

**Oracle® Retail MICROS Retail-J**  
Familiarisation  
Release 12.1

March 2015

Oracle® Retail MICROS Retail-J Familiarisation, Release 12.1

Copyright © 2015, Oracle and/or its affiliates. All rights reserved.

Primary Author: David Nixon

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# **MICROS Retail-J Familiarisation**



# MICROS Retail-J

Familiarisation

July 2014

# Purpose of the Session

- Offers a conceptual framework of Retail-J and the environment in which it operates
- Exposes some of the terminology used within the application
- Drills-down into the some of the key areas of Retail-J

**Introducing Retail-J**  
**Release Components**  
**Deployment Topology**  
**Messaging**  
**Data flows**  
**Notable Tables**  
**Processes**  
**Roles**  
**POS**  
**Cash Management**  
**Card Payments**  
**Inventory**  
**Interfaces**

## Introducing Retail-J

**Note:** The rebranding for the latest version of this documentation set is in development as part of post MICROS acquisition activities. References to former MICROS product names may exist throughout this existing documentation set.

- Retail-J is a POS and Back office store system developed in 2001.
- Online POS, HHTs, laptops and store back office applications are run centrally from Estate Manager, with all parts of the system linked together by Retail-J's integrated communications.
- Estate Manager enables online central management and control of multiple store fascias and formats from a single set of servers.
- Configuration, management, maintenance and support of Retail-J is controlled from the Estate Manager's browser-based user interface.

# Retail-J Highlights

- 3 tier Java application (POS/Store/Estate) for retailing
- Identical software on each tier (switch on what you need)
- In-built resilience to network failures (local storage)
- Near real-time (trickle-feeds move transaction data upwards)
- Portfolio of configurable store applications based on a common set of foundation classes, message processing and data interfaces
- In-built L10n and I18n
- Interface focus consuming and using web services; using file imports and exports as either the recipient or source for third party applications. Can be a full (stand alone) or partial (hub connected and integrated) solution.
- Thick or Servlet POS

- Atelier (workshop tailored goods)
- Hospitality (table layouts, kitchen printing, split billing)
- Airports (sales restrictions, terminal, zone and flight organization for destination pricing)
- Telecommunications (option products and associated workflows, product serial number tracking,)
- Timber Sales (chain of custody)
- Electrical (WEEE charging)
- Fashion Goods (multiple product attributes including range, size, shape, colour; fuzzy search)

# Retail-J Functionality Summary



## Standard POS Features:

- Cash & Tender Management
- Returns
- Price Management
- Discounts and Promotions
- Electronic Voucher Tracking
- Reports – Automatic & Ad hoc
- Credit Authorisation
- Chip & PIN

## Specialist Functionality:

### Product Linking, e.g. phones to accessories

### Workflow Transactions:

- Linking products to contracts
- Linking products to services e.g. insurance
- Customer Surveys
- Liquid Restrictions (airlines)

### Serial Number Capture:

- IMEI numbers
- IMSI numbers

### Commission Selling

### Location Management Profiles

## Enhanced Customer Service Capability:

- Product and Fuzzy Search
- Stock Locator at the POS
- Customer Orders, Quotes, Layaways
- Customer Accounts, Loyalty
- Gift Cards and E Top-Up
- Tax Free Shopping
- Multi-Media
- Weigh Scales

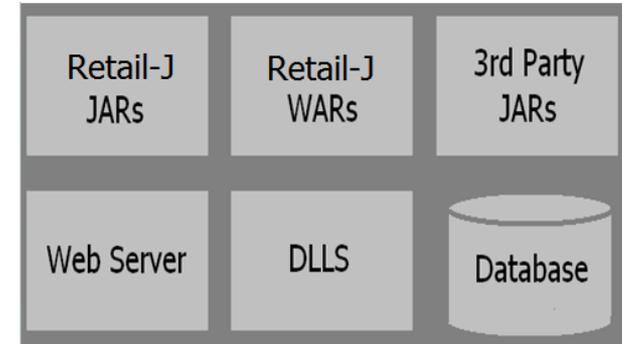
## Store Management:

- Time & Attendance
- Store Inventory
- Stock Transfers
- Stock Counting
- Store Diary
- Employee / User Set up
- KPI's
- Email and Instant Messaging
- CCTV Integration
- Store Manager Portal
- Mobile Devices – HHT's, tablets, PDA's, laptops
- Management Alerts

## Release Components

Name	Date modified	Type
 data	22/05/2014 16:34	File folder
 jdk1.6.0_41	30/04/2014 18:46	File folder
 TOMCAT	30/04/2014 18:29	File folder
 CompanyReg	01/05/2014 18:36	Internet Shortcut
 Device Details	01/05/2014 18:37	Internet Shortcut
 manifest.xml	07/05/2014 15:55	XML Document
 retail-j	01/05/2014 18:37	Internet Shortcut
 StartTPOS.cmd	08/05/2014 20:40	Windows Command Script
 TomcatStart	01/05/2014 18:17	Shortcut
 Upgrade Database.cmd	28/01/2013 17:23	Windows Command Script

# Retail-J Release Components



## Each device will have installed:

- JAVA – dependant on Retail-J configuration a particular version would be used e.g. Java 1.5, 1.6 etc.
- WebServer - (usually Tomcat)
- DLLs
- Retail-J JARs
- Any third party JARs
- Retail-J WARs
- Database – (EM – SQL Server) (Store Server / POS – MySQL)
- Supporting scripts

**The JAR and WAR files contain the code that makes up Retail-J and so all the functionality that comes as standard**

## **JAR files**

- Contain the core code which runs the system(code engine)
- Main JAR file is the rjRetailFoundationClasses.jar
- rjMegaRetailer.jar (for example) contains bespoke development

## **WAR files**

- Contain JSPs (GUI)
- Contain Web Services functions

**All devices contain the same JAR and WAR files**

**JAR files, additional to those provided by Retail-J, are required to:**

- Connect to the database (JDBC drivers for SQL DB - sqljdbc.jar)
- Gain access to FTP services (FTP.jar, FTP2.jar)
- Connection to 3<sup>rd</sup> party systems such as QAS or Solve (qas.jar, SolveLink.jar)
- Support for peripherals (either JPOS or directly driven)

**The exact number and types of 3<sup>rd</sup> party JARs required in any given install depend on the setup of the system**

**Unnecessary and optional JARs should not be put onto a system**

## **The purpose of the Web Server:**

- Serves the Retail-J Back Office UI (Browser based)
- Runs the Retail-J background processes (XML Processing)
- Hosts the Retail-J Web Services

## **The Tomcat Web Server can be started via:**

- A script
- A service

**Usually configured to start automatically (as a Windows service) when a machine is powered up.**

## **DLLs or Windows drivers required in order for Retail-J to interact with other pieces of software and hardware components**

- Win32com.dll required on tills for connection to peripherals via a COM port
- JBrowser.dll required on tills to enable Internet Explorer to be viewable from within POS Application. This is useful to view the BackOffice from within the POS screen

- The manifest file is an XML file which contains the version number of Retail-J
- Used as part of the Software Update Process (SUP)
- The filename of the manifest file is manifest.xml and is located in <RJAppHome>/

## The Retail POS properties file:

- Defines the Application Installation directories
- Defines the unique Device ID identifier, Example - ALL.<ORG\_ID>.0100.101
- Allows the addition of specific debug
- Specifies customer specific functionality
- Customise the POS UI:
  - POS Font Types / Sizes
  - Button Colours
  - POS screen size

**The filename is `com.retailJava.javaPOS.properties` and is located in `<RJAppHome>\Libs1` (locations may vary)**

**Access sets are required to allow Retail POS to connect to the appropriate instance of the database**

**The access sets contain**

- The database connection string ( location of the database e.g. local host and port number)
- The credentials to connect to the database in encrypted format ( user name and password)

**The access set reside in the <RJAppHome>\Data**

**Two files make up the access sets, namely:**

- Access.hdx
- Access.dat

**Access Sets are created as part of the Installation process but can also be recreated manually if the database password is changed**

- Types of database:
  - Operational
  - Audit (Estate Manager only)
- Database agnostic
  - Microsoft SQL Server
  - MySQL
  - Oracle
  - DB2
- Database schema the same across the entire estate
- No referential integrity, No foreign keys.
- Most data stored in XML (CLOB)

- By default is named with the Organisation ID
- Stores all configuration including (amongst others):
  - **Products**
  - **Prices**
  - **Reasons**
  - **Users**
  - **Tendering Information**
- Contains majority of tables
- Present on all thick-client instances of Retail-J

- By default is named with the Organisation ID followed by \_AM  
e.g. DOC\_AM
- Validates Transactions against set rules.
- Validates Cash Management Actions against set rules.
- Used to construct card data for submission (APACS29).
- Used to construct inventory information exports.
- Used to run reports from.
- Schema a cut-down version of the Operational Database
- Only present on an Estate Manager
- Can be written to at the same time as the Operational Database or transactions can be placed in their own processing queue.

# Not Supplied but Essential

- Web server start-up script
- Database upgrade script
- Organisation registration shortcut
- Device registration shortcut
- TPOS start-up script

## Prerequisites

- Installed database
- Licence key
- Organisation ID

## Why does a device need registering?

A device requires registration in order to identify the device within the network and set administrator user credentials.

Re-registering of a device may occur when web cookies have been deleted from a browser

## How to register a device?

- Logon to the back office
- Specify:-
  - Device ID i.e. ALL.TOREX.xxxxx (where x represents the store id e.g. Manc = S4403)
  - Device Name i.e. EM, or Store etc.
  - Retail-J admin username
  - Retail-J admin password
  - Organisation ID i.e. Torex

## Device Registration Responses

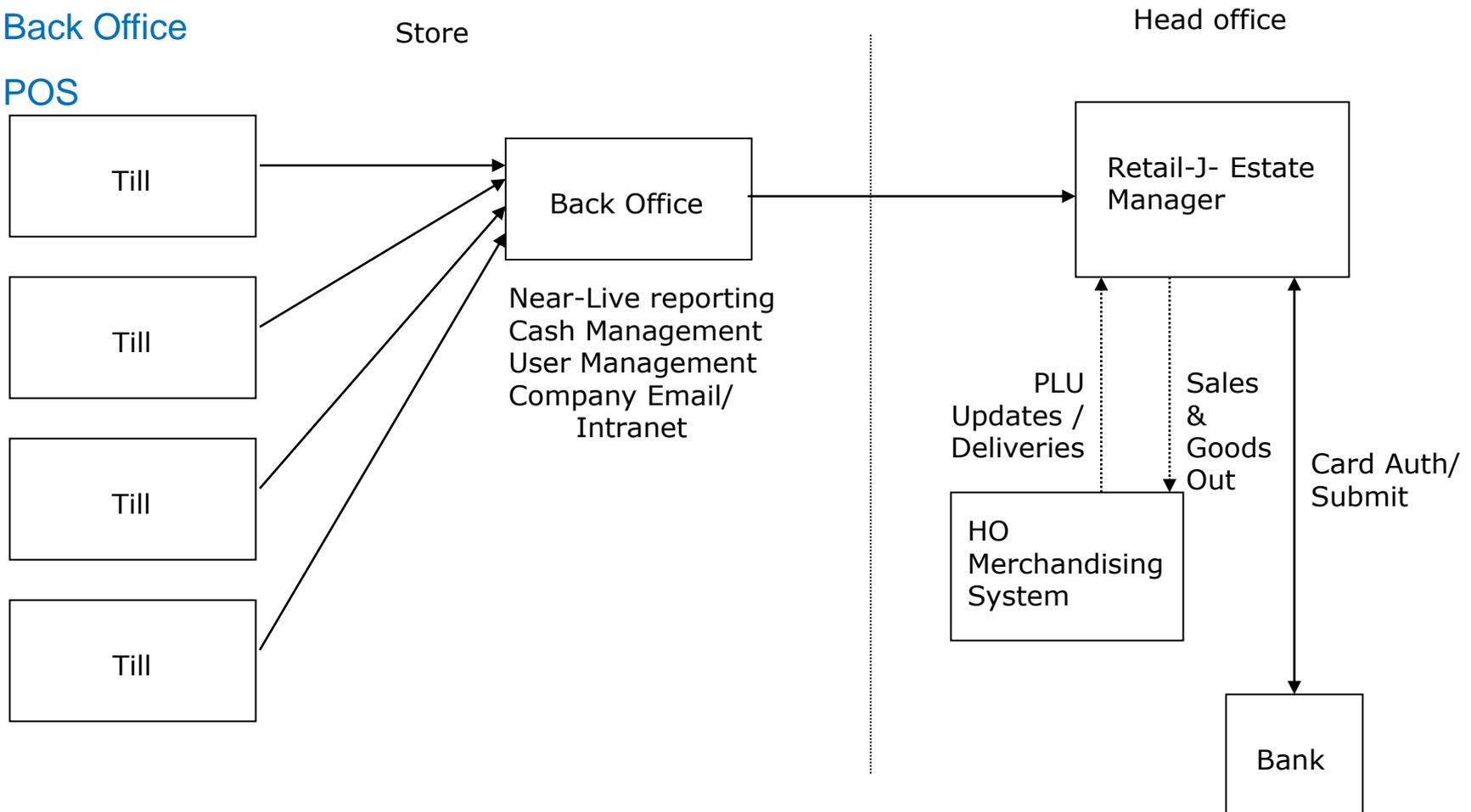
- Device registered successfully in which case refresh the browser to proceed to the logon screen
- Failed to connect to data store
  - Check registration details – most likely incorrect device ID, user/password credentials or organisation ID

# Deployment Topology

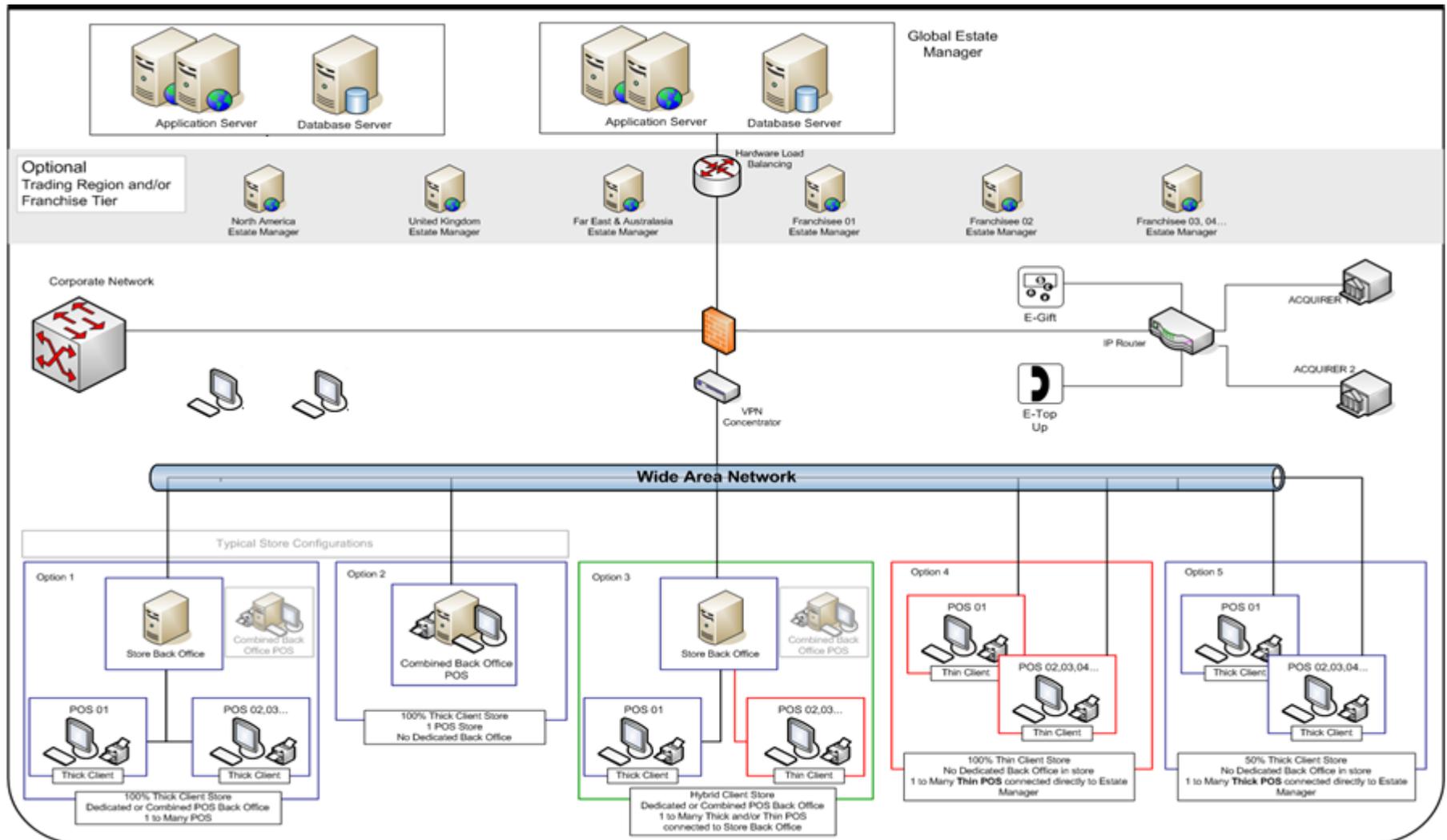
# Deployment Topology

## Retail-J is typically a 3 tier architecture

- Estate Manager (EM)
- Back Office
- POS

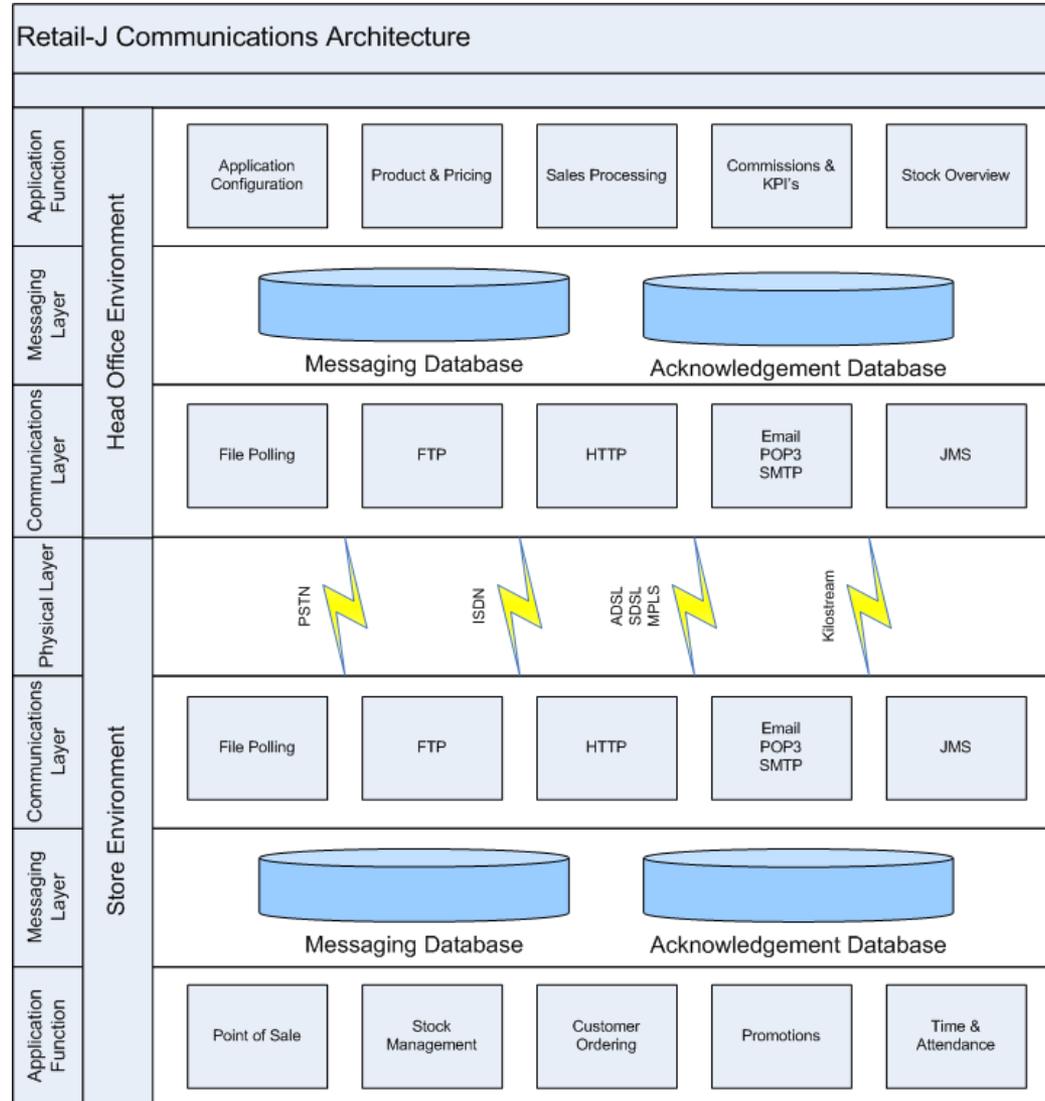


# Deployment Topology in Practice

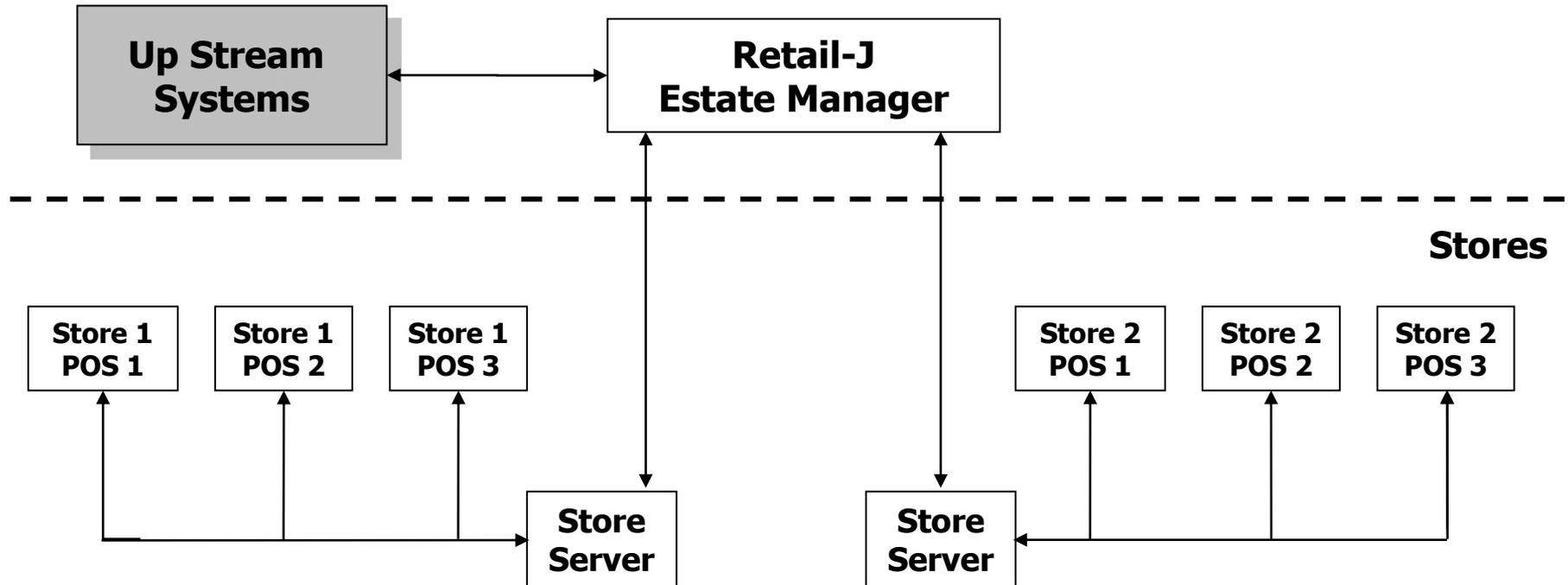


# Messaging

# Communications Architecture



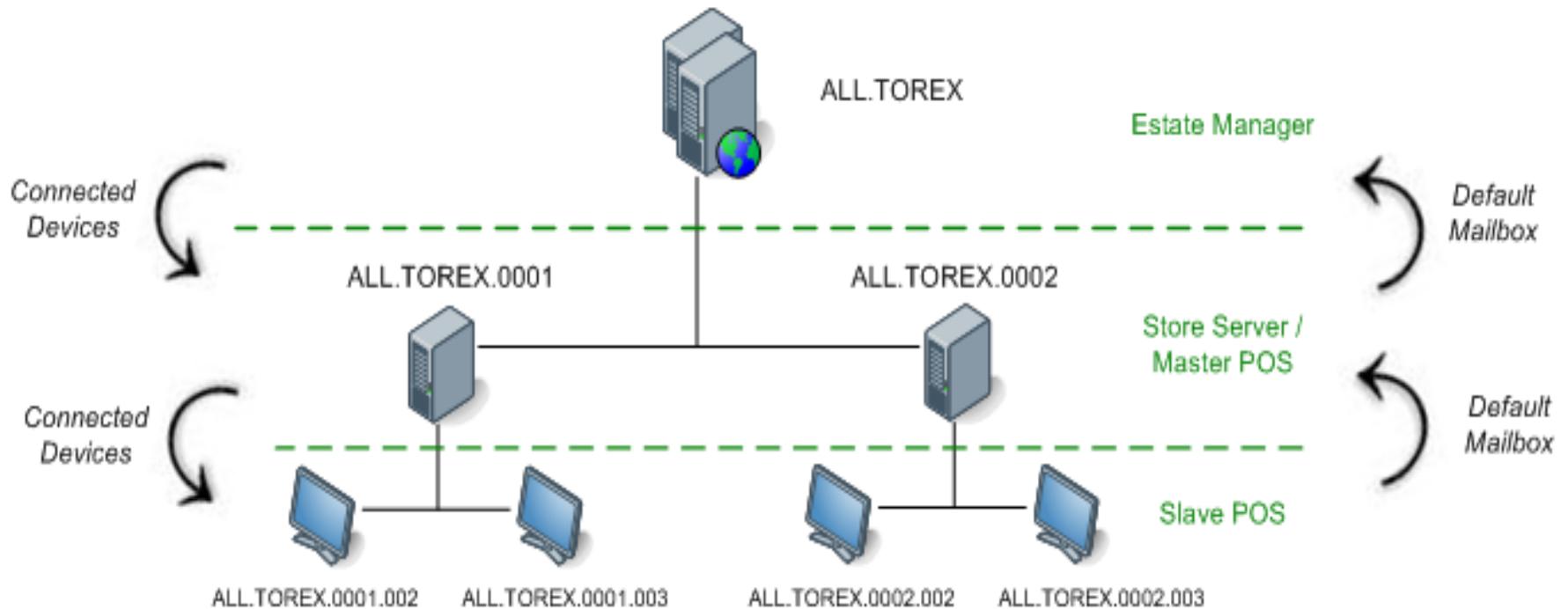
Head Office



## All hardware with Retail-J installed is identified as a Device

<b>Device</b>	<b>Example ID</b>
Estate Manager	ALL.<ORGID> Where <ORGID> is the organisation ID
Store Server	ALL.<ORGID>.XXXX Where XXXX is the store Example: ALL. <ORGID>.0100
Tills	ALL.<ORGID>.XXXX.YYY Where YYY is the Till Example – ALL. <ORGID>.0100.101

# Message Flow



## The Messaging System

- Routes messages to defined mailboxes within Retail-J
- Message – XML entity
- Messenger – the server
- Messaging system supports HTTP(S), (S)FTP and email.
- Messages are accumulated offline and forwarded when a connection is available
- Acknowledgements are sent only when the destination has read and processed the message

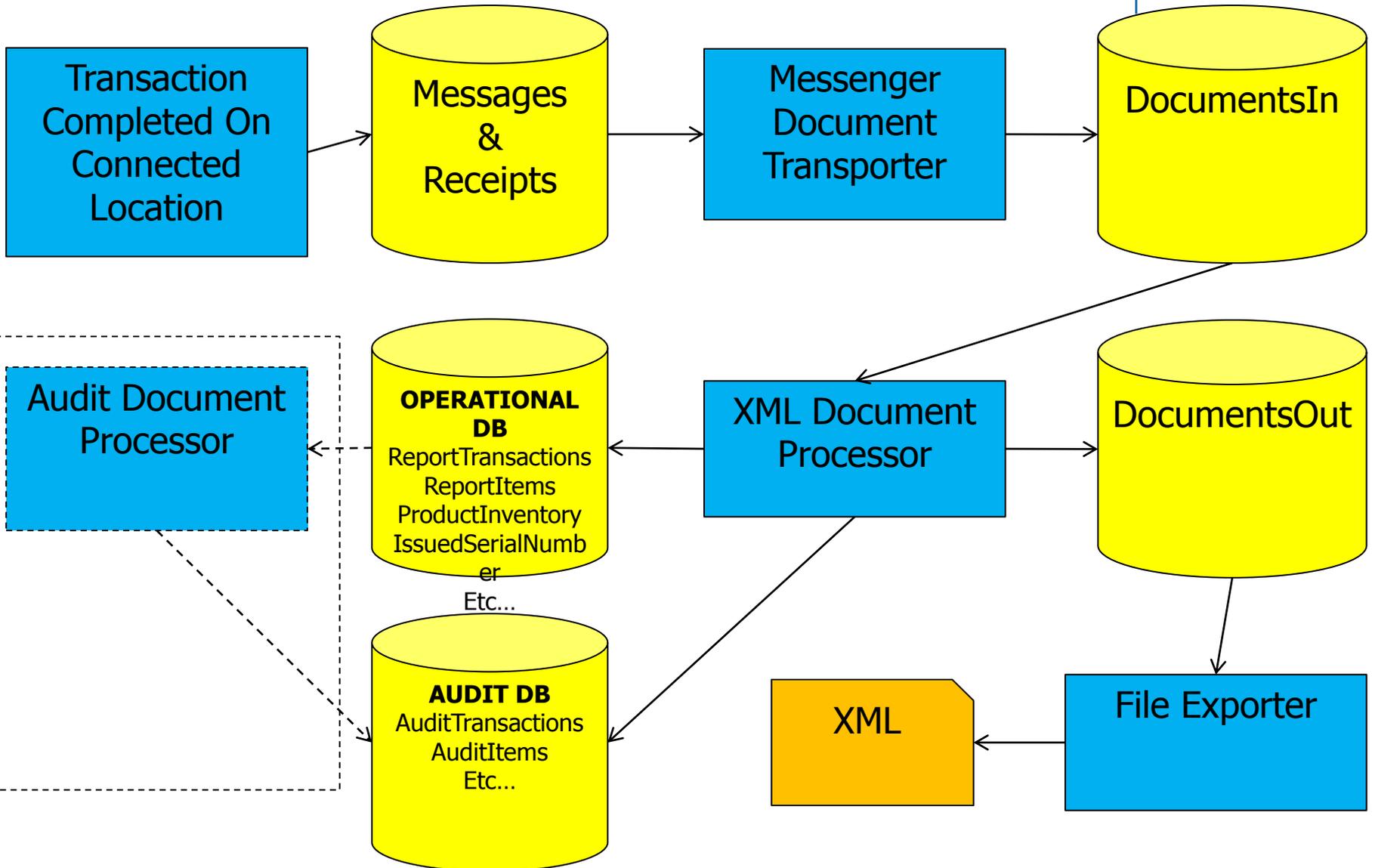
## Setting up Mailboxes and Routing

- Each device has a/an:
  - Default mailbox e.g. ALL.DOC.S0001 (to determine where any unroutable messages should be sent)
  - Local LER mailbox e.g. ALL.DOC.S0001.LER
  - Local XMLPROCESSING mailbox
  - One or more Connected Devices mailboxes to accept messages routed from any mailbox or child of that device e.g. ALL.DOC and ALL.DOC.S0001.T1

# Data Flows

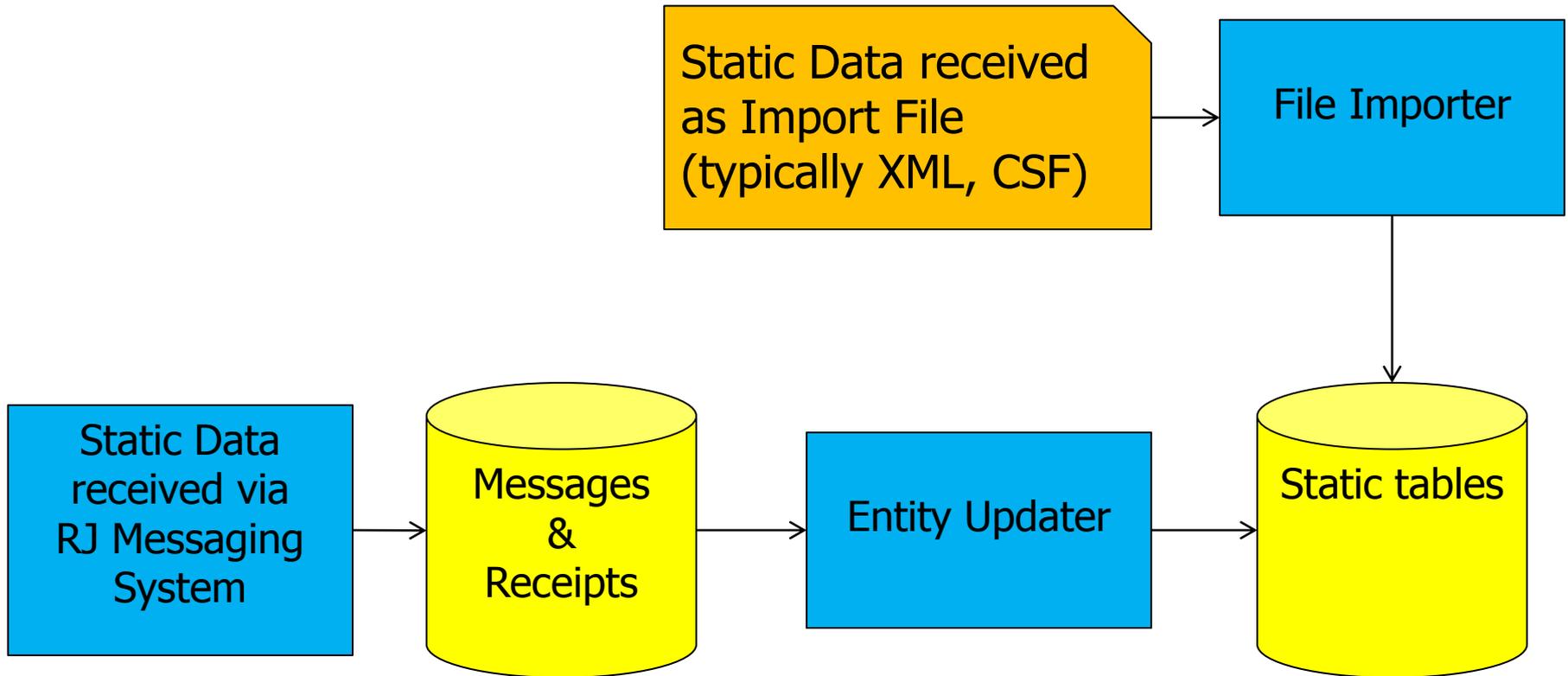
# Retail-J Transaction Processing Data Flow

micros®



# Retail-J Static Data Processing Flow

micros®



- XML Document Processor
  - Processes the documents that have been inserted in to the DocumentsIn table. Once the documents have been processed they are written to the DocumentsOut table ready to be sent to the EM or exported to a third party application.
- Entity Updator
  - Applies the configuration changes delivered from the messaging system to the database. If the entity updator is not running, the database will not be written to. The process picks up messages destined for the 'LER' mailbox.
- File Importer
  - Processes import files containing static data such as a CSF broadcast or product price updates. The files can be in either ZIP or XML format but must adhere to the Retail-J schemas.
- File Document Transporter
  - This process will import files and export files from Documents Out. The Process writes documents directly into and out of Documents In and Documents Out as opposed to processing the files itself.
- Message Document Transporter
  - Responsible for converting the documents into messages

# Notable Tables

- Used to hold the XML data held in the message queue
- Contains both incoming and outgoing data. Both dynamic and static data.
- Messages automatically purged after seven days
- Linked with the Receipts table using the messages.id field and receipts.messageID

- Holds the mailbox details for each message in the receipt table
- Not to be confused with physical receipts from the tills
- Each mailbox has a status:
  - 1 – Unsent
  - 2 – Sent
  - 3 – Acknowledged
- Receipts automatically purged after seven days

- Stores all dynamic data (transactions) for the 'XML Document Processor' to process.
- 'Messenger Document Transporter' writes to the table, inserting the XML of the transaction and setting the status to 1 (Received).
- 'XML Document Processor' reads from the table, processes the transaction then sets the status to 2 (successful)
- Any failures are set to status 3 (failed).
- If database problems (such as locks) are detected, the status is set to 4 (Lock Error). Any transactions with a status of 4 are periodically reset to status 1.

- XML Document Processor writes to this table once the documents in the DocumentsIn table have been successfully processed.
- Contents of this table are identical to DocumentsIn
- Table used to store the outbound transactions
  - If on a Master Till, the transactions may be output or sent to the Estate Manager.
  - If on an Estate Manager, the transactions may be output for the data warehouse or other Head Office system (e.g. SAP)
- Status codes same as for DocumentsIn.

**Stores every basket transaction in XML format**

**Used mainly for refund purposes but also for orders, layaways, gift lists etc.**

**Used as a means to eliminate duplicate transactions being processed**

**Child tables include:**

- BasketExtendedInformation – stores customer details such as name and postcode for the basket
- BasketCards – stores any card details contained in the basket to allow a quick lookup to be performed at the till against a given card number. **Note restrictions on the storage of card numbers and sensitive cardholder data.**

# REPORTTRANSACTIONS



- Used to store transactions for the purposes of reporting against.
- No XML so queries are efficient.
- Most XML elements in a transaction are separated into their own columns.
- Only written to by the XML Document Processor
- Contains the details of the transaction, not the individual items within the transaction.
- Primary key is ReportTransaction\_ID column
- Examples of types of transactions:

<b>POS basket</b>	<b>Cash pickup</b>
Order	Cash reconcile
Layaway	Pickup correction
Product wastage	Income
Cash open session	Expense
Cash Number	

- Child table of ReportTransactions table
- Contains item level transaction details (such as cash payment, card payment, item sale etc)
- Linked to ReportTransactions using the Report\_Transaction\_Id column.
- No referential integrity by design
- No XML

# Processes

**Configured Through:** Administration > Processes > Process configuration

**Stop/Start & Read Logs:** Administration > Processes > Process Management

## Types:

Pre-configured process – set up in process configuration screen

Event driven processes which appear, run and disappear as they are needed

## Run:

Automatic – when Retail-J is first started.

Manual – Through the Process Management screen

When an event occurs – i.e. a broadcast being started.



**Changes to process config only come into effect when the process stopped & restarted**

## Core RJ Database Processes (see earlier slide)

### FTP replicator

- Grabs exported CSF (comma separated) files provided by the estate manager.

### HTTP messenger connector

- Responsible for moving document messages to the EM

### Queued job feeder

- Manages jobs created with the job management function i.e. running of a broadcast or weekend job task.

### Software update process

- Responsible for managing the receipt, activation and successful application of Software Update Files.

# Starting Processes from the Command Line

```
webserver
Process Management Command Line Interface

java ...ProcessManagementClient <device_id> <host_name> <action> <parameter>

where <action> = LIST, START, or STOP
for <action> = LIST, <parameter> = ACTIVE, INACTIVE, or ALL
for <action> = START or STOP, <parameter> = process ID or ALL

The case-sensitive process IDs are :-
AuditDataExtractionProcess AuditDataFinaliseProcess
AuditTransactionProcessingProcess AuditValidators AutomaticReportsProcessor
BatchBroadcasterProcess BatchDocumentExportProcess Broadcaster
CardDataExtractionProcess CardDataOutputProcess CardDataSubmissionProcess
CardSubmissionProcess CashSessionValidator CommissionsCalculatorProcess
DOSPOSListenerProcess DayEndScheduler DayStartScheduler
DownloadSchedulerProcess EmailConnectorProcess EmailDocumentTransferProcess
EmployeeBalanceAdjustmentProcess EncryptionKeyImportProcess
EnhancedHTTPConnector EntityExporterProcess EntityUpdater FTPConnectorProcess
FTPImporterProcess FTPReplicatorProcess Feature Pack Importer
FeatureItemPurgeProcess FeaturePackValidator FileConnectorProcess
FileDistributorProcess FileDocumentTransferProcess FileExportProcess
FileExporterProcess FileImportProcess HTTPConnector HTTPFileTransferProcess
HandPointListenerProcess IndexerProcess InstantMessageBroadcastProcess
InventoryTaskRunnerProcess JMSConnectorProcess JMSExporterProcess
JMSImporterProcess LoyaltyCustomerDetectionProcess
LoyaltyCustomerRegistrationProcess MessengerDocumentTransferProcess
MessengerFileTransferProcess MobilePOSDataExporterProcess POSApplicationProcess
PostponedTransactionValidator PredefinedReportExporterProcess
PrintQueueListenerProcess ProductPriceUpdateProcess PurgeRunnerProcess
QASPostcodeInvokerProcess QueuedJobFeederProcess QueuedJobRunnerProcess
RebuildBroadcasterProcess RemoteTerminalProcess ReplenishmentRequestProcessor
SoftwareUpdateProcess StockCountSnapshotProcess SystemStatusMonitorProcess
TransactionIntegrityValidator TransactionProcessingProcess WebMailProcess
WeekEndScheduler WorkFlowScheduler XMLProcessingMessengerProcess
XMLTransformProcess
Press any key to continue . . .
```

For a list of parameters:

```
%JAVA_HOME%\Bin\JAVA
com.retailJava.retailFoundatio
nClasses.systemManager.Process
ManagementClient
```

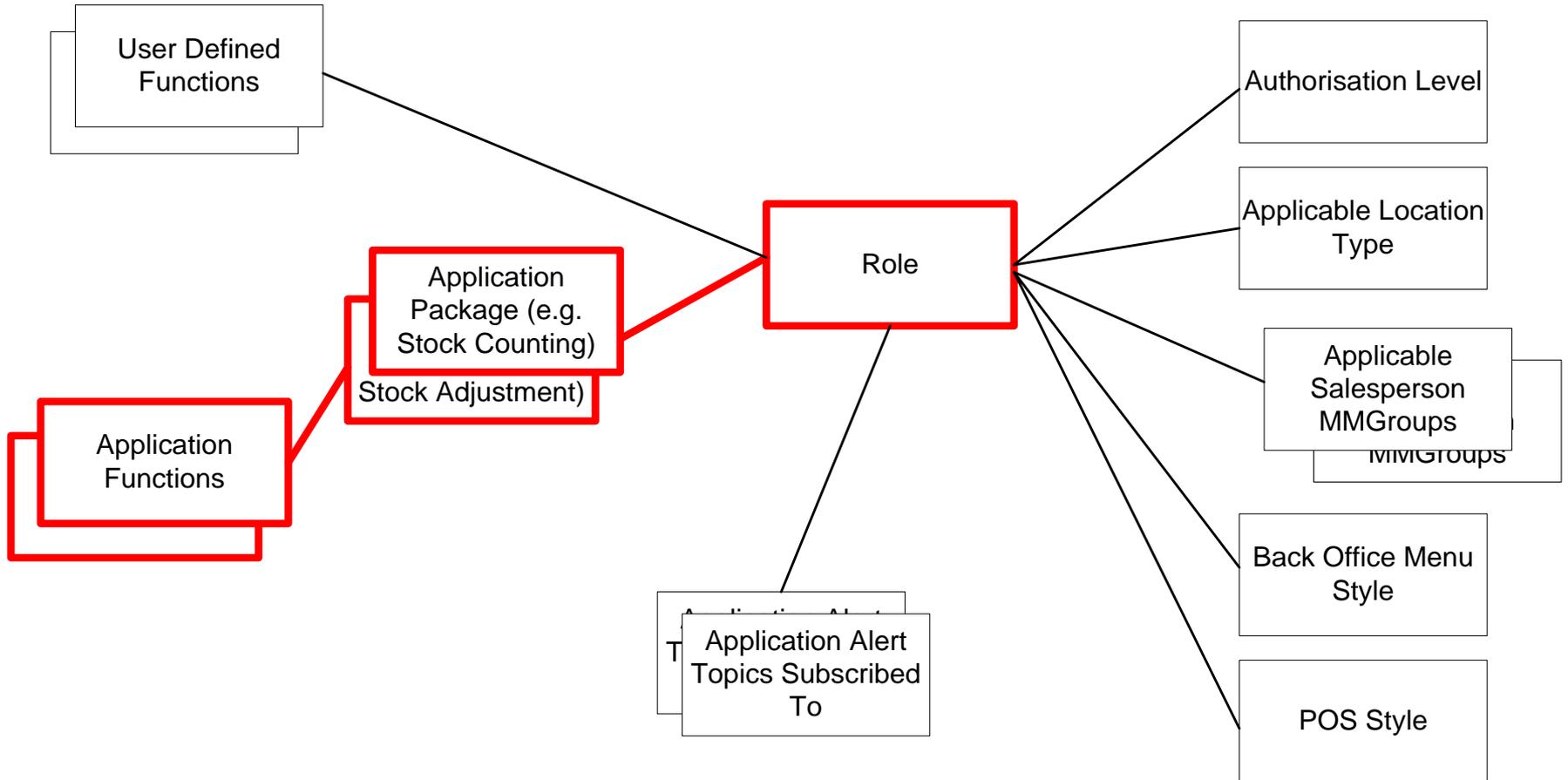
To stop all Retail-J processes:

```
%JAVA_HOME%\Bin\JAVA
com.retailJava.retailFoundatio
nClasses.systemManager.Process
ManagementClient ALL.R1511.S01
127.0.0.1 STOP ALL#
```

To just stop one, replace the ALL with a process name listed above.

# Permissions

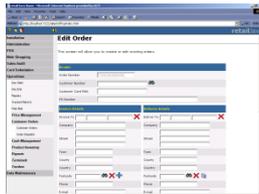
# Role Based Permissions



# POS

- Thick/Client Master Till
- Servlet POS
- POS Configuration
  - Screen Layout
  - Prompts Workflow

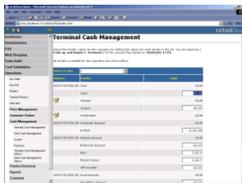
# POS – Thick Client/Master Till



POS



Cash Management



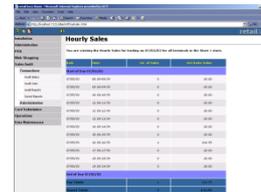
Time & Attendance



Inventory



Reporting



# POS – Servlet

BRANCH: 0001    TERMINAL: 00001    TRANSACTION: 00001    OPERATOR: 00001    24/09/2013    07:43

TAX: £ 0.00  
BALANCE DUE: £ 0.00  
No. of Items: 0

ITEM ONE  
ITEM TWO  
ITEM THREE  
ITEM FOUR  
ITEM FIVE  
ITEM SIX  
ITEM SEVEN  
ITEM EIGHT

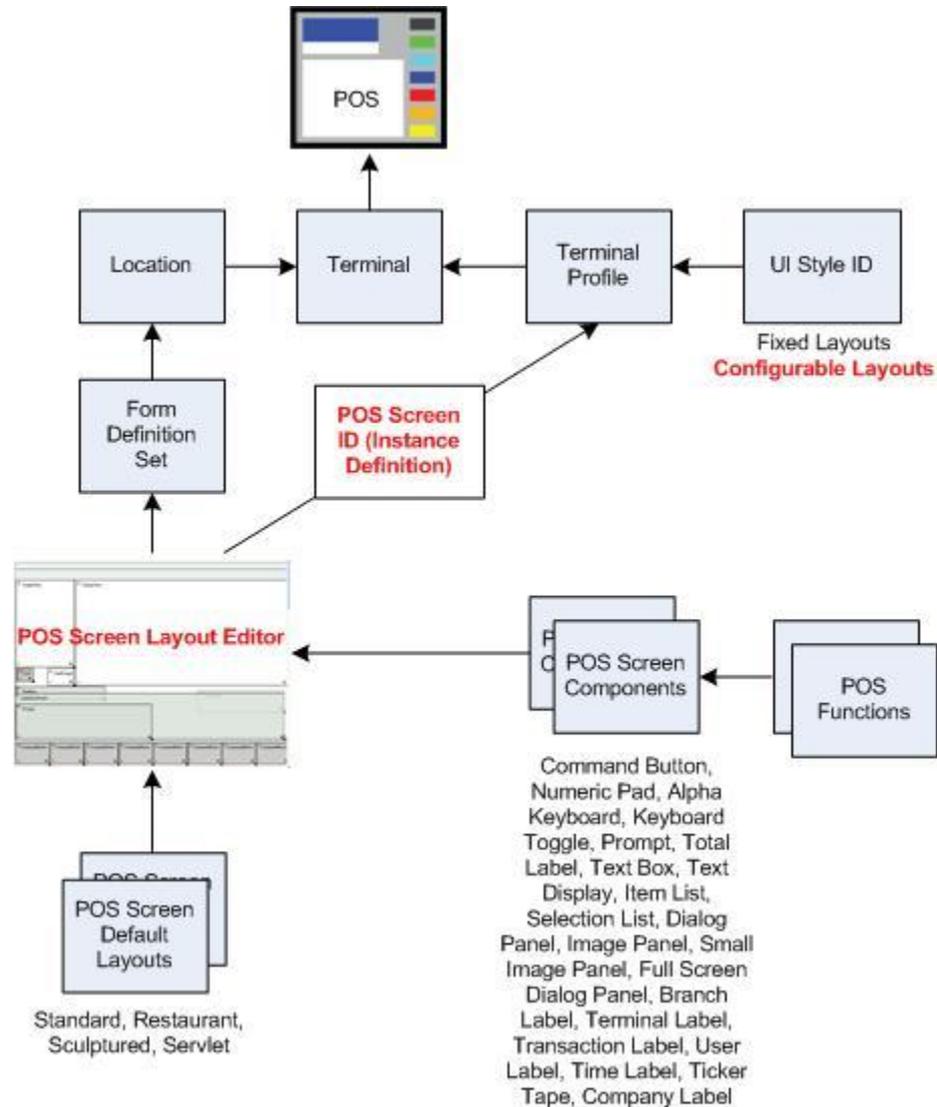
7 8 9 CLR  
4 5 6  
1 2 3 ENTER  
0 .

£ 0.00

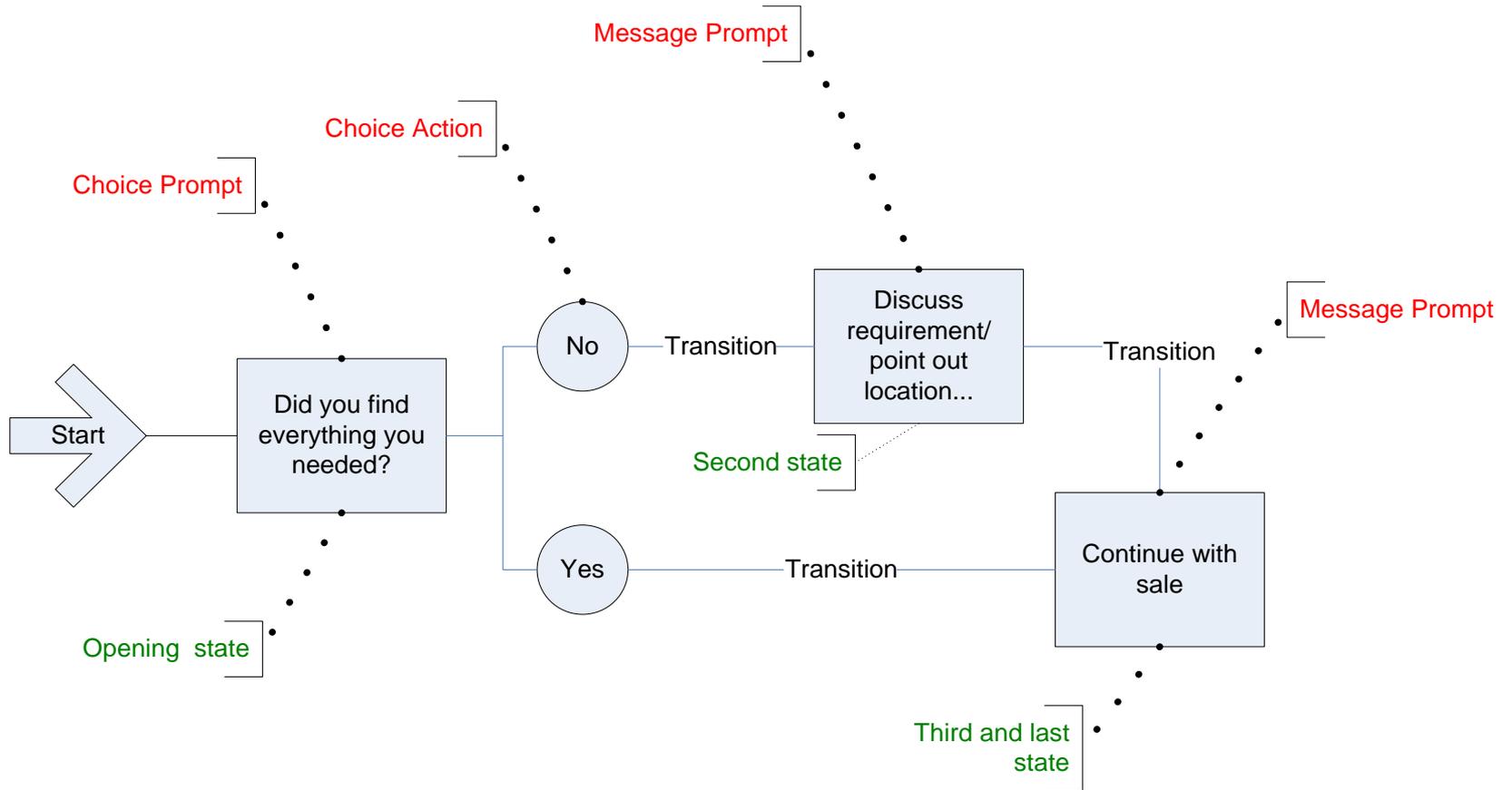
micros Retail-J v12.1

Please enter / scan a PLU

# POS – Layout Designer



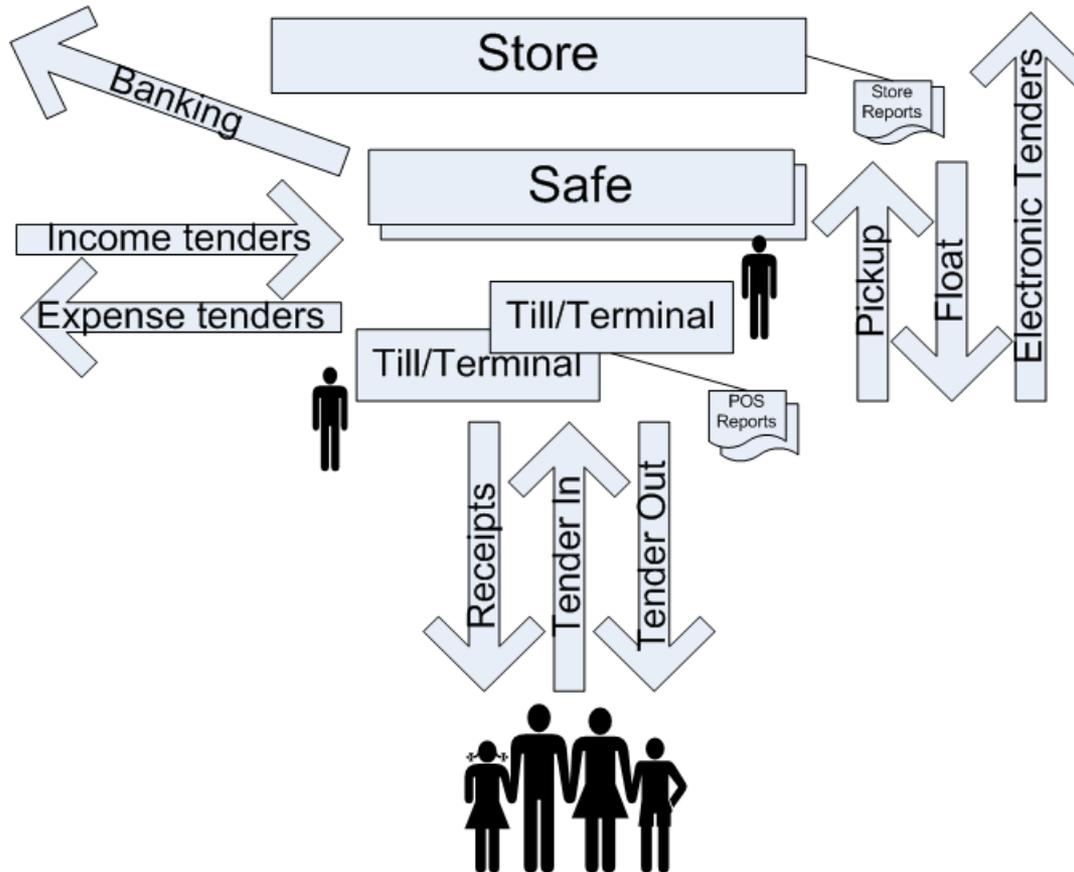
# POS – Configurable Prompt Designer



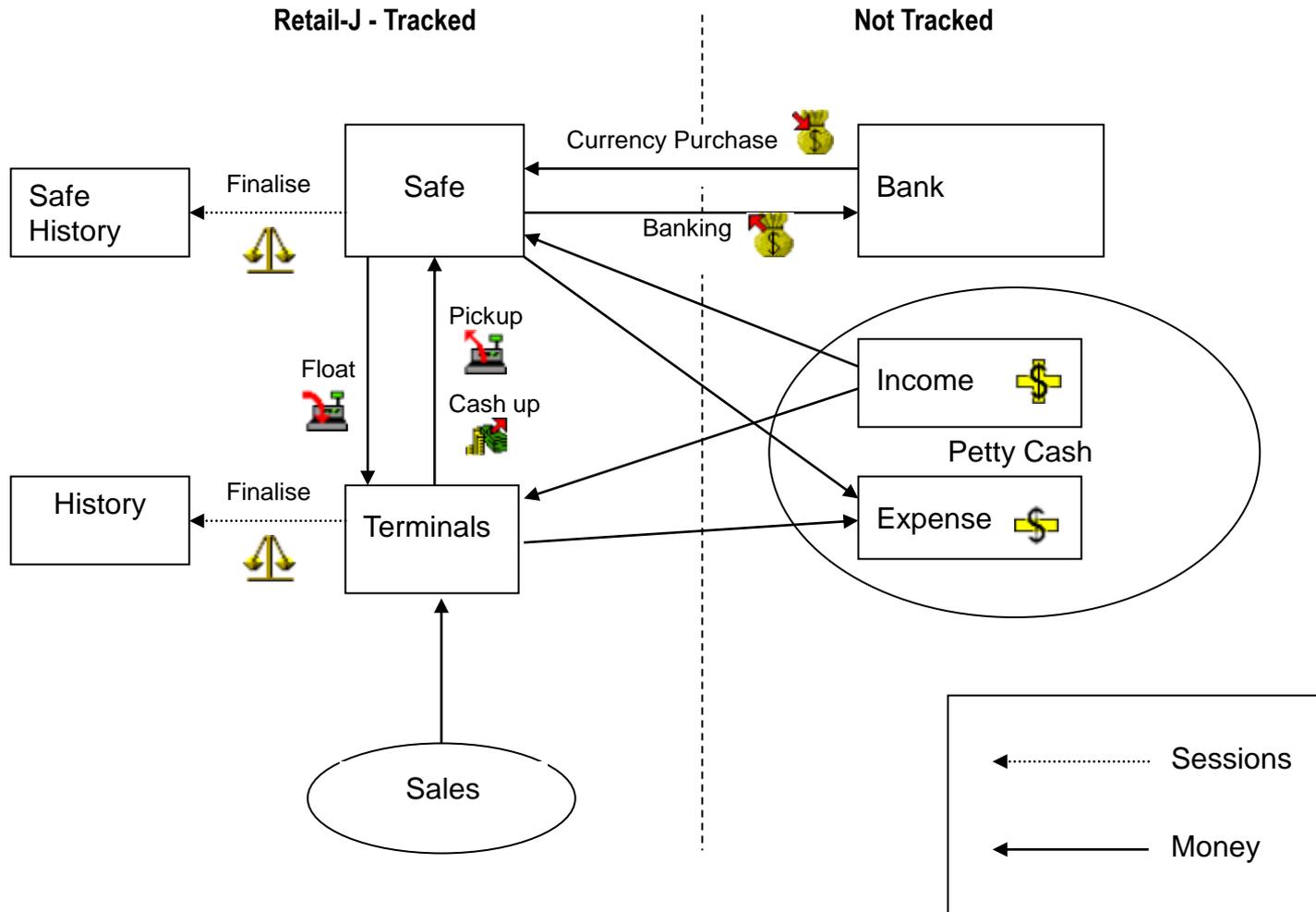
- Transaction flow triggered by sale of a base product
- Progresses through the accessories and services that can be purchased with the base product
- Maps the decisions and subsequent options in a transaction workflow
- The transaction workflow is messaged through the estate to nominated POS profiles

# Cash Management

# Tender Circulation



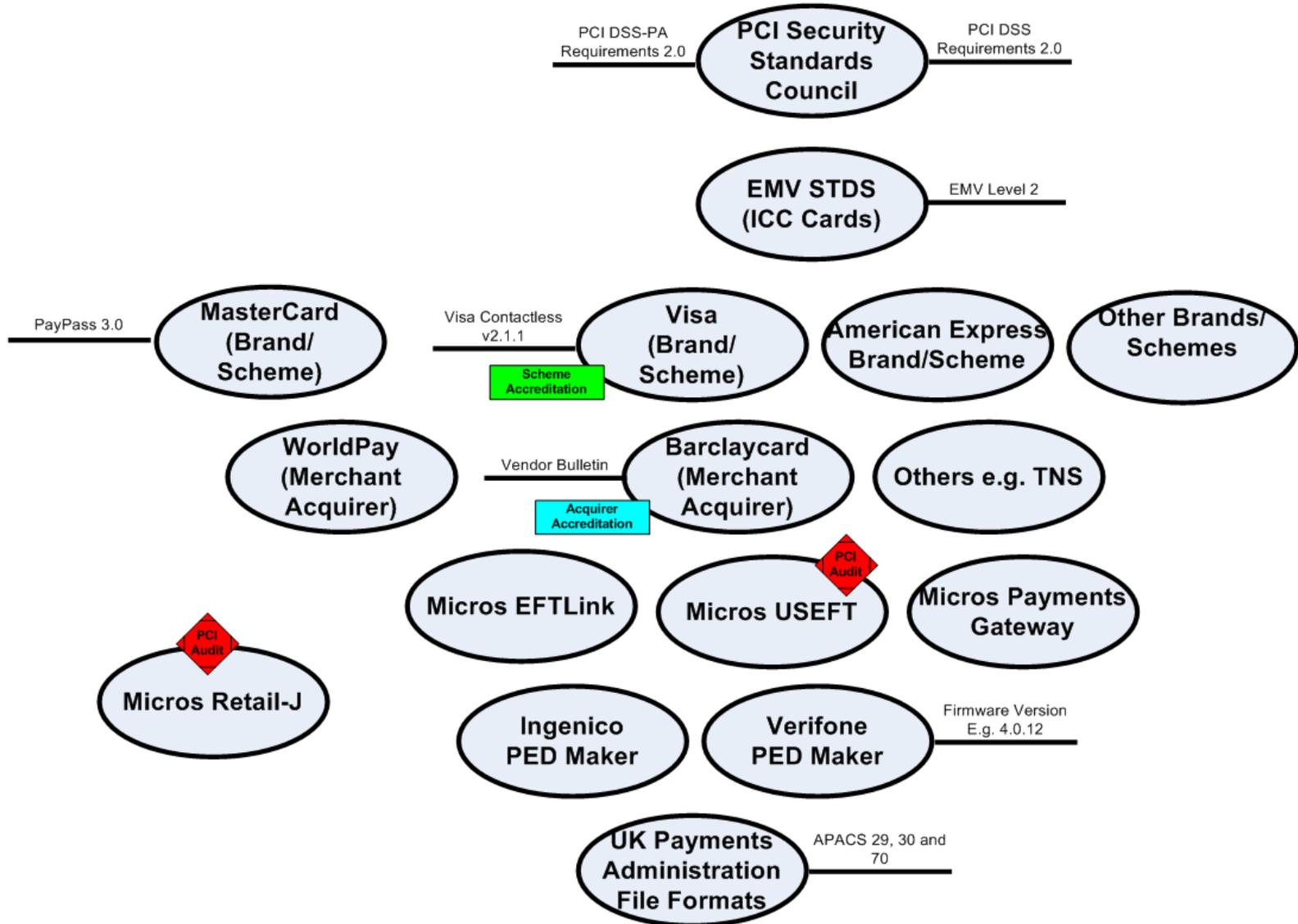
# Cash Management Overview



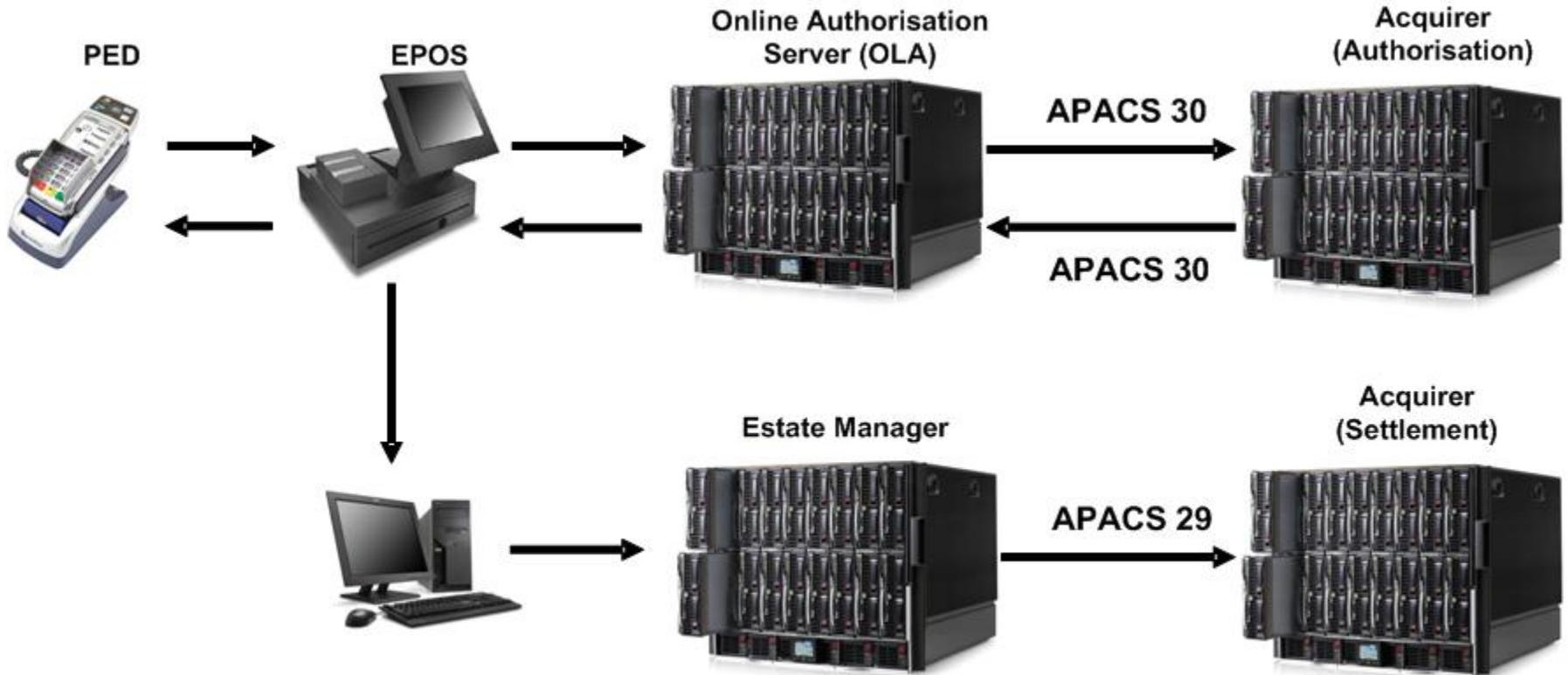
# Card Payments

- Environment
  - Players, Standards, Accreditation and Certification
- Two-part model
  - Authorisation
  - Settlement
- Gift Cards, Rechargeable Vouchers, Loyalty Cards
- Ultra Secure EFT/Managed Service
- PayPal
- Multi-currency and Dynamic Currency Conversion

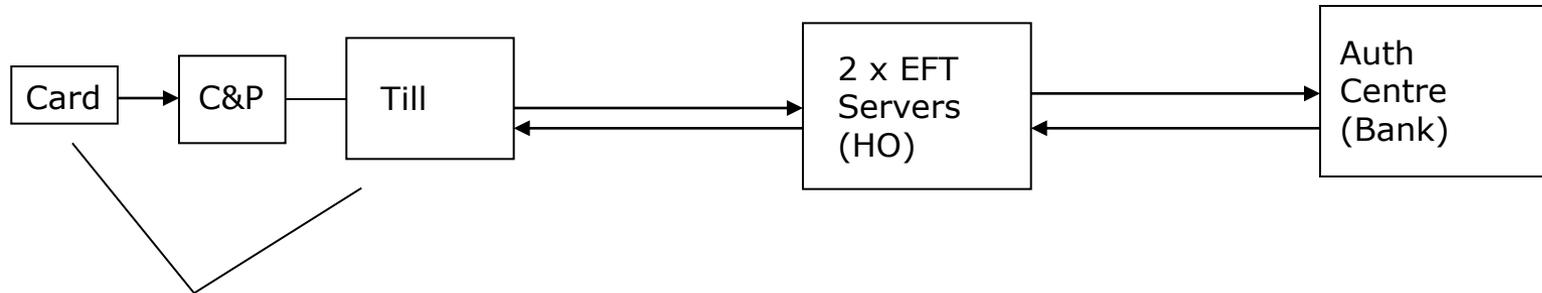
# Card Payments – Environment



# Authorisation and Settlement



# Card Payments Authorisation



Either the chip on the card or the rules configured within Retail-J on the till can trigger a card to go for online Auth

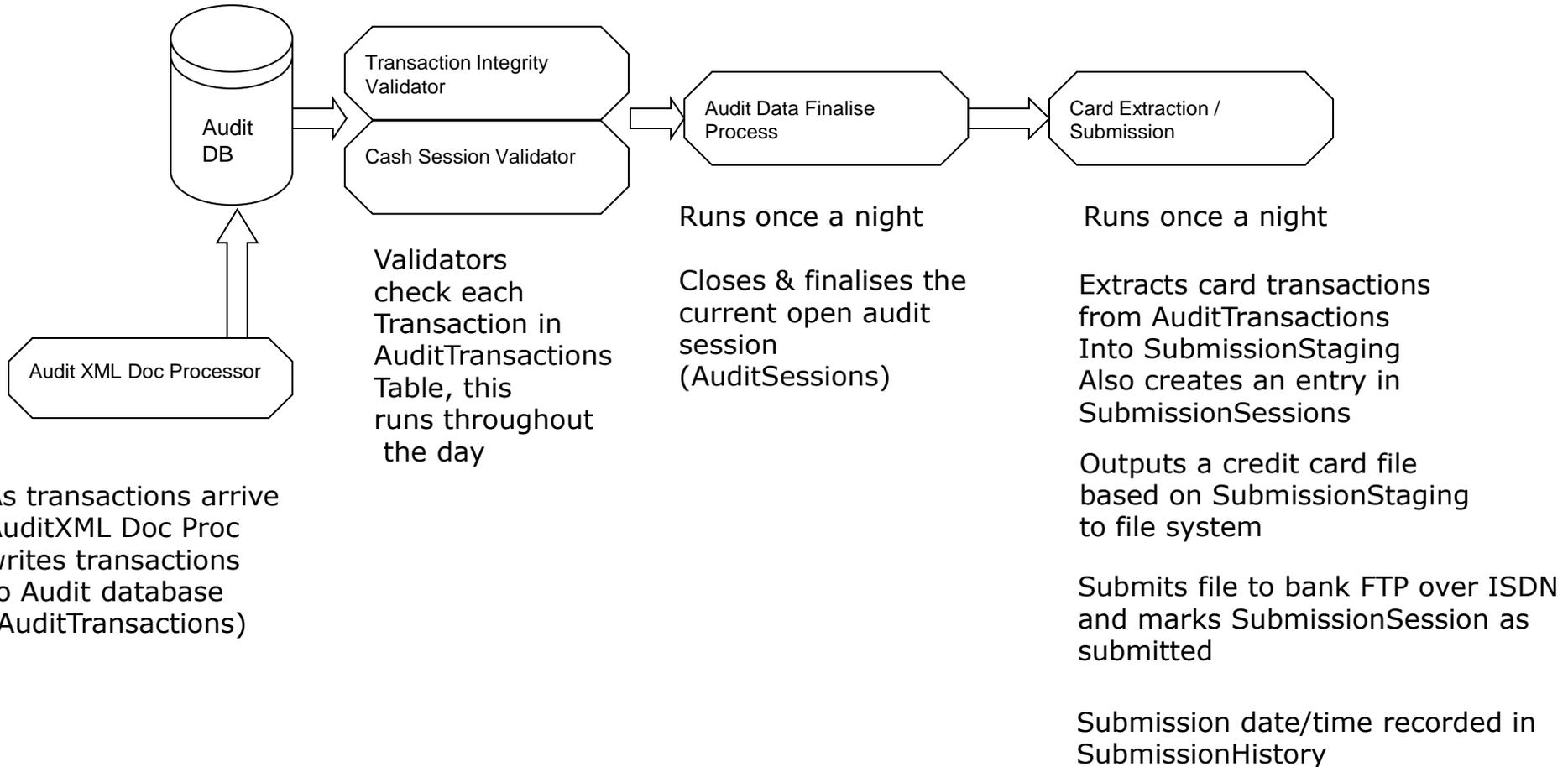
Till passes transaction details to EFT Server and requests an auth

EFT Server connects to bank and requests auth

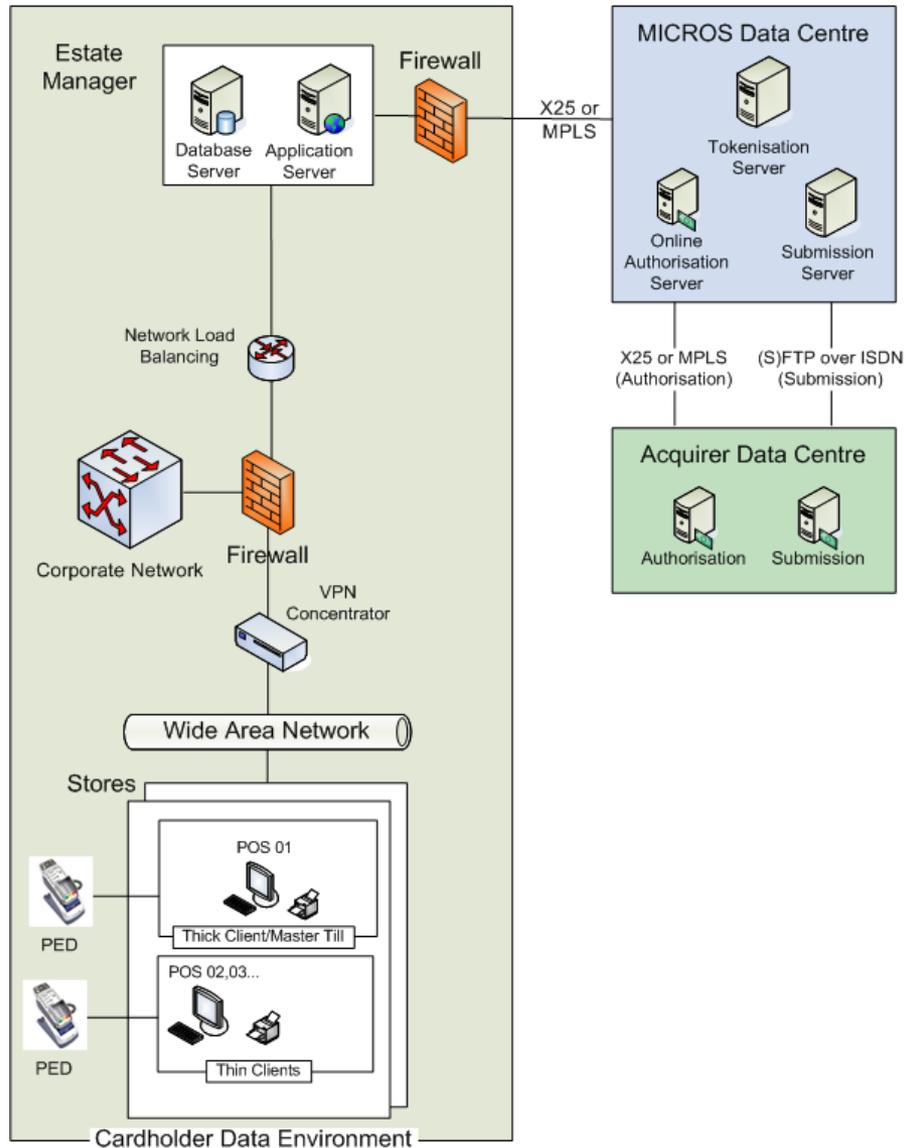
Result of auth (Auth, Decline, Refer, Error) is sent back to the EFT server

EFT server passes result to till

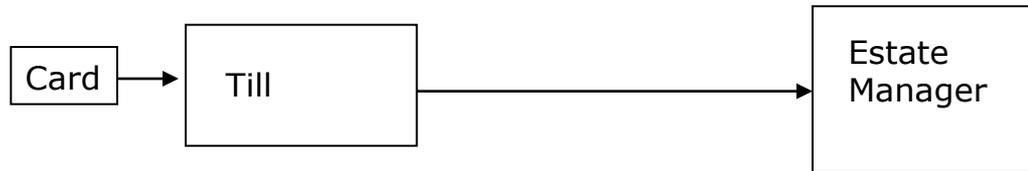
# Card Payments Submission



# Ultra Secure EFT



# Electronic Gift Cards



Card swiped on till



Till makes direct connection to EM using HTTP Servlet and passes card number (+current user & device details)



EM checks card is active and responds with current balance

Till allows tender of card (up to account balance)



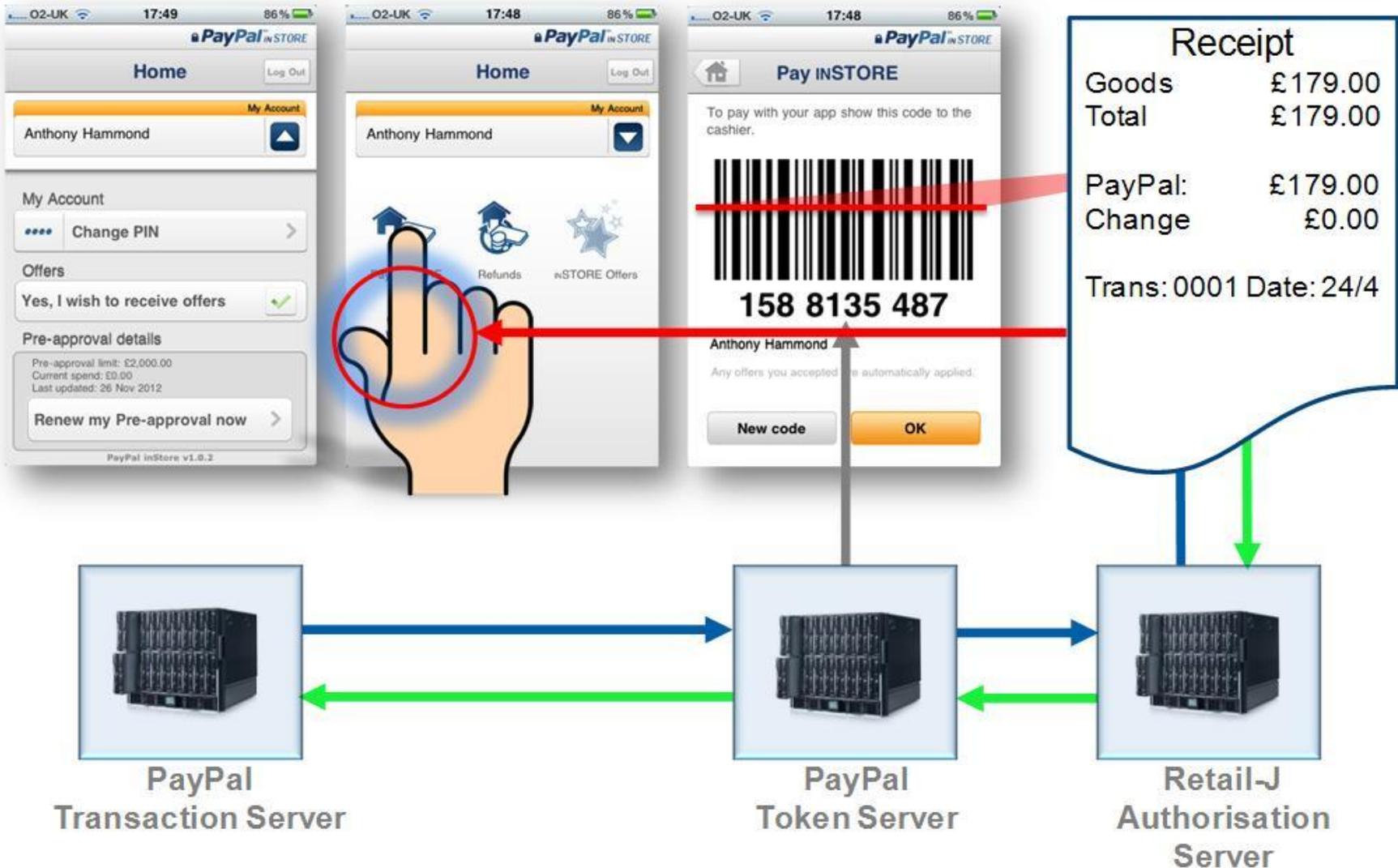
As soon as tender is confirmed, till makes another direct connection to EM to confirm tender



EM deducts tender from card balance And confirms update back to till

When sale reaches EM via normal messaging, the update to the card balance is checked to confirm it has been processed

# Payments - PayPal



# Inventory

# Inventory Architecture

## Browser

### Product Inventory Requests

Product Inventory Requests allows you to request items from another stock holding location within your organisation..

Select a location: Type  Name

Options	Request ID	Source Name	Status	Date Requested	Products Requested	Quantity Requested
	PIR0018	AUT Store	New	21/11/06 10:45	2	7
	PIR0017	AUT Store	New	21/11/06 10:44	2	4
	PIR0016	UK Store	New	21/11/06 10:44	1	10

New Request →

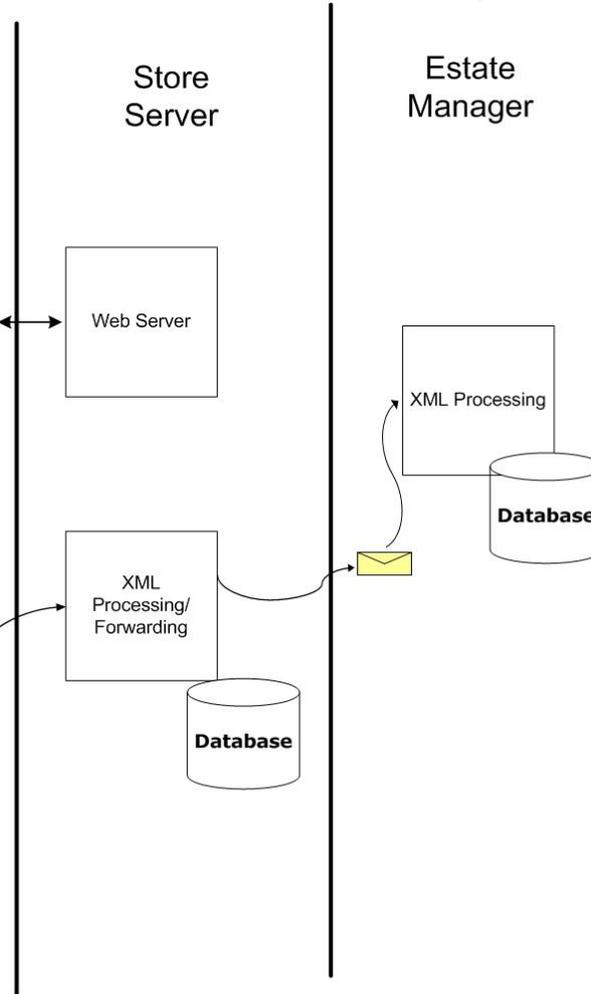
## Terminal

Please enter the cash paid by the customer.

Single-breasted 2pce	£149.00
BALANCE DUE	£149.00

Buttons: OK, £149.00, £150.00, £150.00, Cancel

Branch: 080066 Terminal: 0001 Transaction: 0504 Micie Carrell 04/04/02 14:23



# Product Inventory Functions

Product Search

Stock Enquiry

Product Enquiry

Product Exporter

Expected Deliveries

Goods In

Goods Out

Product Inventory Request

Inventory Reservation Request

Product Inventory Allocations

Purchase Orders

Stock Adjustments

Mobile Counting Device

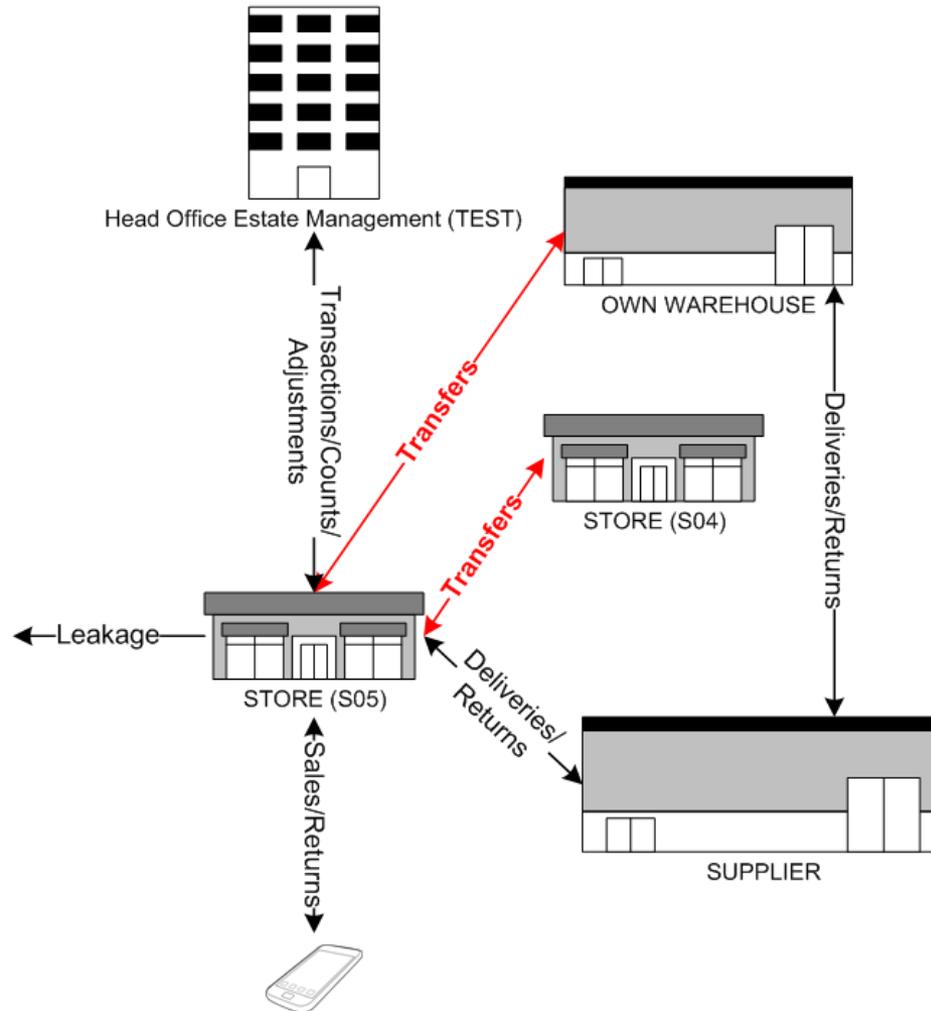
Stock Counting

Replenishment Requests

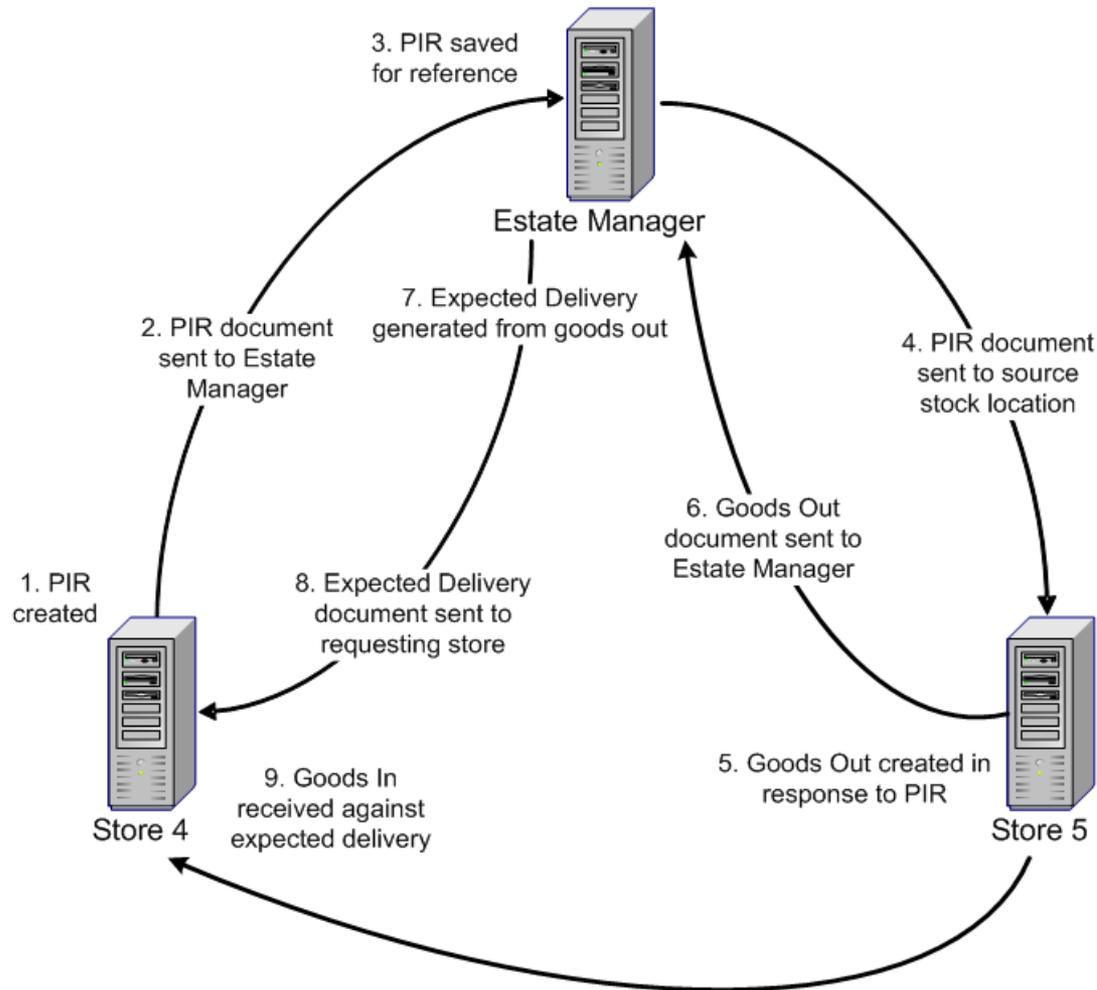
Inventory Reports

# Inventory Flows

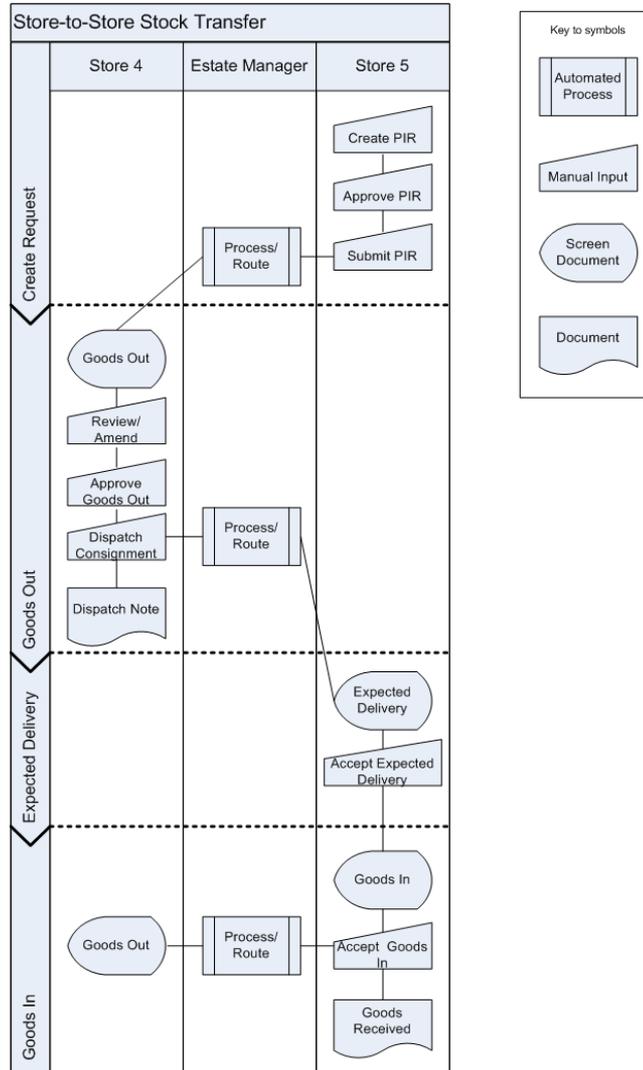
## Inventory Flows



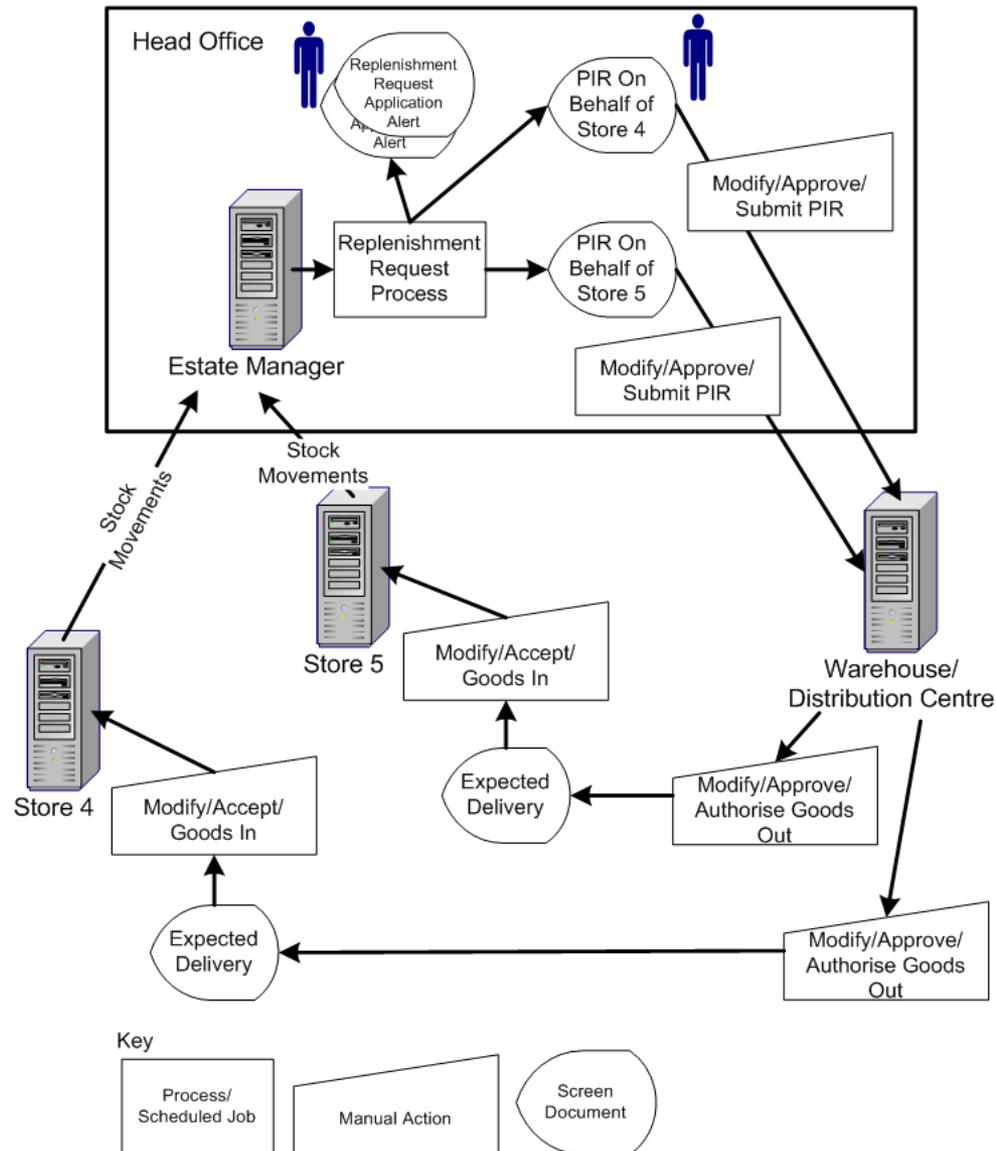
# Product Inventory Request



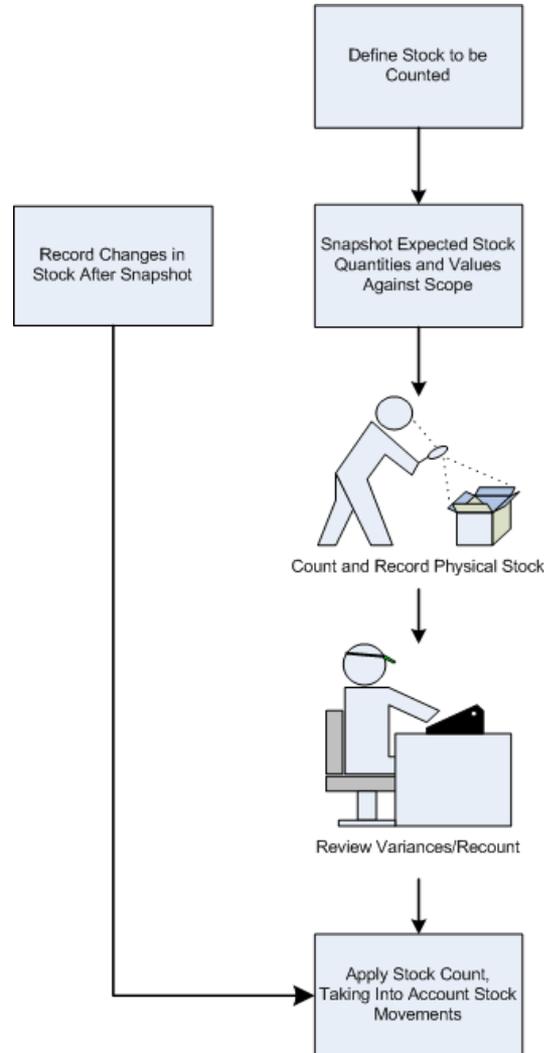
# Inter Branch Transfers



# Central Replenishment



# Stock Count Process



# Inventory Reconciliation

Units	Store 1 Product A	Store 2 Product A	Estate Manager Product A	
			Store 1	Store 2
'Opening Stock'	100	200	100	200
Sales	5	10	5	10
Count	90	195	90	195
Adjustment	-5	+5	-5	+5
'Closing Stock'	90	195	90	195

# **Interfaces**

# Types of Interface

