calculated from the AppConfig.EPOSSStockUnit-P50A-00 object, along with the presence of any enabling product).

The final migration status is written to collection MiMAN object ## attribute MigStatus. The success value is C (Complete), any other value indicates migration has not been performed successfully.

4.1.6 Troubleshooting

4.1.6.1 Diagnostics Written

Errors are written to the Audit log.

4.1.6.2 Tools

No specific diagnostic tools are available.

4.1.6.3 Fault Diagnosis

- Check the MiMAN ## object for the migration status (see 4.1.4 above).
- Check the contents of the EPOSSCAP.Office and StockUnits.DEF objects and the associated rollover trailers (see section 4.3 for details of rollover trailers).
- Check the Audit Log for exception messages

FUJ00091192

4.2 CAP to TP Migration

4.2.1 Purpose

[EA/HLD/005] section 9.1 contains a diagram showing the timeline of the various points that S80 Impact Release 3 goes through before operating fully in TP mode. In summary, those that have a functional effect at the counter are:

- Point 20 – S80 code arrives at counters (S80 reference data will have arrived at a point prior to this).
- Point 30 – Final CBDB CAP passed (happens at different times for each stock unit and the branch).
- Point 50 – Final CAP passed / first TP entered (happens at different times for each stock unit and the branch).

This functional area covers the changes that occur at the various points on this time line, in terms of the counter logic and user interface.

4.2.2 High Level Design

The high level design covering the CAP/TP concepts and their transition / impact is given in [EA/HLD/005] section 5.1.2.

The changes are in the following areas:

- Desktop status area CAP/TP legend.
- User Interface Menu Buttons.
- User Interface Messages.
- Event messages.
- User Interface Report Criteria.
- Receipts and Reports (see section 4.7).
- Internal counter logic.

There is no single place that defines what changes at which point, but see:

- [EA/HLD/008] for the wider context.
- [SD/SPE/022] for the definition of which menu buttons change at which point.
- [SD/DOC/009] for the definition of which message texts change as which point.
- [EP/LLD/011] for how BESReports reports change at the various points.
- [EA/LLD/006] for report criteria and internal counter logic.

FUJ00091192

4.2.3 Low Level Design

The low level design is given in [EA/LLD/006].

4.2.3.1 Desktop status area

The Desktop status area is the lower right-hand area of the screen which displays several items of status information, i.e. the name of the stock unit in use, the current user and the current CAP/BP of the stock unit in use. When the stock unit transitions to use Trading Periods (instead of CAPs) the 'CAPnn BPn' status line is replaced by a 'TPnn BPn' status line. Note that whether CAP/BP or TP/BP is displayed is dependent on the current stock unit only; other stock units, and indeed the branch, may be operating with a different CAP or TP state. Further detail can be found in [EA/LLD/006] and is implemented in component EPOSSStockUnit.dll.

4.2.3.2 User Interface Menu Buttons

During the migration for S80 Impact Release 3 each branch will move through various migration points indicating the current migration level. For full details of migration points see [EA/HLD/008]. For a counter, various menu buttons will be altered during this migration process to reflect altered functionality available to the user. These button alterations occur primarily at points 20, 30 and 50. Point 50 occurs when the final CAP is rolled over into the first TP. This will occur at different times for each stock unit and also for the branch. The buttons altered when the branch rolls into first TP are extra to the alterations already made for the stock units. See [SD/SPE/022] (version 235 or later) for a comprehensive list of all menu button alterations.

The transition of User Interface Menu Buttons from CAP mode to TP mode is handled via SoftLaunch (see [NB/LLD/056]). The design of the SoftLaunch data for menu button transitions is described in [EA/LLD/006].

4.2.3.3 User Interface Messages

Many messages presented on screen to the user contain the text 'CAP'. When the stock unit of the current user is operating in TP mode then these items of text will be replaced by 'TP'. Certain messages relate specifically to the branch rather than a stock unit and so any 'CAP' text for these will only be replaced when the branch is fully in TP mode. Certain messages will also make the distinction between the first TP and subsequent TPs, typically with the text 'first TP' and 'next TP'. See [SD/DOC/009] for full details of the textual changes to messages.

The software mechanism for making the CAP to TP message replacements is described in [EA/LLD/006]. This transition logic (i.e. which message is used at which point) is implemented via code and fixed data in component EPOSSMessage.

FUJ00091192

4.2.3.4 Event messages

Similar to User Interface Messages, certain events have the text 'CAP' which, at the appropriate time, will be replaced with 'TP'. See [EA/HLD/006] for details of the textual changes to messages.

The software mechanism for making the CAP to TP event replacements is described in [EA/LLD/006]. This transition logic (i.e. which message is used at which point) is implemented in component EPOSSMessage. The data to drive the CAP or TP replacement is embedded within the event definition itself. Each event is defined within the Events collection.

4.2.3.5 User Interface Report Criteria

For many reports the CAP value is one of the criteria which determines the report's contents. When operating in TP mode the CAP criteria legend is replaced by a TP legend, both for the on-screen user interface and on the printed report.

The transition of user interface legends on Report Criteria from CAP to TP is described in [EA/LLD/006]. The transition logic is implemented in component EPOSSReportBroker and BESReports.

4.2.3.6 Internal counter logic.

4.2.3.6.1 CAP/TP State Interfaces

Component EPOSSCommon provides new interfaces to determine the current transition state of stock units and the branch. Other components call these interfaces and then make decisions impacting the user interface, reports and other counter logic. See [EA/LLD/006] for details.

4.2.3.6.2 BP Markers

Prior to S80 all BP markers (other than BP1, which is implicit in a CAP/TP rollover) for a stock unit were stored in a single record in the StockUnitBPMarkers collection which was keyed on *StockUnit-CAPn*. Since there are likely to be more BPs per TP than there were per CAP (since a TP is 4 times longer than a CAP) and that records in the message store have a fixed limited size, each BP marker (other than BP1) is now stored in an individual record, still in the StockUnitBPMarkers collection, but now keyed on *StockUnit-CAPn-BPn*. In TP mode the TP number will be used rather than the CAP number. See [EA/LLD/006] for further details.

4.2.4 Message Store Messages

No new message formats are introduced.

4.2.5 Reference Data

The following reference data objects are of note. See [EA/LLD/006] for details.

FUJ00091192

Collection	Object	Attribute	Comment
Accounting Periods	<*yyyy*>		In group 111111111. <*yyyy*> is the year. Contains a calendar of accounting periods with their associated date ranges. Trading periods are accounting periods offset by the value of the BTSOffset specified in Outlet.OutletDetails.
AppConfig	EPOSSStock Unit-P<*tail*>		<*tail*> includes the point, and possibly scope, of the soft launch action e.g. EPOSSStockUnit-P30B-SU-10.
CounterConfig Params	Counter	CAPLegend TPLegend	Contains 'CAP'. Contains 'TP'.
DesktopButtons IItem<*n*>	IItem<*m*>		<*n*> and <*m*> identify the menu and button. SoftLaunch changes the in-memory visibility attribute at the required point.
EPOSSCAP	Office	TP TPTransition CAP, NextCAP, PreviousCAP	1=TP mode, *blank*=CAP mode. Transitional state: final CAP or first TP. CAP or TP values depending on the value of the TP and TPTransition attributes.
MessageDefs	MSG*nnnn*		New messages that contain 'TP' text replacements for 'CAP' text.
RepCriteria	44		New criteria to represent the selected TP for a stock unit. TP mode equivalent of RepCriteria.16.
StockUnits	<*SU Name*>	TP TPTransition CAP, NextCAP	See EPOSSCAP.Office.

4.2.6 Troubleshooting

4.2.6.1 Diagnostics Written

- Any errors detected are written to the Audit log.

FUJ00091192

- No new events are raised.

4.2.6.2 Tools

No specific diagnostic tools are available.

4.2.6.3 Fault Diagnosis

The following section describes some useful information about the mechanics of the CAP to TP transition that may help in performing fault diagnosis.

Stock Unit roll expected to enter TP mode but did not

The final CAP for every stock unit in a branch is determined by SoftLaunch data in :-

AppConfig.EPOSSStockUnit-P50A-00

This record lists all possible candidate final CAPs. A non-core product will be used to select a single entry from the list to be used at a particular branch (these product numbers are in the range 6247 through 6266 inclusive). When that final CAP is rolled out of then the stock unit/branch should enter the first TP, and thus TP mode.

Entry into TP mode might be delayed if the AppConfig data is missing, the appropriate non-core product is not present or the AppConfig data was not present on the counter in good time. The AppConfig data needs to be present before rolling into the final CAP; if it arrives after rolling into the final CAP then the transition will still occur but it will be delayed by one, or more, CAPs.

Selection of first TP

When rolling out of the final CAP the selection of the first TP will be made in this manner :-

What would have been the next CAP is determined,

The start date for that CAP is looked up,

The TP that covers that start date is found from checking the AccountingPeriods collection and is then used as the first TP. Note that the BTSOffset from Outlet.OutletDetails adjusts the start/end dates of the AccountingPeriods.

How to determine the CAP/TP state of a stock unit or branch

Every stock unit has a record in the StockUnits collection, keyed on the stock unit's name, which contains much of the state data for the stock unit. In each of these records the state of that stock unit's individual CAP/TP mode transition is maintained. This involves 2 new attributes introduced in S80, namely TP and TPTransition.

Note that either of these attributes being absent is treated as though it were blank.

The TP attribute taken alone indicates whether the stock unit is in CAP or TP mode.

TP:*blank*	=>	CAP mode
TP:1	=>	TP mode

If taken together the two new attributes can provide a slightly more detailed picture :-

FUJ00091192

TP:*blank*	and	TPTransition:*blank*	=> before final CAP
TP:*blank*	and	TPTransition:1	=> in final CAP
TP:1	and	TPTransition:1	=> in first TP
TP:1	and	TPTransition:*blank*	=> after first TP

A similar set of attributes exist to determine the CAP/TP state of the branch; they are located in the following record :-

EPOSSCAP.Office

Counter slow during user logon, or SU rollover, or SU attachment

During the transition to a fully migrated Impact Release 3 branch there are periods when the stock units could be in differing migration points. For example, at Point 50 a branch may contain some stock units still operating in the final CAP, thus operating in CAP mode, and other stock units that have rolled into the first TP, and thus operating in TP mode. The set of menu buttons presented to the user is different between CAP and TP mode. Therefore if a counter has to switch between stock units then it must alter the menu buttons according to the mode of the new stock unit. This alteration is performed via SoftLaunch and can take several seconds to complete. This alteration may happen during user logon, stock unit rollover or stock unit attachment.

Unexpectedly rolling into TP 13 (or above)

If, when rolling a stock unit in TP mode, the next TP selected is TP13 (or above) then this means that the AccountingPeriods data for the following year was not available at the counter and the counter has therefore rolled into a stopgap 'virtual' TP. Further rollovers will go into incrementing TP numbers until the problem with the missing AccountingPeriods data has been resolved.

Unexpectedly rolling into TP61 (or above)

If, when rolling a stock unit currently in a CAP range of 1 to 6 into the first TP, the TP selected is 61 (or above) then this means a 'bias' has been added to the normal TP number to make it distinct from the low numbered CAPs in which the counter had previously operated. The bias value added is typically 60. For the remainder of that year all TPs in that branch will also be biased, i.e. rather then TPs going from 1 to 12 they will instead go from 61 to 72. When rolling into the next year the biasing will no longer be applied.

AccountingPeriods collection in subscription group 111111111

The AccountingPeriods collection contains the calendar for the accounting periods and thus the trading periods. It is the replacement for the CAWeek calendar when in TP mode. Note that the data for the AccountingPeriods collection exists in a subscription group unlike many items of reference data. Creating dumps of messages stores may need to cater for this to ensure that AccountingPeriods data is dumped.

FUJ00091192

4.3 Counter Balancing

4.3.1 Purpose

The purpose of this functional area is to incorporate the new Stock By Volume and Trading Period requirements in to the EPOSS counter balancing logic.

This involves:

- Stock by volume changes
- Performance improvements
- Provision of figures for the Branch Trading Statement
- Reports (see section 4.7)
- Suspense using POL settlement products
- Local Suspense

4.3.2 High Level Design

The low level design is given in [EA/HLD/005] (see most of the document, but exclude reports and reprints which are covered in section 4.7 of this support guide).

User interface changes are given in [EA/IFS/013] sections 2 and 4.

4.3.3 Low Level Design

The low level design is given in [EA/LLD/004].

As described in [EA/HLD/005] new versions of the accumulators SV (Sale Value), QTY (Quantity) and RC (Record Count) are introduced in to the accounting node hierarchy, post-fixed by NTM (No Tertiary Mappings – for TP mode Value Stock and Non-inventory Stock transactions) or ETM (Extant Tertiary Mappings – for TP mode Volume Stock transactions). The accounting model is implemented in component EPOSSDataServer.dll.

New rollover trailers and user interface handling is implemented in component EPOSSStockUnit.dll.

EPOSSStatus.dll is an interface that shields EPOSSStockUnit from accessing EPOSSDataServer directly. It also manages a common instance of an EPOSSDataServer handle to be shared between EPOSSStockUnit, ReportBroker, and ReportProcessor. EPOSSStatus has been updated to reflect minor interface changes. ReportProcessor has been altered to accommodate the new accumulator types as well as changes to payments / receipts checks.

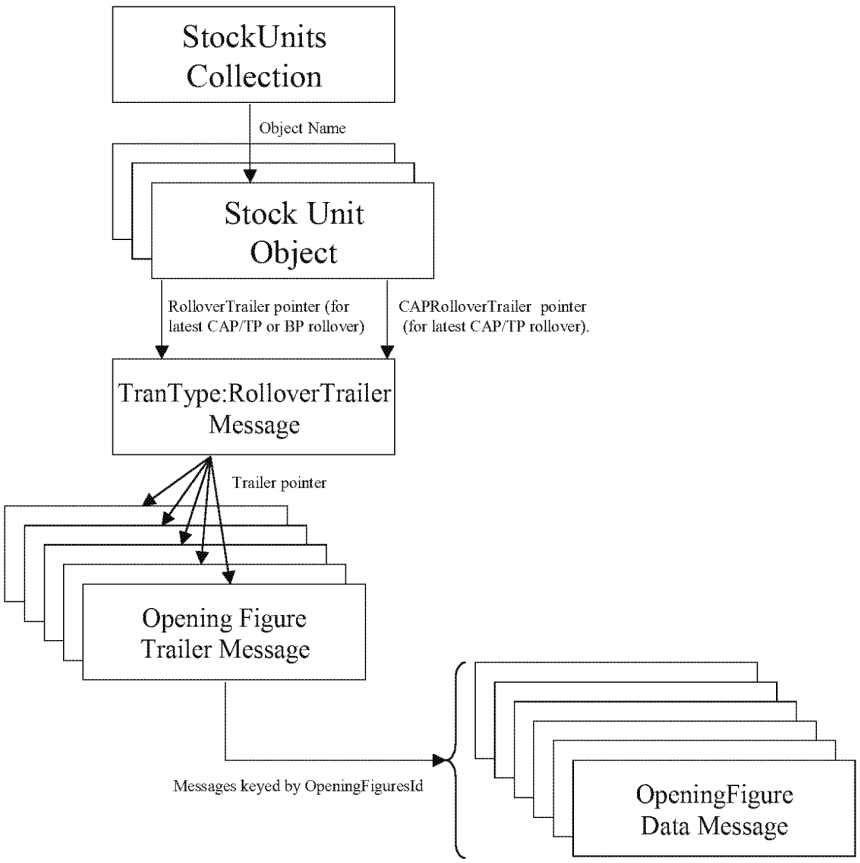
FUJ00091192

4.3.4 Message Store Messages

The stock unit and office rollover trailers have additional opening figure trailers (largely introduced when rolling in to the first TP to support the BTS) – see the following sub-sections for details.

The various opening figure trailers follow their opening figures. The opening figures id consists of GroupId_NodeId_MsgId (e.g. 90177_1_38985) where MsgId is the id of the first opening figure transaction.

4.3.4.1 Stock Unit Rollover Trailers



The following RolloverTrailer message has the following general attributes:

Attribute	Comment
BP	The CAP/TP number being rolled in to.
CAP	The CAP/TP number being rolled in to.

FUJ00091192

CAPRollover	A pointer to a message with TranType:CAPRollOver containing a marker for the start of the CAP/TP.
Container	The stock unit name.
LastCAP	The last CAP/TP number.
Previous	Points to the previous CAP/TP or BP Rollover Trailer.
PreviousCAP	Points to the previous CAP/TP rollover Trailer.
Rollover	A pointer to a message with TranType:RollOver containing a marker for the start of the CAP/TP or BP.

The RolloverTrailer message has the following attributes that are pointers to opening figures trailers:

Attribute	**Opening Figure Trailer Message TranType**	**Opening Figure Data Messages**
BTSCFFiguresTrailer	BTSCFFiguresTrailer	BTS line value messages. Used by the BTS for the C, D, E, G, H and Y sections. Written when rolling into the first TP, and on subsequent rollovers.
CurrentBTSBFFigures Trailer		BTS line value messages, copied from the previous rollover's PreBTSBFFTrailer. Used by the BTS for the B section (excluding Lines BBB and BBBB which are read from the office rollover CurrentBTSBFFiguresTrailer see section 4.3.4.2). Written when rolling into the first TP, and on subsequent rollovers.
DeclarationIds	DeclarationTrailer	Brought forward declarations; EPOSS Transactions for declaration products (aka STX messages). Written on CAP and TP rollovers (as pre S80).
NonInventory	RemProductsTrailer	Brought forward values of non-inventory product sales (i.e. Products with I:False e.g. Parcel Traffic) and movements (i.e. Remittances / Transfers). These figures are harvested, but otherwise are not used.

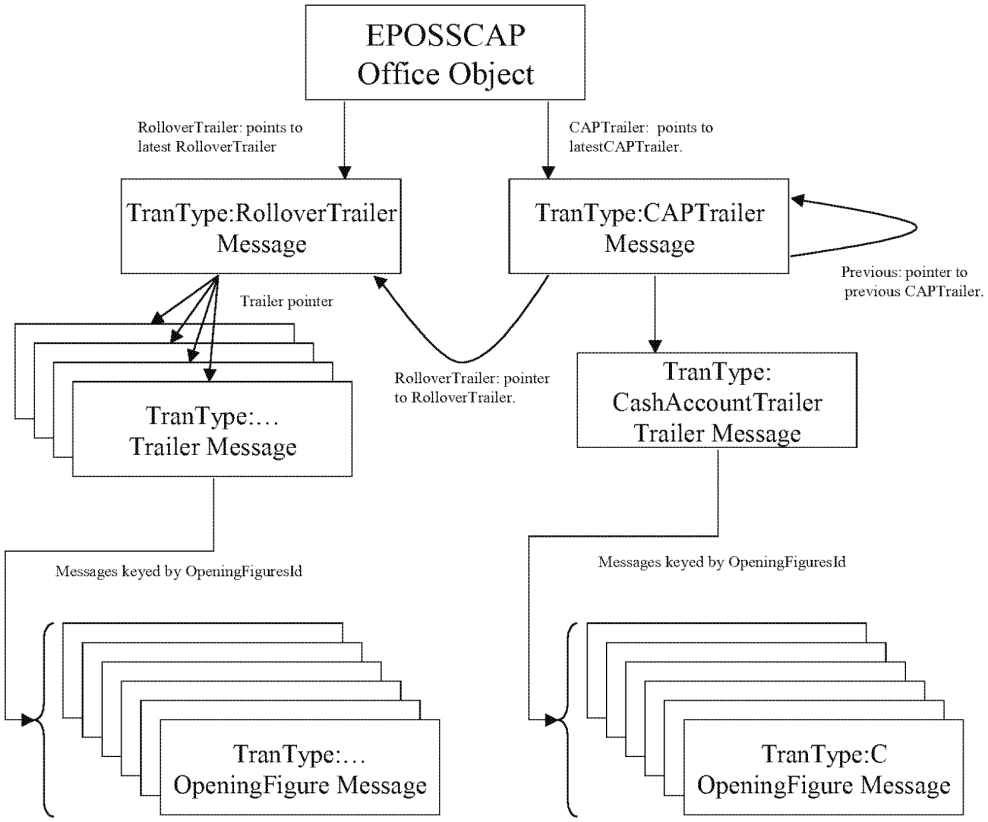
FUJ00091192

		Written on CAP and TP rollovers (as pre S80).
OpeningCAProduction FiguresOnly	OpeningFiguresTrailer	Brought forward stock holdings in CAP style (stock by value); EPOSS Transactions messages. Written only on rollover in to first TP, and not on subsequent rollovers.
OpeningFigures	OpeningFiguresTrailer	Brought forward stock holdings; EPOSS Transactions messages. Written on CAP and TP rollovers (as pre S80). When rolling into the first TP, and on subsequent rollovers, Volume Stock transactions have VolS:True and SaleValue:0.
PreBTSBFFTrailer	CurrentBTSBFFigures Trailer	BTS line value messages. Values calculated for the BTS B section, not used directly by the BTS but copied at the next rollover to become the CurrentBTSBFFiguresTrailer. Written when rolling into the first TP, and on subsequent rollovers.
SuspenseTrailer	SuspenseTrailer	Brought forward values of suspense products; EPOSSTransactions that also have <SuspenseContainer:$$>. Written on CAP and TP rollovers (as pre S80).

The details of the above messages are given in [EA/LLD/004].

FUJ00091192

4.3.4.2 Office Rollover Trailers



The RolloverTrailer message has the following general attributes:

Attribute	Comment
CAP	The CAP/TP number being rolled in to.
CAPRollover	Pointer to message with TranType:CAPRollover that contains a marker to the start of this CAP/TP.
Container	##

The RolloverTrailer message has the following attributes that are pointers to trailer messages:

FUJ00091192

Attribute	Opening Figure Trailer Message TranType	Opening Figure Data Messages
CurrentBTSBFFigures Trailer	CurrentBTSBFFigures Trailer	BTS line value messages. Used by the BTS for suspense brought forward (Lines BBB and BBBB). Written when rolling into the first TP, and on subsequent rollovers.
DeclarationTrailer	DeclarationTrailer	Keys office opening declarations: messages with <Container:##>. Written only in CAP mode; i.e. not written on rollover in to first TP and subsequent rollovers.
OpeningFigures	CAP mode: OpeningFiguresTrailer TP mode: SuspenseTrailer	In TP mode (i.e. on rolling over into first TP) this only keys suspense brought forward figures; messages with <Container:##> and EPOSSTransaction.SuspenseContainer:$$>. In CAP mode it also keys stock holdings for all stock units; messages with <Container:##>.

The Cash Account Trailer is only written until the final cash account is produced.

After the above ## figures are written, #1 figures are also written. The latter are a summation of the figures from the various stock units, unlike the ## figures which are a concatenation. The #1 figures are used by the Office Snapshot and other office wide searches and reports. No trailer points to the #1 figures; the OpeningFiguresId is found by scanning for the first #1 message from the CAP marker.

The details of the above messages are given in [EA/LLD/004].

4.3.5 Reference Data

The following new/changed persistent objects are used:

- EPOSSStockUnit/Parameters includes new local suspense configuration
- EPOSSStockUnit/OfficeVariances defines suspense and transaction correction queries

The following collections remain central to the use of EPOSSStockUnit:

- EPOSSCAP

FUJ00091192

- EPOSSDNodes
- EPOSSNodes
- EPOSSProducts
- EPOSSStockUnit
- StockUnits
- StockUnitMarkers
- StockUnitBPMarkers

4.3.6 Troubleshooting

The major differences between the Stock balancing rollovers produced pre-S80 and those at S80 lie in the production of:

4.3.6.1 Figures relating to Volume Stock

At point 50 only, opening figures are converted from sale value type stock to volume stock only where the stock is indicated as being volume stock. Should the reference data for a product arrive late when the stock unit rolls into the first TP, then there is danger that a product could roll with a combination of volume and a residual sale value also. This situation will not be easily rectified without a special transaction introduced that will have zero quantity which does not disturb the stock holdings, but has a contra sale value will eliminate the residual value carried forward. Such a transaction would have to be specially prepared by SSC staff and run in with the likes of the Riposte tool 'rclient.exe'; this action might need POL authorisation.

4.3.6.2 New figures and trailers for the 'Branch Trading Statement'.

The Stock unit rollover process will feed figures through to the Branch Trading Statement report, see 4.7.6.3.1 for details of these new trailers. Figures are prepared in line with table objects within the collection BTSSourceData. The figures for each line are based on a formula for one or more EPOSS Nodes accumulations; these nodes may be a variety of accumulate type based on sale value, quantity or row count. Brought forward figures (i.e. previous period's figures) are calculated in the previous rollover to current, but are re-linked into the current period's figures. Therefore the figures for the BTS must have the correct BTSSourceData in place at the very least in the CAP prior to point 50, otherwise the brought forward figures will not be accurate.

For fault diagnosis, in CAP mode the pre S80 CashAccLines are present. In TP mode, these are no longer produced and the BTS rollover trailers and BTSSourceData must be examined (see section 4.7.6.3.1).

FUJ00091192

4.3.6.3 Last Cash Account

At point 50 a one off trailer is written that points to a set of opening figures specially prepared for the last Cash Account. Figures that would normally be shareable between the existing balance snapshot/balance report process (for subsequent actions after a rollover) and the Cash Account diverge after point 50, hence the need for the last trailer. In the CAP Rollover trailer at point 50 a trailer will exist called the "OpeningCAProductionFiguresOnly" to supplement (and not replace) the "OpeningFigures" trailer.

4.3.6.4 Diagnostics Written

- Windows events are written – see section 5.1.
- Audit log lines are written – see section 5.2.

4.3.6.5 Tools

4.3.6.5.1 Check Balance

Pre-S80, there were three previous means of reconciling the balancing process against the counter balancing code:

- CheckBalance tool.
- EOD counter cash account reconciliation (CASEPOSSDilayRecon and CASEPOSSWeeklyRecon).
- Agent reconciliation based on harvested Cash Account trailers, i.e. the host recalculated cash account.

All of these reconciliation means are no longer available with S80 working in TP mode.

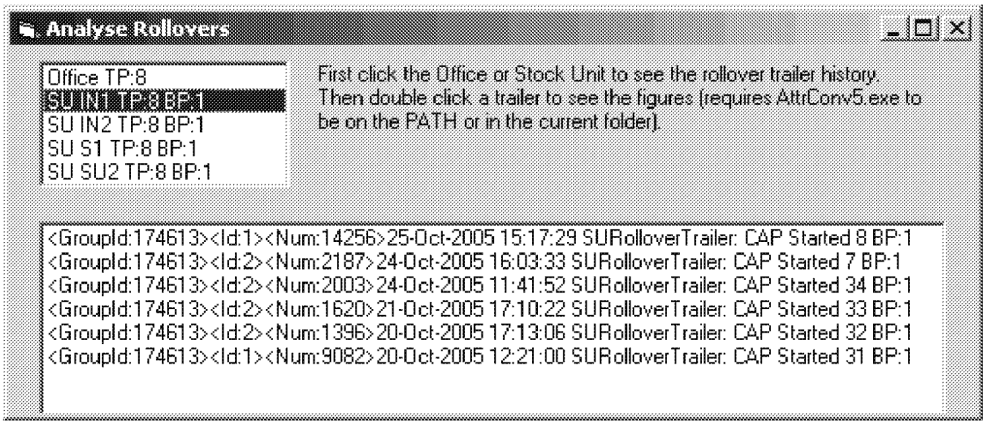
4.3.6.5.2 Analyse Rollovers

The tool AnalyseRollover trailers can be used on a loaded message store to show:

- The current CAP/TP and BP for the office and each Stock Unit.
- The previous rollovers for the office or a selected Stock Unit.
- The rollover trailers for a selected rollover together with their opening figures (CAP mode specific trailers are not shown – this tool is intended for analysing TP mode rollovers).

FUJ00091192

e.g.



4.3.6.6 Fault Diagnosis

- Use the above AnalyseRollovers tool to identify the message range for the CAP/TP and BP in question.
- The actual transactions that have been performed during a Stock Unit CAP/TP need to be analysed (as pre S80) in order to check the values for a carried forward figure or an accounting node value.

FUJ00091192

4.4 In Day Transactions

4.4.1 Purpose

In order to implement the high level requirement to handle most stock by volume rather than by value, changes are needed to the transaction attributes for Volume Stock.

The following changes are needed to the way transactions are conducted when in TP mode:

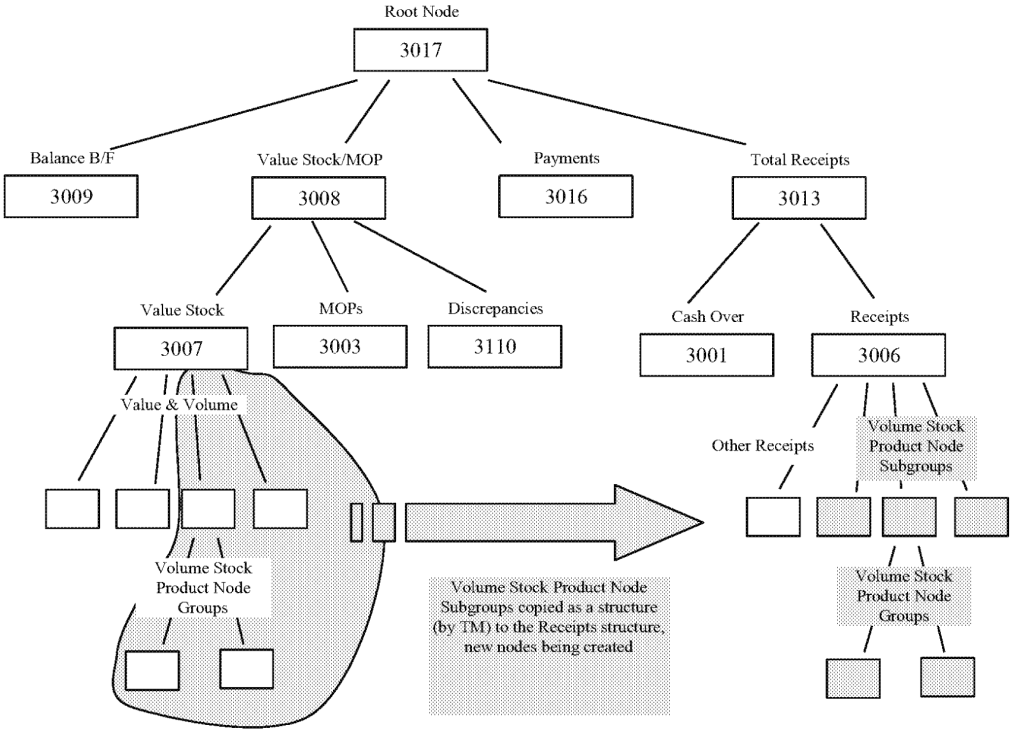
- Products now handled by volume have a new Tertiary Mappings (TM) attribute, which contains a set of accounting node mappings into the Receipts accounting hierarchy. In TP mode these mapping are used instead of the existing Primary Mappings (PM) attribute which maps to the Value Stock & MOP accounting hierarchy.
- Volume stock movements (i.e. stock transacted in Remittance or Transfer modes) record a zero sale value.
- Settlement products used in transactions will now be POL products rather than POA products.
- APS transaction will now use EPOSSCore in order to utilise the above new volume stock changes.

4.4.2 High Level Design

The high level design in given in [EA/HLD/005] section 5.1.4.

The following diagram (adapted from [EA/HLD/005] section 5.1.7) shows the accounting hierarchy and the use of Tertiary mappings.

FUJ00091192



The following diagram shows the various stock types.

I:True **Inventory Stock**		I:False VolS:False **Non-inventory Stock** (no intrinsic value until issued. Accounting is only about the sale value not the stock) PM maps into 3006 or 3016 No TM
VolS:False ***Value Stock*** PM maps into 3008 & below, and 3013/3016 & below No TM	VolS:True ***Volume Stock*** PM maps into 3008 & below TM maps into 3013 & below	

4.4.3 Low Level Design

The low level design is given in [EA/LLD/005].

The changes are implemented in components:

- EPOSSCore

FUJ00091192

- EPOSSSettlementObject
- APS [outside the scope of this document]

[EA/HLD/005] and [EA/LLD/005] define which change is made in which component.

The existing but undocumented EPOSSCore interface for obtaining transaction attribute grammar stings and selling a product is now documented in [EA/LLD/005].

4.4.4 Message Store Messages

Example transaction messages are given in [EA/LLD/005].

Volume stock transactions include the attribute "EPOSSTransaction.VolS:True".

4.4.5 Reference Data

No new collection or objects are introduced, but the existing collections remain relevant (e.g. EPOSSProducts, EPOSSNodes, ModeParameters, ProductModes).

4.4.6 Troubleshooting

4.4.6.1 Diagnostics Written

EPOSSCore writes some error and diagnostic information to the Windows Event log (see section 5.1), the Audit log (see section 5.2), and in addition can write some System Errors (see section 5.3).

EPOSSSettlement has very few occurrences of writing to the Windows Event log, the Audit log, and of writing System Errors.

4.4.6.2 Tools

Riposte tools are available for viewing transactions written to the message store e.g. Riposte Message Spy, Riposte Tail, Riposte Scan Message, Ripose Query, Object Editor, Attribute Convertor etc.

4.4.6.3 Fault Diagnosis

Fault diagnosis is unchanged; the transactions in the message store and the various accounting collections should be examined.

FUJ00091192

4.5 Transaction Corrections

4.5.1 Purpose

A transaction correction request is sent electronically from the central accounting body of POL to a branch in order for the branch to perform a "correction", transacting a supplied product against a supplied settlement product (or other supplied options).

On logon, the user is informed if there are outstanding transaction correction requests that require processing (provided the user has the appropriate role). A button is provided at this point which can be used to enter the Transaction Correction User Interface.

The Transaction Correction User Interface can also be entered via the Housekeeping menu (which is also only available to users with the appropriate role).

The Transaction Correction User Interface lists outstanding transaction corrections, and the user can select one. The details are then displayed and various options provided to the user to process and (in some cases transact) the transaction correction.

Two reports are available on the Reports menu to show Outstanding Transaction Corrections and Processed Transaction Corrections.

4.5.2 High Level Design

The high level design is given in [EA/HLD/006] section 4.5, and the user interface for transaction correction processing is given in [EA/IFS/012] section 2.1.3, 2.4 and 4.3.

The Outstanding Transaction Corrections report and the Processed Transaction Corrections Report are defined from a design perspective in [EA/IFS/012] section 3.4, with examples listed in [SD/DES/005]. See the Reports and Receipts section 4.7.

4.5.3 Low Level Design

The low level design is given in [EA/LLD/008].

The following components implement this functional area:

EPOSSCore	Provides interface for transaction validation.
EPOSSDeclare.	Include Transaction Corrections.
EPOSSStockUnit	Logon check linkage.
EPOSSTxnCorrection	User interface and processing logic.
EPOSSWatchDog	Provide interface to check counter isolation status.
MaintainOfficeVariances	Count transaction corrections processed and write to VarianceHistory objects.

FUJ00091192

If a transaction correction fails to be transacted, a general failure message is displayed to the user.

4.5.4 Message Store Messages

The relevant messages are (see [EA/HLD/006 section 4.10.4] and [EA/LLD/008] for details):

- Transaction Correction Request message (WAIndex.LFSFlag:TC).
- Transaction Correction Processed message (WAIndex.LFSFlag:TP).

The event messages are (see [EA/HLD/006] for details):

- Outstanding Transaction Correction Reminder Displayed (Event ID 62)

4.5.5 Reference Data

The relevant collections and persistent objects are:

- EPOSSStockUnit / TCParams

  Used by Transaction Correction Processing logic and roles. See [EA/LLD/008] for details.

- EPOSSStockUnit / OfficeVariances

  Attributes TxnCorrRequests and TxnCorrProcessed define Riposte queries. See [EA/HLD/006] for details.

- ModeParameters

  Six new modes for transacting Transaction Corrections. See [EA/LLD/008] for details:

  - AN Assign Nominee
  - EV Request Evidence
  - HD Plead Hardship
  - MG Make Good
  - SW Stock Write Off
  - WO Write Off

4.5.6 Troubleshooting

4.5.6.1 Diagnostics Written

No specific diagnostics are written to the event log or audit log.

4.5.6.2 Tools

There are no specific diagnostic tools.

4.5.6.3 Fault Diagnosis

Be aware of the following when diagnosing faults:

FUJ00091192

- It is not possible to reverse a correction transaction. However an erroneous correction transaction could be negated by POL FS producing a fresh correction request with opposite sense.
- The counter is locked during Transaction Correction processing, so that one cannot transfer the session. Locks also prevent someone processing the same Transaction Correction at the same time on multiple counters. It is not permitted to perform Transaction Correction processing on an isolated counter (one that is not, and cannot contact, the Gateway).
- Transaction correction request messages must come from the correspondence server (i.e. node id > 31), and transaction correction processed messages must originate from a counter (i.e. node id < = 31).
- Generated correcting transactions are distinct because they can only be created in one of the specific new modes that are peculiar to Transaction Correction processing.
- Generated correcting transactions are also distinguished by other attributes in messages including a secondary mapping. For a full description and justification see [EA/HLD/006] section 4.5.

FUJ00091192

4.6 Cash Declarations & Variances

4.6.1 Purpose

In order to reconcile cash on a regular basis, the following changes have been made (in summary):

- Cash Declarations will now continue to exist until declared with zero cash, in which case they will be logically deleted at the next desktop load (overnight).
- The Overnight Cash Holding (ONCH) declaration mechanism is replaced with the normal Cash Declaration mechanism.
- New Add/Remove Cash buttons are introduced to allow a cash discrepancy (aka variance) to be rectified at any time prior to stock unit rollover.
- A new Cash Variance button is introduced to check for a cash discrepancy for shared stock units.

A new Cash Variances report was provided and tested, but was dropped by a late POL Change Proposal. The report listed stock unit cash discrepancies, cash declarations, cash variance transactions and transaction corrections. It derived its figures from the EOD process MaintainOfficeVariances (see section 4.9).

In addition, changes have been made to Stock Declaration and Adjustment, Non Value stock handling, and Revaluation.

4.6.2 High Level Design

The high level design is given in [EA/HLD/006] section 4.1, 4.2, 4.3, 4.4 and 4.8. The user interface changes are given in [EA/IFS/012] section 2.1.2, 2.2, 2.3 and 4.2. The button changes are given in [SD/SPE/022].

4.6.3 Low Level Design

The above functionality is implemented in component EPOSSDeclare. The low level design is given in [EP/LLD/013].

4.6.4 Message Store Messages

Declaration objects:

Collection: Declaration_*sss*, ObjectName: *ii_d_nn* (Shared stockunits)

Collection: Declaration_*sss*, ObjectName: *_d_nn* (Individual stockunits)

FUJ00091192

Variance objects:

Collection: Variance_*ww*, ObjectName: FG_*sss*

Collection: Variance_*ww*, ObjectName: FG_*sss*_*ii* (Shared stockunits only)

Declaration messages

Event messages:

ID 58: Remove Excess Cash

ID 59: Cash Made Good

ID 63: Variance check with no Discrepancy

ID 64: Variance Check with Discrepancy

In all the objects and messages described above, the following keys apply:

sss	-	Stockunit identifier
ii	-	Declaration ID
d	-	Drawer item type (2 = Stock, 3 = Stamps, 4 = Cash, 6 = Currency, 8 = Travellers Cheques Unsold)
nn	-	Node Id
ww	-	Week number (calendar week of the year)

4.6.5 Reference Data

The variance history objects are created and maintained by EPOSSDeclare. They are defined in [EA/HLD/006] section 4.10.3. They are held in reference data collection Variance_*ww* where *ww* is the variance week number.

4.6.6 Troubleshooting

4.6.6.1 Diagnostics Written

EPOSSDeclare writes some errors and warnings to the Windows Event log (see section 5.1) and the Audit log (see section 5.2).

4.6.6.2 Tools

The variance week number is in the range 1 through 53. It starts at 1 for the week commencing the first Thursday of the calendar year. It can be calculated for any given date using the VarDate.exe test tool. See [EP/LLD/011] section 14.3.3.9 regarding the Cash Variances Report diagnostics for details.

FUJ00091192

4.6.6.3 Fault Diagnosis

No change to fault diagnosis is needed; the initial requirement is to check the transactions written to the message store.

FUJ00091192

4.7 Receipts and Reports

4.7.1 Purpose

The EPOSS counter produces a wide range of Receipts and Reports output using the:

- Tally Printer (also used as a slip printer for pre-printed stationery and labels)
- Back Office A4 Printer (S80 introduces A4 landscape reports in addition to A4 portrait reports)
- Preview (S80 introduces A4 landscape preview in addition to A4 Portrait preview, Pre-printed stationery Slip preview and Tally Roll preview)

4.7.2 High Level Design

The layout and user interface for reports changed at S80 is given in [EA/IFS/011].

A sample of each report showing the report layout is given in [SD/DES/005].

The high level design of Non-value Stock, Variance, Transaction Correction and Remuneration reporting is given in [EA/HLD/006].

The high level design of Stock Unit, Suspense Account and Branch Trading Statement reporting is given in [EA/HLD/005].

4.7.3 Low Level Design

The EPOSS counter uses four means to display the print user interface and print reports:

- Report Broker, Report Processor, and Global Objects
- BESReports
- LFSConfirm and LFSPServer
- XSLT (Report or receipt defined in XSLT, then printed via Report Processor and Global Objects, using Blank 'Skeleton' Report)
- Printing to the Tally Roll printer and to Preview is via Peripheral Server. Printing to the Back Office printer is via the Visual Basic Printer object to default Windows printer.

There are some circumstances in which ReportBroker handles the initial user interface, and then routes to BESReports for actual printing.

The entire set of reports and the changes made at S80 are listed in [EA/LLD/015]. This defines which reports use the Global Objects route, which use BESReports and which use LFSPServer.

FUJ00091192

The low level design of BESReports is given in [EP/LLD/011]. This has been updated at S80 with an Appendix that lists each changed report and provides detail on the code paths, interfaces and low level design.

The Pre-S80 low level design of ReportBroker is given in [EP/LLD/009], and the Pre-S80 low level design of Report Processor is given in [EP/LLD/010].

Reprints of the final Stock Unit Balance Report and the final Branch Trading Statement are changed to use a BLOB (Binary Large OBject) reprint mechanism (Riposte PutObjectWithASCIIAttachment). The Stock Unit Report BLOB data is written by ReportProcessor, and the Branch Trading Statement BLOB data is written by BESReports. The reprint user interface is handled by ReportBroker, but the BLOBs for both reports are read (using Riposte GetASCIIMessage Attachment) and printed by BESReports.

4.7.4 Message Store Messages

GlobalObjects, LFS and BESReports write EPOSS event messages whenever a report is previewed or printed (<EPOSSTransaction.TranType:E>). In addition completion of report printing writes a PrintInfo message to the message store, with PrintInfo.Success:1 for success and 0 for failure.

4.7.5 Reference Data

The various reports use a variety of collections and persistent objects. See [EP/LLD/011] and [EP/LLD/009] for details.

4.7.6 Troubleshooting

4.7.6.1 Diagnostics Written

The audit log (not the Windows Event log) can contain diagnostic information (see section 5.2). Some reports write significant diagnostic information to the audit log (see the sections under Fault Diagnosis below).

The Peripheral Server log contains the print data sent to the Tally Roll printer (c:\counters\temp\PSStandard.log).

JPEG page images of the last previewed report are held in c:\counters\spool.

4.7.6.2 Tools

See the sections under Fault Diagnosis below.

4.7.6.3 Fault Diagnosis

To determine when a report was printed, and its outcome, check for the report printed events messages in the message store.

FUJ00091192

If the fault is in the user interface, determine whether ReportBroker or BESReports displays the user interface (see [EA/LLD/015] or check the destination component on the menu button).

If the fault is in the report, determine whether BESReports, LFS or GlobalObjects produces the report in question, and whether is it a Back Office or Tally Roll report.

The following sections cover the most significant reports out of those that are new or have been re-written at S80 Impact Release 3.

4.7.6.3.1 Branch Trading Statement

This report (and the associated report / office rollover user interface) is handled by BESReports.

Figures printed on the Branch Trading Statement are derived using equations held in the BTSSourceData reference data collection. Two equations (or "methods") are used for each figure: a source method and a print method. The source method is used by EPOSSStockUnit and contains an equation using accounting nodes and/or products. The calculated value is written out as a BTS trailer during stock unit rollover. The print method calculates the value to be printed using these BTS trailer values and/or values already calculated by previous print methods. For details see [EP/LLD/011].

The BTS has two major sections:

- The Summary section
- The Stock Holdings section

4.7.6.3.1.1 Summary Section

The Summary section reports accounting information with columns for the Branch Total, Suspense and Stock Units. The branch total column is the sum of columns to the right.

Below is an example BTS summary section, spanning two pages.

FUJ00091192

Westwood PAD 242384 Page 1
12:39 29/11/2005 TP 07

Trial Branch Trading Statement - Office Copy From 28/11/2005 To 29/11/2005

	Branch Total	Suspense	SU I1	SU IM1	SU S1	SU SM1
Cash on Hand B Fwd	£1816.88		4580.79	0.00	£237.60	198.00
Cash Awaiting Collection B Fwd	0.00	0.00				
Suspense B Fwd	532.00-	532.00-				
Other MOP B Fwd	0.00		0.00	0.00	0.00	0.00
ForEx B Fwd	0.77		0.00	0.00	0.77	0.00
Other Postage B Fwd	1002.46		554.90	0.00	400.12	0.00
Remittances In Total	0.00		0.00	0.00	0.00	0.00
Cash Rems from SUs	0.00	0.00				
Gains to/from Suspense	0.00	0.00	0.00	0.00	0.00	0.00
Cash Pouches Despatched	800.00		0.00	0.00	0.00	800.00
Transfers In from other SUs	0.00		0.00	0.00	0.00	0.00
Other Receipts	873.68		575.78	264.00	0.24	121.00
Remittances Out Total	800.00		0.00	0.00	0.00	800.00
Cash Pouches Despatched via SUs	800.00	800.00				
Losses to/from Suspense	0.00	0.00	0.00	0.00	0.00	0.00
Cash Rems to Suspense	0.00		0.00	0.00	0.00	0.00
Transfers Out to other SUs	0.00		0.00	0.00	0.00	0.00
Other Payments	1158.78		25.87	264.00	0.77	319.00
Cash on Hand C Fwd	£1367.67		£129.83	0.00	£237.84	0.00
Cash Awaiting Collection C Fwd	800.00-	800.00-				
Suspense C Fwd	532.00-	532.00-				
Other MOP C Fwd	0.00		0.00	0.00	0.00	0.00
ForEx C Fwd	0.00		0.00	0.00	0.00	0.00
Other Postage C Fwd	1167.34		555.77	0.00	400.12	0.00
Total C Fwd	£1203.01	1332.00-	£685.60	0.00	£637.96	0.00
Trading position (+/-)	0.00		0.00	0.00	0.00	0.00
Discrepancy OVER Transferred	45.87		25.87	20.00	0.00	0.00
Discrepancy SHORT Transferred	793.91		0.00	0.00	0.77	244.00
Discrepancy OVER Resolved	45.87		25.87	20.00	0.00	0.00
Discrepancy SHORT Resolved	793.91		549.91	244.00	0.00	0.00
Excess Cash Removed	0.00		0.00	0.00	0.00	0.00
Cash Shortage Made Good	0.00		0.00	0.00	0.00	0.00
Total Branch adjustments	748.04-		524.04-	224.00-	0.00	0.00

Westwood PAD 242384 Page 2
12:39 29/11/2005 TP 07

Trial Branch Trading Statement - Office Copy From 28/11/2005 To 29/11/2005

	SU SH2
Cash on Hand B Fwd	11800.49
Cash Awaiting Collection B Fwd	
Suspense B Fwd	
Other MOP B Fwd	0.00
ForEx B Fwd	0.00
Other Postage B Fwd	47.44
Remittances In Total	0.00
Cash Rems from SUs	
Gains to/from Suspense	0.00
Cash Pouches Despatched	0.00
Transfers In from other SUs	0.00
Other Receipts	87.38-
Remittances Out Total	0.00
Cash Pouches Despatched via SUs	
Losses to/from Suspense	0.00
Cash Rems to Suspense	0.00
Transfers Out to other SUs	0.00
Other Payments	549.14
Cash on Hand C Fwd	11000.00
Cash Awaiting Collection C Fwd	
Suspense C Fwd	
Other MOP C Fwd	0.00
ForEx C Fwd	0.00
Other Postage C Fwd	211.45
Total C Fwd	11211.45
Trading position (+/-)	0.00
Discrepancy OVER Transferred	0.00
Discrepancy SHORT Transferred	549.14
Discrepancy OVER Resolved	0.00
Discrepancy SHORT Resolved	0.00
Excess Cash Removed	0.00
Cash Shortage Made Good	0.00
Total Branch adjustments	0.00

FUJ00091192

The summary section is divided into the following sub-sections, which are separated by blank lines:

Section Description	**Defined by BTSSourceData objects starting with "Line" followed by letter:**
Brought Forward Figures	B
Receipts Movements	C
Payments Movements	D
Carried Forward Position	E
Trading Position	F
Discrepancies transferred to/resolved from local suspense	G
Adjustments to local suspense	H
Total branch adjustments to local suspense	I

The Trading Position should always be zero. A non-zero trading position value in any column indicates a software or reference data fault (e.g. incorrect equations in BTSSourceData, incorrect product PM / SM / TM mappings, incorrect EPOSSNodes, or internal software fault in accounting logic).

Appendix 7.0 explains what each figure in the BTS summary section means in business terms (and makes comparisons to the Cash Account).

4.7.6.3.1.2 Stock Holdings Section

The Stock Holdings section reports the quantities of Volume Stock held by the branch (i.e. the sum of quantities held by all stock units). This is defined by BTSSourceData section object LineX.

This is followed by the number of transaction corrections applied during the TP (defined by objects starting LineY) and the declaration (defined by objects starting LineZ).

Below is an example BTS Stock Holdings Section.

FUJ00091192


```
Westwood                                    FAD 242384                                    Page 3
12:39 29/11/2005                                                                          TP 07
                Trial Branch Trading Statement - Office Copy From 28/11/2005 To 29/11/2005

                                    Stock Holdings at End of Period

      DESCRIPTION          VOLUME          DESCRIPTION          VOLUME          DESCRIPTION          VOLUME

      Game red               30
      Game blue              21
      TV stmp £2             10
      First Day Env         100
      1st class stmp        342
      2nd class stmp        905
      Airletter Pack         13
      Airletter PicPk         1
      Airletter Single       25
      Special 1st           120
      SAE bk 1st & 2Z        10
      Barnsley HCS            1
      Blackburn HCS           1
      PO phonecrd £5         11
      PO phonecrd £10        20
      PO   50p                9
      PO  £1                 20
      PO fee   50p            9
      PO fee  £1             20

Transaction Corrections       0

I certify that the content of this balancing and trading statement is an accurate reflection of the cash and stock on hand at this branch.

                    Signature:

                                        *** END OF REPORT ***
```

4.7.6.3.1.3 Diagnosing Incorrect BTS Values

To identify why a figure is wrong on the BTS:

- Check the BTSSourceData equations used to create the figure and verify they are correct.
- Check the BTS values written on Stock Unit rollover and verify they have been correctly used by the print method and printed on the BTS. The figures read by the BTS from the Stock Unit trailer are printed in the Audit log. See also section 4.3 for details of the Stock Unit BTS rollover trailers and a tool for looking at them.
- Check the BTS values written by the Stock Unit rollover have been correctly calculated by EPOSSStockUnit using the BTSSourceData source method equation. This will involve identifying the transactions and/or opening figures that contribute to a given accounting node or product, and checking the summation is correct. Comparison can be made to the Stock Unit Final Balance Reports (which can be reprinted if necessary) and the Office Snapshot.

  It is vital to understand the difference between the three types of Stock Unit BTS trailer, when they are written and what they are used for. See [EP/LLD/011] for detail, but in summary the BTS trailers are:

BTSCFFiguresTrailer	Written on stock unit rollover and holds values for BTS lines starting C, D, E, G, H, Y.
PreBTSBFFTrailer	Written on stock unit rollover. Not directly used by the BTS, but are used to produce the

FUJ00091192

	CurrentBTSBFFiguresTrailer on the next stock unit rollover.
CurrentBTSBFFiguresTrailer	Written on stock unit rollover as a copy of the values from the previous rollover's PreBTSBFFTrailer. Holds values for BTS lines starting B (i.e. Brought Forward Figures), excluding Lines BBB and BBBB which are held in the office rollover CurrentBTSBFFiguresTrailer.

The last year's worth of Final BTS prints are held in BLOB objects in ref data collection Reprints, and the last BTS can be reprinted from the Reprints menu. The other historic BTS BLOBs could be reprinted by changing the Reprints object being accessed in BESReports.clsBESReports.GenReprintReport via the debugger.

An Excel spreadsheet version of the BTS (linked to the Trial and Final Stock Unit Balance report, and the Suspense Account report) is available.

4.7.6.3.2 Suspense Account Report

This report is handled by BESReports.

This report is now soft configured via reference data collection SuspenseSections. Each section defines the section title and which suspense products should be reported in that section.

If transactions are not appearing on the Suspense Account report it may be that the SuspenseSections has not been updated with the correct products.

Brought forward figures are read from the ## figures written by the office CAP or TP rollover. Suspense movements are read from transactions that have occurred while stock units were in the current CAP or TP. Both brought forward and movement transactions are filtered for primary mapping L2 = 490 or 740.

The various Suspense section totals combined to form the net suspense figure that appears on the Branch Trading Statement. The printed sense (positive/negative) of section values is changed depending on the section so that they are all normally printed positive. However, when summing to derive the net suspense figure the transaction sense is used. Sections with products mapping to node 740 represent Unclaimed Payments (a loss to the branch), whereas sections with products mapping to node 490 represent Uncharged Receipts (a gain to the branch).

See [EP/LLD/011] for details.

4.7.6.3.3 Stock Unit Balance Reports

These reports are handled by GlobalObjects. The SU Balance Snapshot is a different physical report from the SU Trial Balance and SU Final Balance Report. New versions of both these reports were introduced at S80 to allow for the different layouts caused by the introduction of 'Stock by Volume'.

FUJ00091192

4.8 Message Retention

4.8.1 Purpose

A message written to the message store can include an expiry attribute that defines the number of days after which the message expires. Expired messages are archived and deleted from the message store.

Riposte defines a default expiry value for messages that do not explicitly define an expiry value, and a minimum and maximum value that override any explicitly defined values.

The purpose of the change is to:

- Increase the expiry value of messages that need to be retained for a trading period.
- Decrease the expiry value of other messages where sensible so that message store size is minimised.

4.8.2 High Level Design

The high level design is given in [EA/HLD/005] section 5.1.1.

4.8.3 Low Level Design

The low level design in given in [EA/LLD/003].

The following components are impacted:

- BESReports
- EPOSSCommon
- EPOSSCore
- EPOSSDataServer
- EPOSSDeclare
- EPOSSMessage
- EPOSSReportBroker
- EPOSSReportProcessor
- EPOSSSettlement
- EPOSSStockUnit
- EPOSSWatchDog
- LFSConfirm
- NBFinalise

FUJ00091192

- OBCS
- TestPeripherals

The components typically use modCachedData.bas to read TxnExpiry and MinAppExpiry at load time from ref data collection CounterConfigParam in to global variables. The global variables are then used when writing transaction messages (which use TxnExpiry) and other messages (which typically use MinAppExpiry). Some transient messages use a hard-coded value of 1.

4.8.4 Message Store Messages

See [EA/LLD/009] for details as to which messages are impacted.

4.8.5 Reference Data

Collection CounterConfigParams object Counter contains the attributes:

Attribute Name	Value
TxnExpiry	42
MinAppExpiry	20

Although the design permits different values for different components via component specific objects, this is not currently used, and should not be used until PEAK PC0117648 is fixed.

4.8.6 Troubleshooting

4.8.6.1 Diagnostics Written

No explicit diagnostics are written.

4.8.6.2 Tools

No explicit tools are available – the standard Riposte tools for examining messages should be used.

4.8.6.3 Fault diagnosis

If messages are not being written with the correct expiry values:

- Check the expiry value on the message by examining the expiry attribute (e.g. <Expiry:42>).
- Check the correct value for the message type (see [EA/LLD/003]). Not all messages have been changed at S80 Impact Release 3.
- Check the following Riposte configuration values on the counter (via RiposteConfig get <Name>), which can override the value specified by the code.

FUJ00091192

Name	Value
MinMessageExpiry	34
MaxMessageExpiry	50
DefaultMessageExpiry	43
MaxReanimation	49

- Check the expiry values in CounterConfigParams object Counter (see section 4.8.5 above).
- Identify the code writing the message and ensure it is using the correct value.

FUJ00091192

4.9 Maintain Office Variances

4.9.1 Purpose

This purpose of Maintain Office Variances is to analyse transactions that have occurred during the day and create / maintain the Variance History persistent objects.

These objects were used by the now defunct Cash Variances report, and thus strictly speaking there is no need for this functionality. It has not been removed in order to i) avoid the risk associated with a late change when most of the validation has happened and ii) be present in case POL wish to re-instate the report.

4.9.2 High Level Design

The high level design is given in [EA/HLD/006] section 4.1.8 and 4.6.3.

Since this is an end of day process, there is no user interface.

4.9.3 Low Level Design

The low level design is given in [EA/LLD/007].

The design is implemented in component CASEPOSSImpact.exe (for historic reasons – one would expect it to be called MaintainOfficeVariances.exe). This component is executed by the Counter Application Scheduler as part of the EOD Service.

The LLD refers to the Data Protection check being part of this component. This is no longer true as the Data Protection check has been moved to the separate DataProtection component (see section 4.10 below).

4.9.4 Message Store Messages

CASEPOSSImpact writes warning messages and fatal error messages to the message store (see section 4.9.6.1 below).

4.9.5 Reference Data

The variance history objects are used/updated. These are defined in [EA/HLD/006] section 4.10.3. They are held in reference data collection Variance_ww where ww is the variance week number.

Riposte Queries are used that are defined in collection EPOSSStockUnit, object OfficeVariances. See [EA/HLD/006] for details.

FUJ00091192

4.9.6 Troubleshooting

4.9.6.1 Diagnostics Written

Fatal errors are written to the Windows Event log (see section 5.1).

CASEPOSSImpact supports Tunable Tracing, if required. See [EA/LLD/007] for details.

FUJ00091192

CASEPOSSImpact writes warning messages and fatal error messages to the message store. These have attribute <Application:CASEPOSSImpact>. It does not write "informational" messages (for example to log when it started or finished). See [EA/LLD/007] for the message format.

The following table shows the message text (included here since not in LLD):

Reason Number	Type	Code Literal	Text	Comment
1	Fatal	MN_UNUSED		Message id not found in VB Resource File.
101	Warning	MN_MORE_THAN_ONE_INSTANCE	More than one instance of this DLL have been initialised.	More than one instance of the project has been started. Not written as a message – raised as Windows warning event
102	Fatal	MN_FAILURE_TO_SET	The data <*Data*> cannot have the attribute <*AttName*> with value <*Value*> set in it.	Used when there has been a failure setting an attribute.
103	Fatal	MN_ILLEGAL_ATTRIBUTE_VALUE	The <*AttName*> attribute of the <*Data*> data has the value <*Value*> - this is illegal.	Used when the returned attribute value is illegal.
104	Fatal	MN_ILLEGAL_ENUMERATED_TYPE_VALUE	The enumerated type <*Type*> has been passed in the value <*Value*> which is illegal.	Used when the value of an enumerated type is not one of its allowed values.
105	Fatal	MN_ILLEGAL_WEEK_NUMBER	The week number is <*WeekNo*> - it should be between 1 and 53.	Used when a week number supplied is illegal.
106	Warning	MN_ILLEGAL_VARIANCE_OBJECT_NAME	The object name <*Name*> does not analyse as expected.	The supplied object name can not be properly analysed as a collection Variance_ object.
107	Fatal	MN_CANNOT_FIND_OBJECT	The object <*ObjName*> cannot be found from the collection <*Collection*> for reason <*Reason*>.	The persistent object cannot be found.

FUJ00091192

108	Fatal	MN_DUPLICATE_OFFICE	Duplicate Office object found whose data is <*Data*>.	Message produced when a duplicate Office object is found which should be impossible.
109	Fatal	MN_NO_END_OF_DAY_AT_ALL	No End of Day marker has been found at all!	No End Of Day marker has been found at all. This should be impossible since this program is only run if the End Of Day process has run correctly. However it does occur in the very rare case that it is the first day the program has ever run on a new counter and you have disconnected counters since in that case the EOD Marker program runs and produces no Marker; then CASEPOSSImpact runs and incorrectly complains no marker can be found.
110	Fatal	MN_ILLEGAL_VALUE_RETURNED_FROM_TWO_POSSIBLE	The value <*Value*> was returned whereas only <*Value1*> or <*Value2*> should be returned.	Illegal value returned.
111	Fatal	MN_UNEXPECTED_STRING_START	The string must start <*Text*> whereas it actually started <*Text*>.	Unexpected string start.
112	Fatal	MN_NOT_DAY_START	Unrecognised day prefix returned.	The data was not one of Decl_Mon, Decl_Tue etc and could not be converted.
113	Fatal	MN_QUERY_STATUS_UNEXPECTED	The query completed with a status of <*Status*>!	Waiting for the query produced an unexpected result.
114	Fatal	MN_INVALID_PROCESS_KEY	The counter application has failed because the Process Key from the calling process is invalid.	An invalid key has been received from the application scheduler.
115	Fatal	MN_INTEGRITY_CHECK_FAILED	The scheduler integrity check has failed.	The scheduler failed its integrity check
116	Fatal	MN_TIMEOUT_EXTENDED	It has been necessary to extend the timeout period at point	Warning produced when the timeout

FUJ00091192

			<*Point*>.	period is extended
117	Fatal	MN_TIMEOUT_NOT_EXTENDED	Failure to extend the timeout period at point <*Point*>.	Failure to extend timeout period
118	Fatal	MN_UNKNOWN_COMMAND	Unknown command <*Command*> received.	Unknown command received.
119	Fatal	MN_NO_ROLLOVER_BEFORE_MARK	No Office Rollover Trailer found before the mark <*Mark*>.	No Office Rollover trailer found before the supplied marker.
120	Fatal	MN_NO_OFFICE_OPENING_FIGURES_FOUND	No Opening figures found in Office Rollover Trailer before mark <*Mark*>.	No opening figures in Office Rollover trailer
121	Fatal	MN_NO_MARKER_IN_OFFICE_OPENING_FIGURES_FOUND	No marker found in Office Rollover Trailer before mark <*Mark*>.	No marker found in Office Rollover trailer
122	Fatal	MN_VARYING_OFFICE_ROLLOVER_ID	The message Id of Office Rollover messages varies - this is not allowed.	Defensive message when the id of Office Rollover messages vary

FUJ00091192

4.9.6.2 Tools

The variance week number is in the range 1 through 53. It starts at 1 for the week commencing the first Thursday of the calendar year. It can be calculated for any given date using the VarDate.exe test tool. See [EP/LLD/011] section 14.3.3.9 regarding the Cash Variances Report diagnostics for details.

4.9.6.3 Fault diagnosis

A typical approach to troubleshooting involves:

- Check the Windows Event log for error messages. One "red" event log error with source CASEPOSSImpact is written if one or more fatal error messages have been written to the message store.
- Check the message store for warning or fatal error messages (as mentioned in section 4.9.6.1 above).
- Check the contents of the Variance History objects that are updated by CASEPOSSImpact using an object viewer (e.g. Object Editor2.exe). The variance week number is in the range 1 through 53. It starts at 1 for the week commencing the first Thursday of the calendar year. It can be calculated for any given date using the VarDate.exe test tool. See [EP/LLD/011] section 14.3.3.9 regarding the Cash Variances Report diagnostics for details.

The contents of the Variance History objects could also be viewed indirectly via the Cash Variances Report, except that as stated previously this report is no longer used.

FUJ00091192

4.10 Data Protection

4.10.1 Purpose

The purpose of this Data Protection is to avoid Riposte messages from being archived (and hence deleted) in the case where they have exceeded their expiry date, but are still needed (this typically means stock unit EPOSS transactions and related accounting messages that need to be kept until the end of a Trading Period for accounting purposes).

EOD Service logic runs to disable Risposte archiving, if necessary. This DataProtection component is scheduled to run overnight via the Windows NT scheduler ("at" job) at 3:15 a.m., unlike other EOD tasks which are run by the Counter Application Scheduler at 3:30 a.m. (+/- 5 minutes).

In-Day Service logic runs at logon to warn the user that the office should be rolled over, if necessary. This logic is described here, although it is part of the In-Day service not the EOD Service.

4.10.2 High Level Design

The EOD high level design is given in [EA/HLD/006] section 4.6.5 and 4.7, and [EA/HLD/005].

The In Day high level design is given in [EA/HLD/006] section 4.7.1.

4.10.3 Low Level Design

The low level design is given in [EA/LLD/019].

During the End Of Day Service, a check is made. The number of days that have past since the earliest stock unit rolled into the current office CAP/TP is compared with collection CounterConfigParams object Counter attribute RollExpiry (currently 38 days). If the number of elapsed days is great than this, an NT event log error is raised and Riposte archiving is disabled (otherwise Riposte archiving is enabled). This is implemented in component DataProtection.exe.

A further check is made on logon. The number of days that have past since the earliest stock unit rolled into the current office CAP/TP is compared with collection CounterConfigParams object Counter attribute WarnLogonRolloverDays (currently 37 days). If the number of elapsed days is great than this, a warning is presented to the user asking them to rollover the office. This is implemented in component EPOSSStockUnit. The logon warning text is:

> Warning
>
> It is [*Days Since Rollover*] days since stock unit [*Stock Unit Name*] was rolled into the current branch trading period. Please roll the branch into a new period as soon as possible to avoid risk of losing data.

FUJ00091192

4.10.4 Message Store Messages

The EOD Service writes DataProtection error and warning messages the message store. These have attribute <Application:DataProtection>. It does not write "informational" messages (for example to log when it started or finished). See [EA/LLD/019] for details.

4.10.5 Reference Data

Collection CounterConfigParams object Counter attributes RollExpiry and WarnLogonRolloverDays (see section 4.10.3 above).

4.10.6 Troubleshooting

4.10.6.1 Diagnostics Written

Fatal errors are written to the Windows Event log (see section 5.1).

4.10.6.2 Tools

Whether or not the archiver is disabled can be found via command "riposteconfig get DisableArchiving":

C:\counters\bin>**riposteconfig get DisableArchiving**

DisableArchiving = 0

C:\counters\bin>

Any change to this value will not be displayed until the Riposte service has been restarted - for example by the overnight process Cleardesk.

The last run time can be found via command "ripostearchive info":

C:\counters\bin>**ripostearchive info**

Archive in progress = 0

Archive completed = 1

Last end date/time = 2005-04-25 10:06:28

C:\counters\bin>

Riposte archiving can be forced to run by executing "c:\counters\bin\ripostearchive start".

4.10.6.3 Fault diagnosis

- Check the Windows Event log for error messages. One "red" event log error with source DataProtection is written if one or more fatal error messages have been written to the message store.

  The archiver disabled message is "Office Rollover is Overdue - Archiver has been disabled". This is raised each day (at 3:15 am when DataProtection.exe is run) until the check passes successfully.

FUJ00091192

- Check the message store for DataProtection error or warning messages (as mentioned in section 4.10.4 above).
- Check the value of RollExpiry in collection CounterConfigParams object Counter has the correct value (38 days at the time of writing). Archiving could be disabled on the next run of DataProtection.exe by setting this value to –1, but must be reset to the original value once problems are resolved.
- Check the current state of Risposte archiving (see section 4.10.6.2 above).
- DataProtection.exe on single counter offices (SCOs) disables archiving on the RiposteMirror service and well as the main Riposte service.

FUJ00091192

5.0 Errors

Unless otherwise specified, Tuneable Trace is not used by EPOSS components.

5.1 Windows Event Log

This section lists all event log messages for EPOSS components that have changed for S80 Impact Release 3.

Source Application	Event Message	Type	Support Notes
CASEEPOSSImpact	Problem in Main - *<VBErrDescription>*	Error	VB runtime error in modMain.Main
CASEEPOSSImpact	VB failure in psModuleName	Error	VB runtime error in clsErrorHandling.psModuleName reported via fSimpleError. The passed module name failed to be assigned to the error handling object.
CASEEPOSSImpact	VB failure in fsDoASubstitute	Error	VB runtime error in clsErrorHandling.fsDoASubstitute reported via fSimpleError. Substitution failed.
CASEPOSSImpact	Failure to write trace message - *<msg>*	Error	VBTrace returned False in clsErrorHandling.fTrace
CASEEPOSSImpact	Problem in fTraceAndErrorInit - *<VBErrDescription>*	Error	VB runtime error in modTraceAndDiagnostics.fTraceAndErrorInit.
CASEEPOSSImpact	Failure to write trace version	Error	Called from modTraceAndDiagnostics.fWriteVersionInformation. Tuneable Trace call VBTrace returned false.
CASEEPOSSImpact	CASEPOSSImpact has failed - see the message store for more detailed error messages!	Error	Called from modTraceAndDiagnostics. fLogAtMostOneRedEvent. This is an important error as it means that CASEPOSSImpact has stopped working.
CASEPOSSEvProducts	Unable to initialise scheduler object	Error	Called from Load event of form frmEOD. Happens if the function funMain=false.

FUJ00091192

Source Application	Event Message	Type	Support Notes
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 environmental parameters	Warning	Called from funMain in modMain. Happens if the identifier key passed by the scheduler is an empty string.
DataProtection	VB failure in fLogMessage - *<LogMessage>*	Error	VB runtime error in clsErrorHandling.fLogMessage reported via fSimpleError.
DataProtection	VB failure in psModuleName	Error	VB runtime error in clsErrorHandling.psModuleName reported via fSimpleError. The passed module name failed to be assigned to the error handling object.
DataProtection	VB failure in fsDoASubstitute	Error	VB runtime error in clsErrorHandling.fsDoASubstitute reported via fSimpleError. Substitution failed.
DataProtection	VB failure in fsDoASubstitute	Error	VB runtime error in clsErrorHandling.fsDoASubstitute reported via fSimpleError. Substitution failed.
DataProtection	Problem in Main - *<VBErrDescription>*	Error	VB runtime error in modDataProtection.Main
DataProtection	Office Rollover is Overdue - Archiver has been disabled	Error	Called from modDataProtection.fSetRiposteArchiverStatus. Riposte archiving disabled since office rollover is overdue.
DataProtection	Problem in fTraceAndErrorInit - *<VBErrDescription>*	Error	VB runtime error in modTraceAndDiagnostics.fTraceAndErrorInit.
DataProtection	Failure to write trace version	Error	Called from modTraceAndDiagnostics.fWriteVersionInformation. Tuneable Trace call VBTrace returned false.
DataProtection	DataProtection has failed - see the message store for more detailed error messages!	Error	Called from modTraceAndDiagnostics. fLogAtMostOneRedEvent. This is an important error as it means that DataProtection has stopped working.

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSCommon	Unexpected error in Main. Error: *<HexErrNo>* *<ErrDescription>*	Error	VB runtime error caught in top level error handler in modMain.Main.
EPOSSCommon	StockUnitBPMarker object overflow *<NewBPMarkersObjectName>*	Error	Called from clsStockUnitBPMarkers.fbWriteSUBPMarker if the fsPutPersistentObject call fails.
EPOSSAppMain (aka EPOSSCore)	Mode inconsistencies detected on Stack. Previous mode *<<PrevMode>>*, Current Transaction *<<NewMode>>*	Error	Called from modEPOSSHelpers.bCheckStackModeConsistent.
EPOSSAppMain (aka EPOSSCore)	*<ErrorMsg>* was returned trying to read EPOSSStockUnit Parameters	Error	Error returned from GetPersistentObject in clsEPOSS.Initialize.
EPOSSAppMain (aka EPOSSCore)	*<ErrorMsg>* was returned trying to read EPOSSCore ApplicationNames	Error	Error returned from GetPersistentObject in clsEPOSS.Initialize.
EPOSSAppMain (aka EPOSSCore)	Unexpected error in Main. Error: *<HexErrNo>*] *<ErrMessage>*	Error	Called from Initialize in EPOSS class. Used in general procedure error handler.
EPOSSAppMain (aka EPOSSCore)	"Possible index corruption suffered on search for Exisiting Reversal *<Reference>*	Error	Called from clsEPOSS.ProcessReversalReferenceEntered if transaction not found.
EPOSSAppMain (aka EPOSSCore)	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 (aka EPOSSCore)	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 (aka EPOSSCore)	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.

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSAppMain (aka EPOSSCore)	Internally generated error in CallInterface <*ErrDescription*> Command was <*Command*>	Warning	Called from clsEPOSS.CallInterface. An error has already been logged at a lower level.
EPOSSAppMain (aka EPOSSCore)	ProductGroup processing performance. Elapsed Time:<*time*> seconds. (New) ProductGroup objects written to MsgStore:<*NumWritten*> (Redundant) ProductGroup objects deleted from MsgStore:<*NumDeleted*>	Info	Called from clsEPOSS.fReCreateProductGroups to summarise function results.
EPOSSAppMain (aka EPOSSCore)	RP is 0 but MV is not for <*ProductNo*>!	Info	Called from clsProduct.Initialize. RetailPrice is zero but Multiple Value is not - for the named product.
EPOSSAppMain (aka EPOSSCore)	MV not multiple of RP for <*ProductNo*>!	Info	Called from clsProduct.Initialize. Multiple Value is not a multiple of the Retail Price - for the named product.
EPOSSAppMain (aka EPOSSCore)	Product Data for EPOSSProduct [<*ProductNo*>] state a Min Quantity of [<*MinQty*>] and a Max Quantity of [<*MaxQty*>]. An attempt is being made here to transact a quantity of <*Qty*>.	Info	Called from clsProduct.ValidQty. Indicates that an invalid quantity is being attempted to be transacted. Note:The value of MaxQty is currently incorrectly being logged as the value of MinQty.
EPOSSAppMain (aka EPOSSCore)	Class initialisation completed	Info	Called from Initialise in EPOSS class.
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	Error in node retrieval for node(s) <*Nodes*>	Error	Called by modDesktopHelpers.LogNodeError to log node errors.

FUJ00091192

Source Application	Event Message	Type	Support Notes
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. Also called from clsInternalSession.Reset.
EPOSSDataServer	Error in DeleteQuery: < *ResultError*>	Error	A RIPOSTE function. Called from CompletePopulateTree or TidyUpAfterCompletePopulateTree in class InternalSession. Also called from clsInternalSession.TidyUpAfterCompletePopulateTree
EPOSSDataServer	Error in CreateMessagePort: <*ResultError*>	Error	A RIPOSTE function. Called from CompletePopulateTree in class InternalSession.
EPOSSDataServer	Abort retrieval of record after 5 attempts, record reference = <*CurrentRecord*>	Error	Called by clsInternalSession.CompletePopulateTree if ResultSuccessForRecordRetrieval returns false > 5 times.
EPOSSDataServer	Aligment errors have occurred - see audit log for details.	Error	Called from clsInternalSession.LogAlignmentErrors.
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.
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.

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSDeclare	Unexpected error in fProcessAdjustment. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from clsVariances.fProcessAdjustment. Used in procedure error handler.
EPOSSDeclare	<*message*> - returned from EPOSSCore.	Error	Called from clsVariances.fbWriteEPOSSTransaction if response from EPOSSCore transaction is not successful.
EPOSSDeclare	<*message*> - producing message grammar.	Error	Called from clsVariances.fbWriteEPOSSTransaction if response from fsCreateFormattedMessageId is not successful.
EPOSSDeclare	Unable to get unique identifier for DrawerItemDeclarationId. Error: <*ResultError*>	Error	Called from CommitList 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 from CommitList in EPOSSDeclare class. Happens when calls to CreateDrawerItemDeclarationDetail, CreateDrawerItemDeclaration or CreateONCHDeclaration do not return a successful result.
EPOSSDeclare	Unable to create drawer item declaration. Error: <*ResultError*>	Error	Called from CommitList in EPOSSDeclare class. Happens when calls to CreateDrawerItemDeclaration does not return a successful result.
EPOSSDeclare	Error connecting to OLE Server <*ClassName*>. [0x<*HexErrNo*>] : <*ErrMessage*>	Error	Called from general procedure error handler in clsEPOSSDeclare.Initialize or clsEPOSSDeclare.Connect.
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	Error in Transacting ProcessAdjustments :- (<*Command*>)	Error	Called from clsEPOSSDeclare.ProcessAdjustments if the call to EPOSSCore is not successful.
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.

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSDeclare	Unexpected error executing fsCalculateRate - see audit log for details.	Warning	Called from general procedure error handler in clsEPOSSDeclare.fsCalculateRate.
EPOSSDeclare	Unexpected error executing fCacheSpotRates - see audit log for details.	Warning	Called from general procedure error handler in clsEPOSSDeclare.fsCacheSpotRates.
EPOSSDeclare	Warning: An attempt was made to invoke a public interface before the application has been initialised.	Warning	Called from CallInterface from EPOSSDeclare class. Happens when the flag fAppStarted=false.
EPOSSMessage	EPOSSMessage.LogStandardEvent() failed. Failed to find Persistent Object: [Collection:Events].[Object:<*EventID*>].	Warning	Called from clsEvent.LogStandardEvent if LoadStandardEvent returned false.
EPOSSMessage	EPOSSMessage.fbLogStandardTransaction() failed. Call to RetailBroker.Transaction(1) failed: " RetailBroker returned the following message: <*ErrMessage*>.	Warning	Called from clsEvent.fbLogStandardTransaction if RetailBroker write message fails.
EPOSSMessage	EPOSSMessage.LogSystemEvent() failed. Failed to find Persistent Object: [Collection:Events].[Object:<*EventID*>].	Warning	Called from clsEvent.LogSystemEvent if fbLoadSystemEvent returned false.
EPOSSMessage	EPOSSMessage.fbLogSystemTransaction() failed. Call to RetailBroker.Transaction(2) failed: " RetailBroker returned the following message: <*ErrMessage*>.	Warning	Called from clsEvent.fbLogSystemTransaction if RetailBroker write message fails.
EPOSSReportBroker	Unable to get connection status to node <*NodeID*>	Error	Called from CheckNeighboursConnected in forDesktop class. (Report Broker object, not Report Processor)
EPOSSSettlement	Unexpected error in Class Initialize. Error: [0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from procedure error handler in clsEPOSSSet.Initialize.

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSSettlement	Unable to autosettle a <mode> transaction.	Info	Called from clsEPOSSSet.ProcessFinish if objDesktop.IsLockedLogout is True.
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.
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.
EPOSSStockUnit	Could not get Config Info for Cash Account ID - *<dd/mm/yyyy hh:mm:ss>*	Error	Called from modDesktopHelpers.GetUniqueId if call to TALOP_GET_CONFIG_INFO fails three times.

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSStockUnit	Unexpected error in fsCurrentCAPRolloverMsgId. [0x<*HexErrNo*>] : <*ErrMessage*>	Error	Called from procedure error handler in modDesktopHelpers.fsCurrentCAPRolloverMsgId.
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	Bureau Parameters not Found in collection EPOSSStockunit/Parameters	Error	Called from clsOutlet.Initialize if required Bureau data not found.
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.
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 executing command <*message*> - see audit log for details.	Error	Called from modDesktopHelpers.bCheckResult if supplied result is not success and bLogErrorInEventLog=True.
EPOSSStockUnit	An error has occurred within modDeskTopHelpers.bIsNonValue. Error No: <*ErrNo*> Error Desc: <*ErrDescription*>	Error	Called from procedure error handler in modDesktopHelpers.bIsNonValue.

FUJ00091192

Source Application	Event Message	Type	Support Notes
	Source: <*ErrSource*>.		
EPOSSStockUnit	Unexpected error in fsSetBTSTrailers.[0x<*HexErrNo*>] <*ErrMessage*>	Error	Called from procedure error handler in modDesktopHelpers.fsSetBTSTrailers.
EPOSSStockUnit	StockUnit Persistent Object Failure, Commiting SU <*SUName*>	Error	Called from clsStockUnit.Commit if written persistent object cannot be read back.
EPOSSStockUnit	ERROR: 3 - Missing reference data - Opening Periods::OfficeHours	Error	Called from clsOutlet.Initialize if required Weekday information cannot be found.
EPOSSStockUnit	ERROR: 3 - Missing reference data - Outlet::OutletDetails	Error	Called from clsOutlet.Initialize if required OutletDetails information cannot be found.
EPOSSStockUnit	Unexpected error in clsOutlet.Class_Initialize - <*err.Description*>	Error	Called from procedure error handler in EPOSSStockUnit.Initialize.
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.
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*>]	Error	Called from SetOpeningFiguresTrailer in classes Rollover and Outlet. Used in procedure error handler.

FUJ00091192

Source Application	Event Message	Type	Support Notes
	<ErrMessage>		
EPOSSStockUnit	Unexpected error in SetNonInventoryStockTrailer.[0x*<HexErrNo>*] *<ErrMessage>*	Error	Called from SetNonInventoryStockTrailer in class Rollover. Used in procedure error handler.
EPOSSStockUnit	StockUnitBPMarker collection overflow *<ObjectName>*	Error	Called from clsRollover.UpdateSUBPMarkers if the StockUnitBPMarker collection exceeds the Riposte max message size.
EPOSSStockUnit	Unknown error in UpdateSUBPMarkers *<ErrDescription>*	Error	Called from procedure error handler in clsRollover.UpdateSUBPMarkers.
EPOSSStockUnit	SU:clsRollover.fCreateBTSSummaryLines() [ERROR:*<ErrNo>*:*<ErrDescription>*]	Error	Called from procedure error handler in clsRollover.fCreateBTSSummaryLines.
EPOSSStockUnit	There has been a problem carrying forward declarations on inactive stock unit rollover - see the audit log for today for details!"	Error	Called from procedure error handler in clsPreviousDeclarations.fUnexpectedError.
EPOSSStockUnit	Unexpected error in CheckStockUnitRolloverDate.[0x*<HexErrNo>*] *<ErrMessage>*	Error	Called from CheckStockUnitRolloverDatein class clsOutlet. 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	Revaluation Error executing	Error	Called from clsOutlet.fbProcessBureauTxns if Abs(RevaluedSV) > MxV

FUJ00091192

Source Application	Event Message	Type	Support Notes
	fbProcessBureauTxns - *<ErrDescription>*"		
EPOSSStockUnit	Mode Error in fbProcessBureauTxns - see audit log for details.	Error	Called from clsOutlet.fbProcessBureauTxns if CheckModesForProduct returns False.
EPOSSStockUnit	Unexpected error, Cannot Rollover Default StockUnit	Error	Called from ProcessOfficeBalanceConfirmation 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	SU:ProcessFinalBalance - BDC Discrepancy Value too large to transact :*<DiscrepancyValue>*	Error	Called from clsOutlet.ProcessFinalBalance if DiscrepancyValue > MxV
EPOSSStockUnit	Error attempting to get Rate info. for discrepancy transaction : *<message>*	Error	Called from clsOutlet.ProcessFinalBalance if RateInfo is null.
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 current marker. Error: *<ErrorMsg>*	Error	Called from clsOutlet.OutstandingSummaries if TALOP_GET_MARKER call fails.
EPOSSStockUnit	No CAMappings retrieved(1)	Error	Called from clsOutlet.ProcessNewCashAccount if no cash account mappings were retrieved.
EPOSSStockUnit	No CAMappings retrieved(2)	Error	Called from clsOutlet.ProcessNewCashAccount if no cash account mappings were retrieved.
EPOSSStockUnit	No intermediate ## records returned	Error	Called from clsOutlet.ProcessNewCashAccount if EPOSSStatus.clsForDesktop CreateDataTreeWithRolloverIdForCashAccount does not return ## records.

FUJ00091192

Source Application	Event Message	Type	Support Notes
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	SU:clsRollover.fCreateBTSSuspenseLines() [ERROR:<*ErrNo*>:<*ErrDescription*>]	Error	Called from procedure error handler in clsOutlet.fsCreateBTSSuspenseLines.
EPOSSStockUnit	Unexpected error in fcProcessLocalSuspense: <*ErrDescription*>	Error	Called from procedure error handler in clsOutlet.fcProcessLocalSuspense.
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.
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	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.
EPOSSStockUnit	Mode RD Not permitted for Product <*ProductNo*>, Stock will be eliminated via discrepancy rather than through revaluation and discrepancy	Warning	Called from clsStockFigure.CheckForDiscrepancy if CheckModesForProduct(ProductNo, "RD") returns False.
EPOSSStockUnit	Unexpected error executing fbCurrencyDiscrepancies - see audit log for details.	Warning	Called from procedure error handler in clsOutlet.fbCurrencyDiscrepancies.
EPOSSStockUnit	User <*Username*> Could not be logged on, due to critical reference data missing from Messagestore	Warning	Called from clsOutlet.CallInterface in case “ContinueLogonAfterError”.
EPOSStockUnit	Unexpected error executing	Warning	Caleld from procedure error handler in clsOutlet.fbProcessBureauTxns.

FUJ00091192

Source Application	Event Message	Type	Support Notes
	fbProcessBureauTxns - see audit log for details.		
EPOSSStockUnit	ERROR Double Stockunit Rollover prevented.	Warning	Called from clsOutlet.RollStock.
EPOSSStockUnit	Stockunit <*SUName*> has been set to roll actively, out-of-date stock found in holdings	Warning	Called from clsOutlet.RolloverChecks.
EPOSSStockUnit	Unexpected error executing fPostTxnsToLocalSuspense - see audit log for details.	Warning	Called from procedure error handler in clsOutlet.fPostTxnsToLocalSuspense.
EPOSSStockUnit	[<*CurrentApp*>] Error in CreateQuery(<*Index*>) <*ErrMessage*>	Warning	Called from clsOutlet.OutstandingSummaries if call TALOP_CREATE_QUERY_EX fails.
EPOSSStockUnit	clsOutlet::CAPTrailer - Failed to obtain message for Trailer	Warning	Called from clsOutlet.CAPTrailer if trailer could not be written.
EPOSSStockUnit	clsOutlet::CashAccountTrailer - Failed to obtain message for Trailer	Warning	Called from clsOutlet.CashAccountTrailer if trailer could not be written.
EPOSSStockUnit	SU:clsRollover.fCreateBTSSuspenseLines() [ERROR:<*ErrNo*>:<*ErrDescription*>)	Warning	Called from clsOutlet.fsCreateBTSSuspenseLines if fsSetBTSTrailers returns a non-empty string.
EPOSSStockUnit	Unexpected error fbCheckCap: <*ErrDescription*>	Warning	Called from procedure error handler in clsOutlet.fbCheckCAP.
EPOSSStockUnit	Unexpected error executing fSettleLocalSuspense - see audit log for details.	Warning	Called from the procedure error handler in clsOutlet.fSettleLocalSuspense.
EPOSSStockUnit	Unexpected error executing fCacheSpotRates - see audit log for details.	Warning	Called from the procedure error handler in modDesktopHelpers.fCacheSpotRates. Also called from procedure error handler in fFreecurrencydata and fsCalculateRate (copy and paste coding mistake).

FUJ00091192

Source Application	Event Message	Type	Support Notes
EPOSSStockUnit	NON INVENTORY PM'S (Recovery) <*Index*>	Info	Called from GetNonInventoryPMs in modDesktopHelpers
EPOSSTxnCorrection	Error: Two or more transaction corrections have the same ID (<*CorrectionID*>).	Error	Called from procedure error handler in clsOutstandingCorrn.fbAddedToListOK.
EPOSSTxnCorrection	Unexpected error in Initialize. Error: [0x<*ErrNum*>] <*ErrDescription*>	Error	Called from procedure error handler in clsInterface.Initialize.
EPOSSTxnCorrection	Error connecting to OLE Server <*ClassName*>.	Error	Called from clsInterface.fbBrokerHandleOK if the class handle could not be obtained.
EPOSSTxnCorrection	Warning: Isolation could not be determined so has been assumed. Problem: <*ErrorDescription*>	Warning	Called from clsInterface.fbIsNodeIsolated if the command CMD_TXN_CORR_ISOLATION_Q sent to EPOSSWatchDog was not successful.
EPOSSTxnCorrection	Unexpected error executing fbProcessedCorrectionOK - see audit log for details.	Warning	Called from procedure error handler in clsCorrectionTxns.fbProcessedCorrectionOK.
EPOSSTxnCorrection	Warning: An attempt was made to invoke a public interface before the application has been initialised.	Warning	Called from clsInterface.CallInterface if m_bAppStarted is False.
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	Unable to get connection status to node <*NodeId*>.	Error	Called from clsConnectionWatch.CheckNeighboursConnected if a node connection status could not be obtained.
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.

FUJ00091192

Source Application	Event Message	Type	Support Notes
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	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.
EPOSSWatchDog	Unexpected error executing command <*Message*> - details have already been logged in the audit file C:\temp\AUDIT_<*DayOfWeek*> .LOG	Warning	Called from modDesktopHelpers.fLogErrorInEventLogIfNecessary if mbLogErrorInEventLog is True.
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	Errors reported earlier from fUpdateLANorNWBStatus have now cleared after <*count*> cycles reporting failure.	Info	Called from clsConnectionWatch.fUpdateLANorNWBStatus to report an error cleared.
ReportProcessor	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.
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.
ReportProcessor	Failed to get the current Office CAP used in Office Balancing	Error	Called from clsReport.PrepareReport_ErrHandle if the current office cap is "0".

FUJ00091192

5.2 Audit Log

The audit log contains a variety of information – some useful for fault diagnosis and some just debug "noise".

The audit log is found in folder c:\temp with the name AUDIT_ddd.LOG where ddd is the day of the week in question.

The following information is most likely to be useful for fault diagnosis (concentrating on areas changed). A detailed understanding of the audit messages requires reference to the code.

Component	Information	When logged	Example lines
All	Component name and version.	On component load.	EPOSSReport (EPOSSReportBroker) Version 43.2.1241 [WP21843 Build Thu Mar 31 14:28:37 2005(GMT)]
Various	System error information (see section 5.3).	On system error. Also written as System Error messages to the message store.	14:22:45 SystemError: Application=BESReports.clsBTSReport.fReadLineConfigTable, Code=VB6, Text=Err Description - Overflow
Various	Error reporting	On error (which may not be reported to the user or logged by any other means)	A wide variety of error messages are output when an error is detected. Most (but not all) include the work "error" (search case independently), and if the error was a VB error, most occurrences output the VB error number and description. E.g. BR: ERROR (6) Overflow occurred in clsBESReports.PrintSalesReport BR: ERROR (9) Subscript out of range occurred in clsBESReports.PrintOfficeSnapshot BR::Error in clsBESReports.PrintMailsLabel [6 - Overflow] BR: ERROR (9) Subscript out of rnage occurred in clsBESReports.fProcessTxfrReports BR:utilsBES.SetCutOffMarker [ERROR: Unable to set a Cut-Off marker for ReportID 123]
BESReports	Branch Trading Statement	On preview or print of trial or final BTS.	Starting report: Branch Trading Statement - Office Copy, bTrial=True

FUJ00091192

	diagnostic information.		SU IN1 found CF Line:LineCC, Value:0 SU IN1 read 9 product holding txns, 9 valid, 9 new products Sorting 28 volume product stock holdings across all SUs Finishing: 3 pages BR: Report (Branch Trading Statement - Office Copy) took 2.687
BESReports	Suspense Account Report diagnostics.	On preview or print of the Suspense Account report.	Starting report: Suspense Account - Office Copy Processing section: SectionB Printing section info: lTotalNumTxns=0, dBalanceBF=7 Suppressing empty section Updating SuspenseBF flag to True Finishing: 2 pages Finished successfully BR: Report (Suspense Account - Office Copy) took 2.213
BESReports	BESReports report performance.	On preview or print of all BESReports reports.	BR: Report (Office Snapshot) took 2.474 BR: Report (Miscellaneous Transactions) took 1.682 BR: Report (BTS Reprint) took 3.214 BR: Report (Sales Report - Office Copy) took 4.276
BESReports	Outstanding / Processed Transaction Correction reports	On preview or print of the Outstanding / Processed Transaction Correction reports.	Starting report: Outstanding Transaction Corrections - Office Copy Stored 10 Requested Txn Corr Finishing: 2 pages Finished successfully

FUJ00091192

DataServer	Diagnostics (mostly performance related)	On Preview or Print of DataServer / GlobalObjects report.	DS:: clsInternalSession.CompletePopulateTree: Report [Trial Balance - Office Copy] the executed query took 1.272 seconds. DS:: clsInternalSession.CompletePopulateTree: Report [Final Balance - Office Copy] the executed query took 1.823 seconds. DS:: clsInternalSession.CompletePopulateTree: Report [Balance Snapshot] the executed query returned [13] records.
MiMAN	Diagnostics	On trapping an unexpected VB code error	MI:*{class}.{function-name}* [ERROR: *{err-num}*: *{err-description}*]
MiMAN	Diagnostics	When there is no CAWeeks entry that corresponds to the FINAL_CAP pointed to by the AppConfig data / enabling product	MI: clsOfficeStatus.fInvalidCalendar: No CAWeek for Final CAP
MiMAN	Diagnostics	Having established that MiMAN should be migrating into a CAP, the CAWeek data for that CAP could not be found	MI: clsOfficeStatus.fInvalidCalendar: fbCalculateCAP: No data for CAP *yyyycc*
MiMAN	Diagnostics	Having established that MiMAN should be migrating into a TP, the AccountingPeriods data for that TP year could not be found	MI: clsOfficeStatus.fInvalidCalendar: fbCalculateTP - No AccountingPeriods for year *yyyy* MI: clsOfficeStatus.fInvalidCalendar: fiNextTP - No AccountingPeriods for year *yyyy*
MiMAN	Diagnostics	The AccountingPeriods calendar does not have an entry which covers the date in question	MI: clsOfficeStatus.fInvalidCalendar: fbCalculateTP - No matching Accounting Period for date *dd/mm/yyyy*

FUJ00091192

5.3 System Error Catalogue

The following system errors can be produced from EPOSS components (for an explanation of system errors see [EP/MAN/001] section 8.2.3):

Error Code	Component	Code Literal	Meaning
00001	LFSConfirm	1	The sack was not locked prior to accessing it.
00002	LFSConfirm	2	Failed to create Reversal Transaction.
00003	LFSConfirm	3	Failed to create In Transit Transaction.
00099	LFSConfirm	99	Unrecognised error state.
01001	BESReports EPOSSCore EPOSSSettlement LFSConfirm OBCS	ERR_VBERROR	Unexpected Visual Basic run-time error.
01002	BESReports	ERR_RIPOSTE	Unexpected Riposte error (e.g. on return from Retail Broker Transaction TALOP_GET_ ENTRY, TALOP_CREATE_QUERY, TALOP_QUERY_STATUS)
01002	EPOSSCore	ERR_NOREFDATAMODEPARAMETER	No mode parameter is available for the mode.
01003	BESReports	ERR_BTSMISSINGLINEVALUE	Line value in clsBTSStockUnit.pcolLineValues or clsBTSSuspense.pcolSuspenseValues not found.
01003	EPOSSCore	ERR_NULLMODE	
01004	EPOSSCore	ERR_UNKNOWN_VOLSVALUE	The ModeParameters object quoted has a VolSValue which is not one of Sale, Loss or Zero which are the only recognised values used when transacting volume stock.
01005	BESReports	ERR_BTSUNKNOWNOPERATOR	Unknown node operator value.
01005	EPOSSCore	ERR_NO_CURRENT_CONTAINER	No current stock unit is defined when it should be.

FUJ00091192

01006	BESReports	ERR_BTSUNKNOWNNODETYPE	Unknown node type.
01006	EPOSSCore	ERR_NO_ORIGINAL_MODE	When dealing with an existing reversal transaction it has proved impossible to find the existing mode.
01006	EPOSSSettlement	ERR_MISSING_PERSISTENT_OBJECT	Persistent object not found
01007	BESReports	ERR_BTSUNKNOWNPRINTMETHOD	Unknown print method.
01009	BESReports	ERR_BTSMISSINGCAPROLLOVERTRAILER	SU CAP Rollover Trailer not specified in StockUnit object, or cannot be read.
01011	BESReports	ERR_BTSMISSINGOPENINGFIGURES	OpeningFigures trailer message id not present in CAP Rollover Trailer, or OpeningFigures trailer cannot be read, or OpeningFigures id not present in OpeningFigures trailer.
01012	BESReports	ERR_BTSMISSINGBTSBFFIGURES	BTBFFigures Trailer message id is not present in the CAP rollover trailer, or the BTSBFFigures Trailer message cannot be read, or the BTSBFFigures Id is not present in the BTSBFFigures Trailer.
01013	BESReports	ERR_BTSMISSINGBTSCFFIGURES	BTCFFigures Trailer message id is not present in the CAP rollover trailer, or the BTSCFFigures Trailer message cannot be read, or the BTSCFFigures Id is not present in the BTSCFFigures Trailer.
01014	BESReports	ERR_BTSMISSINGPREVCAPROLLOVERTRAILER	PreviousCAPRolloverTrailer message id is not present in the CAP rollover trailer, or cannot be read (a possible reason is Riposte archiving unexpectedly deleted old messages).
01016	BESReports	ERR_BTSINVALIDTXN	Invalid stock unit opening figure stock holding txn read, or invalid BTS opening figure line txn read.
08001	OBCS	ERR_VBERROR	Unexpected Visual Basic run-time error.
08002	OBCS	ERR_NOREFDATA	Error reading persistent object.
08003	OBCS	ERR_ADMINWRITE	Unable to write Admin message.

FUJ00091192

6.0 Support Route

The support route is the PEAK stack QFP.

FUJ00091192

7.0 Appendix – Explaining the BTS

This Appendix is the contents of an informal document authored by Phil Hemingway. This pre-dates BTS CP 4002, which caused the following changes regarding cash in pouches:

- "Gains to/from Suspense". Does not include any cash in pouches transactions.
- "Losses to/from Suspense". Does not include any cash in pouches transactions.
- "Cash Rems from SUs" renamed from "Cash Rems Moved to Suspense". Reports cash **put in to** pouches transactions (product 5610) as a suspense figure.
- "Cash Rems to Suspense". New line, with the same value as "Cash Rems from SUs" but is reported per stock unit rather than as a suspense figure.
- "Cash Pouches Despatched via SUs" renamed from "Cash Collected from Branch". Reports cash in pouches **dispatched** transactions (product 6509) as a suspense figure.
- "Cash Pouches Despatched". New line, with the same value as "Cash Pouches Despatched via SUs" but is reported per stock unit rather than as a suspense figure.

INTRODUCTION

This document attempts to provide a presentation of how the Branch Trading Statement (BTS) can be interpreted, in the absence of the more familiar Cash Account.

It is not the intention to provide an explanation of every formatted field, rather the principle business content.

From the outset similarities can be drawn between the two reports. Firstly they are both Accounting Period End Reports for the Office, though the accounting period has changed from being a one week trading period to a four or five week period. Both reports show the period end branch accounting position as a liability the branch has to POL.

Notwithstanding the difference in format, the main difference between the two reports is that whilst the cash account reports analysed figures of individual transactions and balances by product, sub-totalling them, the BTS only shows summary sub-totals.

As an example of the latter, Table 9 on the cash account reports details of all MOP remittances out, and the subtotal is provided on line 1082 of the cash account, whereas the BTS only reports a sub-total, analogous to if not exactly the content of line 1082 as would have been.

Other differences between the two reports are that the BTS does not report non-accounting data transactions such as those that form entries in table 10g or Parcel Traffic.

In other words the primary purpose of the BTS is to report a liability position and a stock position. In this sense the BTS can be viewed as two separate 'sections', the first reporting a financial movement position and the second the volume stock position.

In interpreting the BTS it is necessary to understand the change at S80 that limits the concept of value stock to just a few items. All other stock is now controlled by volume and reported in quantities held. This change does not alter the fact however that the branch has a liability to POL at the end of each Trading Period, for those items that are classified as value stock.

Hence the two sections of the BTS can be viewed as the first reporting financial liability, and the second volume stock position. This document focuses on the first section of the report.

FUJ00091192

EXAMPLE BTS

An example BTS is provided as follows:

Fishguard FAD 1746138 Page 1
06:59 22/11/2005 TP 08

Trial Branch Trading Statement - Office Copy From 25/10/2005 To 26/10/2005

	Branch Total	Suspense	SU IN1	SU IN2	SU S1	SU SU2
Cash on Hand B Fwd	61843.76		1560.00	270.52-	51259.72	9350.56
Cash Awaiting Collection B Fwd	210.00	210.00				
Suspense B Fwd	724.65-	724.65-				
Other MOP B Fwd	5136.72		140.32	0.00	4996.70	0.00
ForEx B Fwd	12332.98		346.45	0.00	11986.53	0.00
Other Postage B Fwd	405.23		0.00	10.00	395.23	0.00
Remittances In Total	1000.00		0.00	1000.00	0.00	0.00
Cash Rems Moved to Suspense	0.00	0.00				
Gains to/from Suspense	180.99	124.99	56.00	0.00	0.00	0.00
Transfers In from other SUs	0.00		0.00	0.00	0.00	0.00
Other Receipts	607.02		607.02	0.00	0.00	0.00
Remittances Out Total	0.00		0.00	0.00	0.00	0.00
Cash Collected from Branch	0.00	0.00				
Losses to/from Suspense	180.99	56.00	124.99	0.00	0.00	0.00
Transfers Out to other SUs	0.00		0.00	0.00	0.00	0.00
Other Payments	434.02		434.02	0.00	0.00	30.00
Cash on Hand C Fwd	62823.77		1864.51	729.48	51259.72	9270.56
Cash Awaiting Collection C Fwd	210.00	210.00				
Suspense C Fwd	655.66-	655.66-				
Other MOP C Fwd	5266.72		270.32	0.00	4996.70	0.00
ForEx C Fwd	12332.98		346.45	0.00	11986.53	0.00
Other Postage C Fwd	405.23		0.00	10.00	395.23	0.00
Total C Fwd	80383.04	445.66-	2180.48	739.48	68638.18	9270.56
Trading position (+/-)	0.00		0.00	0.00	0.00	0.00
Discrepancy OVER Transferred	30.00		30.00	0.00	0.00	0.00
Discrepancy SHORT Transferred	0.00		0.00	0.00	0.00	0.00
Discrepancy OVER Resolved	30.00		0.00	0.00	0.00	30.00
Discrepancy SHORT Resolved	0.00		0.00	0.00	0.00	0.00
Excess Cash Removed	0.00		0.00	0.00	0.00	0.00
Cash Shortage Made Good	0.00		0.00	0.00	0.00	0.00
Total Branch adjustments	30.00		0.00	0.00	0.00	30.00

FUJ00091192

PRESENTATION VIEW

The view of the BTS to be adopted by this document sees the first section of the report as six distinct groupings of information:

Presentation Group	**BTS Lines (denoted by row headings)**
Brought Forward Figures	Cash on Hand B Fwd
	Cash Awaiting Collection B Fwd
	Other MOP B Fwd
	ForEx B Fwd
	Other Postage B Fwd
Receipts Movements	Remittances In Total
	Cash Rems Moved to Suspense
	Gains to/from Suspense
	Transfers In from other SUs
	Other Receipts
Payments Movements	Remittances Out Total
	Cash Collected from Branch
	Losses to/from Suspense
	Transfers Out to other SUs
	Other Payments
Carried Forward Position	Cash on Hand C Fwd
	Cash Awaiting Collection C Fwd
	Other MOP C Fwd
	ForEx C Fwd
	Other Postage C Fwd
	Total C Fwd
Trading Position	Trading Position (+/-)
Adjustments	Discrepancy OVER Transferred
	Discrepancy SHORT Transferred
	Discrepancy OVER Resolved
	Discrepancy SHORT Resolved
	Excess Cash Removed
	Cash Shortage Made Good
	Total Branch Adjustments

FUJ00091192

The cash account view will be used as the primary source for comparison, and the balance report secondly.

DESIGN VIEW

It's perhaps worth a paragraph to provide an overview of the BTS in design terms.

The design of the BTS utilises the fact that most of the raw data content is already available to EPOSS at the time the Office balance report, in the form of the Accounting Node Hierarchy Dataserver memory tree. This provides all the raw data in the form of node accumulations in memory, largely without change to the existing system.

By specifying the BTS in terms of a set of data sources to be printed reference data can be used to define the BTS in terms of a sourcing and formatting structure. Hence the reference data can instruct data to write out the requisite source values and the report function can similarly use the same data to be instructed to report the source data in certain positions.

The only problem is that dataserver has the data of balances and movements for the current accounting period, and has not got the brought forward position. In other words it has the carried forward position. This is resolved by identifying an accounting period item within reference data for dataserver to forward write items for the following BTS.

Hence the BTS figures in design terms are seen as a set of node accumulations. Extensions are provided allowing nodes to be combined though simple equations, and exception node and product accumulations included or excluded.

It is for this reason that the specification of the BTS in the HLD is provided as a set of source data node derivations.

BROUGHT FORWARD FIGURES VIEW

The brought forward figures view on the BTS should be the same as the carried forward figures view from the previous BTS in all except obviously the very first BTS.

It is the intention of this section to explain the derivation of the first set of figures from the final cash account only.

The first set of brought forward figures represent the branch liability to POL AFTER the final office CAP rollover, the majority of the old value stock holdings now being removed from that liability.

Let's take an example:

<table>
<tr><th>Last Cash Account</th><th>First BTS</th></tr>
<tr><td></td><td>Brought Forward Figures</td></tr>
<tr><td>Table 2 (Page 1)</td><td rowspan="6">Suspense B Fwd -£38, the total gains are offset against total losses, excluding cash in pouches. A gain is reflected as a negative as it reduces the liability the branch has to POL, a loss as a positive figure increases the liability

Cash Awaiting Collection B Fwd £66</td></tr>
<tr><td>Line 5031 Shortages in REMS etc £40</td></tr>
<tr><td>Line 5037 Cash Awaiting Collection £66</td></tr>
<tr><td>Table 3 (Page 1)</td></tr>
<tr><td>Line 5063 Pre-purchase £78</td></tr>
<tr><td>Table 5 (Page 2)</td><td></td></tr>
<tr><td>Line 2050 Cash £1000</td><td>Cash on Hand B Fwd £1000</td></tr>
<tr><td>Line 2051 Cheques £500</td><td>Other MOP B Fwd £500</td></tr>
<tr><td>Line 2052 Fgn Curr Sterl Equiv £125</td><td>ForEx B Fwd £125</td></tr>
</table>

FUJ00091192

Line 2053 Game Licences **£60** (let's say office has 10 game red at £5 each)	No contribution to Brought Forward Figures. There will however be an entry in the Stock Section for Game Red indicating a quantity of 10
Line 2057 Postage Stamps **£302** from Table 5b	
Table 5b (Page 4)	
Line 5011 1st Class Stps **£280** (Let's say office has 1000 1st class at 0.28p each)	No contribution to Brought Forward Figures. There will however be an entry in the Stock Section for 1st class stamps indicating a quantity of 1000
Line 5090 Other Postage Items **£22**	Other Postage B Fwd **£22**

The entries of game licences and 1st class stamps on the final cash account will not contribute to the brought forward figures on the first BTS because they are no longer designated as value stock and are now volume stock.

If nothing else happened in the opening trading period the same figures would be reflected in the carried forward group of the BTS. Hence there would be a total carried forward of £1675. This is exactly the same as the total in Table 5 of the final cash account, and is the balance due to PO on line 1085, using the following equation. Entries in Table 5(£1000, £500, £125, £22) = £1647, plus unclaimed payments, Table 2 (£106), gives Sub Total £1753, less Uncharged Receipts, Table 3 (£78), gives Table 5 Total £1675. Note the items for Game Red and 1st class stamps now volume stock are excluded for reconciliation purposes. It is hence possible to abstract:

- Cash on Hand B Fwd is equivalent to line 2050 on the last cash account
- Cash Awaiting Collection B Fwd is equivalent to line 5037 on the last cash account
- Other MOP B Fwd is equivalent to line 2051 on the last cash account
- ForEx B Fwd is equivalent to line 2052 on the last cash account
- Other Postage B Fwd is equivalent to line 5090 on the last cash account
- Suspense B Fwd is equivalent to the sum of tables 2 and 2a, less line 5037, minus the sum of table 3

A variation from this abstraction is likely explained by products no longer mapping to these lines or becoming volume stock.

FUJ00091192

RECEIPTS MOVEMENTS VIEW

The receipts movements view on the BTS represents the total of movements within the current trading period that either increase the liability of the branch to POL or move some of that liability between stock units and/or suspense. It has some equivalence to what would have been represented on the cash account, but as a comparison the cash account is not as useful.

It is the intention of this section to explain the derivation of these figures.

Let's take an example:

Action/Notional Cash Account Effect	**BTS**
	Receipt Movement Figures
Table 6 (Page 4)	Remittances In Total **£4378**
Line 6014 Cash **£2000**	The sum of the value stock remittances in will be represented in the Branch Total, whilst the stock unit figures will represent whatever part of the total each SU remitted in
Line 6015 Cheques **£2300**	
Table 6a (Page 4)	
Line 6150 Other Postage Items **£78**	
Line 6109 2nd Class Stps **£40** (let's say SU remits in 200 to sell at 0.20p each)	No contribution to Receipt Movements. There will however be an entry in the Stock Section for 2nd class stamps indicating a quantity of 200
Table 2 (Page 1)	
Line 5037 Cash In Pouches **£4000**	Cash Rems Moved to Suspense **£4000** A remittance out to the is value will be reflected in the Payments Movements group, which nets the trading position to zero
Table 3 (Page 1)	
Line 5060 Pre-purchase **£230** (Let's say SUA)	Gains to/from Suspense SUA **£230**
Line 5065 Surplus in Rems **£420** (Let's say SUB)	Gains to/from Suspense SUB **£420**
Line 5046 Cash Shortages A **£115**	Gains to/from Suspense (Susp) **-£115** The suspense column is derived as the sum total of all loss suspense movements for each stock unit, in the payments movements group
	Gains to/from Suspense (Branch) **£535** The signs reflect the line descriptions hence the gains take precedence
Transfer In from other SUs	
There is no equivalence of a transfer to the cash account; as all transfers in and out must have been completed before SU rollover the net movement for the office is nil. Let's say SU A transfers cash **£5100** and game blue quantity 10 to SU B within the trading period	Transfers In from SUs (SU B and Branch Total) **£5100** There will be no contribution of the game blue transfer to Receipts Movements nor to the Stock Total as the net stock movement is zero. Reconciliation of transfers in must be performed

FUJ00091192

Action/Notional Cash Account Effect	BTS
	against the office and SU balance reports
Other Receipts	
There is only a partial equivalence between Other Receipts and the receipts table on the cash account. On the cash account the receipts table reflects all receipts whereas on the BTS it is the entire receipts movement group that reflects entire receipts. Hence other receipts reflects everything but the other line entries of the group, that increases the branch liability (ie the increases the branch value) So Remittances in, transfers in, cash in pouches and suspense items are excluded. Included are local suspense receipt items, sale of non-value stock, sale of non-stock items, also the branch balance forward figure from the previous accounting period. EXAMPLE TO BE PROVIDED	

PAYMENTS MOVEMENTS VIEW

The payments movements view on the BTS represents the total of movements within the current trading period that either reduce the liability of the branch to POL or move some of that liability between stock units and/or suspense. It has some equivalence to what would have been represented on the cash account, but as a comparison the cash account is not that useful.

It is the intention of this section to explain the derivation of these figures.

Let's take an example:

Action/Notional Cash Account Effect	BTS
	Receipt Movement Figures
Table 6 (Page 4)	Remittances In Total **£4378** The sum of the value stock remittances in will be represented in the Branch Total, whilst the stock unit figures will represent whatever part of the total each SU remitted in
Line 6014 Cash **£2000**	
Line 6015 Cheques **£2300**	
Table 6a (Page 4)	
Line 6150 Other Postage Items **£78**	
Line 6109 2nd Class Stps **£40** (let's say SU remits in 200 to sell at 0.20p each)	No contribution to Receipt Movements. There will however be an entry in the Stock Section for 2nd class stamps indicating a quantity of 200
Table 2 (Page 1)	
Line 5037 Cash In Pouches **£4000**	Cash Rems Moved to Suspense **£4000** A remittance out to the is value will be reflected in the Payments Movements group, which nets the

FUJ00091192

Action/Notional Cash Account Effect	BTS
	trading position to zero
Table 3 (Page 1)	
Line 5060 Pre-purchase **£230**	Gains to/from Suspense (Susp) **-£650**
Line 5065 Surplus in Rems **£420**	
Line 5046 Cash Shortages A **£115** (Let's say SUC)	Gains to/from Suspense (SUC) **£115**
	Gains to/from Suspense (Branch) **-£535** The signs reflect the line descriptions hence the gains take precedence
Transfer Out to other SUs	
There is no equivalence of a transfer to the cash account; as all transfers in and out must have been completed before SU rollover the net movement for the office is nil. Let's say SU A transfers cash **£5100** and game blue quantity 10 to SU B within the trading period	Transfers Out from SUs (SU A and Branch Total) **£5100** There will be no contribution of the game blue transfer to Payments Movements nor to the Stock Total as the net stock movement is zero. Reconciliation of transfers in must be performed against the office and SU balance reports
Other Payments	
There is only a partial equivalence between Other Payments and the payments table on the cash account. On the cash account the payments table reflects all payments whereas on the BTS it is the entire payments movement group that reflects entire payments. Hence other payments reflects everything but the other line entries of the group, that reduces the branch liability (ie the reduces the branch value) So Remittances out, transfers out, cash in transit and suspense items are excluded. Included are local suspense payment items, and payment items such as cash withdrawal. EXAMPLE TO BE PROVIDED	

CARRIED FORWARD FIGURES VIEW

The carried forward figures view on the BTS establishes a new branch liability position taking into account the brought forward liability position, together with movements within the trading period that increase the liability and movements that decrease the liability.

The figures are derived by the simply accumulation of corresponding lines in the previous 3 groups and examining the current value stock position.

So:

- Cash on Hand C Fwd is the revised Cash position at the end of the trading period.

FUJ00091192

- Cash Awaiting collection C Fwd adds any additional amount of cash awaiting collection in addition to the brought forward figure, and reduces it by any amount now in transit.
- Suspense C Fwd adds any loss during the period to the brought forward net suspense balance and reduces it by any gains during the trading period.
- Other MOP C Fwd is the revised Other MOP stock figure at the end of the trading period.
- ForEx C Fwd is the revised ForEx stock figure at the end of the trading period.
- Other Postage C Fwd is the revised Other Postage stock figure at the end of the trading period.
- Total C Fwd is the sum of all liabilities above.