Document Title:	EPOSS Logical Object Model	
Document Type:	Design	
Abstract:	This document is based on analysis of EPOSS functional requirements and presents the logical object model of the solution for input into the EPOSS physical design process.	
Status:	Draft	
Distribution:	Ian Morrison Richard Sloggett John Englezou Steven Warwick Roy Smethurst Dennis Sandor Library	
Author:	Chris Plunkett	
Approval Authority:	Alan Ward	Technical Manager
Signature/Date:		
Quality Authority:	David Groom	Quality Manager
Signature/Date:		
Comments to:	Chris Plunkett	
Comments by:	16/8/96	

0 DOCUMENT CONTROL

0.1 DOCUMENT HISTORY

Version	Date	Reason

0.2 ASSOCIATED DOCUMENTS

	Reference	Vers	Date	Title	Source
B01	Schedule B01	4	14/5/96	Contract - Schedule B01 - Requirements	
FS5	CR/FSP/003	5	27/6/96	DSS/POCL Functional Specification	

0.3 ABBREVIATIONS

EPOSS	EPOS Service
APS	Automated Products Service

0.4 CHANGES IN THIS VERSION

Add Cash Account Day class + reference it in EPOSS Session.

Add transaction reversal cross reference attribute.

Add EPOSS Token classes.

Modify EPOSS Product Attributes.

0.5 TABLE OF CONTENTS

1 SCOPE

This document describes the logical object model for EPOSS.

It will be extended to cover EPOSS support for APS.

It is under development and is as such incomplete, eg. token classes such as magnetic card, OCR, and EFTPOS have yet to be added and the content of transactions of different transactions types has not yet been addressed..

2 OBJECT MODEL OVERVIEW

The object model is maintained using the Select OMT tool. Class are to be found later in the document. A separate document lists the class catalogue (classes, attributes and operations).

At an overview level, the object model classes are:

Asset
Client
Customer
Item
Organisation
People
Transaction

This section is organised by these classes.

2.1 GENERAL

All classes of object inherit the **All Objects Inherit** class attributes: date/time, PO outlet, employee and register. All updates to objects cause a new object instance to be created thus generating an audit trail.

This inheritance has not been modelled explicitly as it would confuse the model.

2.2 ASSET

The only asset class for EPOSS is **Register** - the equipment where transactions are recorded.

2.3 CLIENT

Client is shown in the model as an external agent, there is no EPOSS data for this class.

2.4 CUSTOMER

Customer is shown in the model as an external agent, there is no EPOSS data for this class.

2.5 ITEM

2.5.1 PRODUCTS

The attributes of **Product** are those generally required to support EPOSS transactions.

Purely for reference currently, **Product** is associated with ARTS **Item**.

There may be a mandatory product relationship ensuring two **Products** are traded together, eg. postal order + fee.

There may be **Pre-Conditions** within a **Session** governing use of a **Product** eg. you can't redeem TV License Stamps unless there is a preceeding TV License Sale in the session.

There are many types of details that require to be captured for different products. Where they cannot be classified as general EPOSS attributes, they are described by **Additional Information**. This mechanism will also be used to support Automated Payment Products.

Products are used to define to settlement products as well as traded products and services. Method of Payment is not identified separately.

2.5.2 PRODUCT TOKENS

Product Tokens link **Products** to generic **Tokens** (magnetic cards or bar codes). The required **Check Digit Method** is referenced by individual **Product Tokens**. The encoding and validation rules of the data on the token are defined by **Product Token Elements** associated with the **Product Token.** These may be linked to **Additional Information** through the Name attribute of both classes. through this link, electronically read data may be used to default product transaction data.

2.6 ORGANISATION

A **PO Outlet** will have a **Fiscal Calendar** associated with it which will include Cash Account Week dates for reporting, however, these can be overridden thus an actual **Cash Account Week** class is required to record instances.

Although strictly part of the Transaction (CO) class, it is useful to reference **Stock Units** here. Each **PO Outlet** may operate several **Stock Units** which are the basic unit of accountability and as such these require to be balanced on demand eg. when handing operation over to another employee. This introduces the concept of the **Balance Period**. There may be several **Balance Periods** for a **Stock Unit** within a **Cash Account Week**.

2.7 PEOPLE

There is considerable functionality around **Employees** relating to auditing system activity.

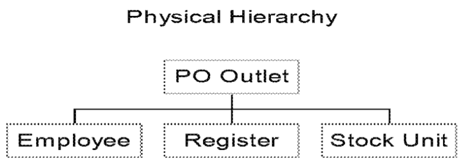
Employees need to log on to the system to use it and to be attached to a **Stock Unit** (they have physical responsibility for stock units). Password and stock unit attachment maintenance is a function of the **Employee** class.

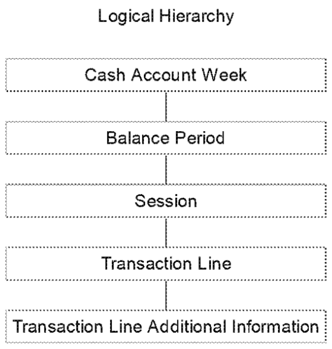
Access to system functions is limited by role and employee role relationships are maintained by this class.

2.8 TRANSACTION

2.8.1 TRANSACTION RECORDING

Trading is organised using the following physical and logical hierarchies:





Business is transacted in a **PO Outlet** by an **Employee** at a **Register** using a **Stock Unit**.

The **PO Outlet** reporting period is by **Cash Account Week**. Each **PO Outlet** may operate several **Stock Units**. **Employees** are responsible for the contents **of Stock Units** so they need to be reconciled (balanced) on demand (eg. at end of shift) - the **Balance Period** defines the periods between committed balances. There may be many **Balance Periods** for a particular **Stock Unit** within a **Cash Account Week**.

During a **Balance Period**, business is transacted in **Sessions**. Only one **Transaction Type** may be transacted within a **Session** eg. Sale, Refund/Reversal, Remittance, Transfer.

Transaction Lines are grouped into **Sessions**. **Transaction Lines** refer to **Products. Transaction Line Additional Information** is added for **Products** which have **Additional Information**.

2.8.2 DATA AGGREGATION FOR ACCOUNTING AND REPORTING

A hierarchy is provided to aggregate raw transaction information for accounting and reporting purposes. This structure includes **Accounting & Reporting Tables, Groups** and **Sub-Groups**. Any of these **Accounting & Reporting Elements** may reference a **Product** and **Transaction Type**.

Report Elements accumulate values from the next level down in the hierarchy, ie. subordinate **Report Elements**, and **Transactions** of the specified **Transaction Type** and **Product**. Accumulation will be bounded by other criteria such **as Cash Account Week**, **Balance Period** or Client Reporting Period for APs.

2.8.2.1 STOCK UNIT BALANCING

This section discusses only the data processing aspects of stock unit accounting and reporting. The preparation of reports to individual formats eg. the Stock Unit Balance Report is excluded.

Stock Unit Balances can be produced at any time during the stock unit balancing period. This requires input data to be prepared such as opening balances and the values and volumes of business transacted for the current balance period plus various aggregates of this raw information. The base data is classified by transaction type and product. The accounting and reporting hierarchy described above is used to drive the aggregation process.

When a stock unit is rolled over to a new balance period, value stock item balances are rolled over creating opening balance transactions. Non-value stock items and suspense accounts do not have stock unit level balances.

For a trail balance (stock unit snapshot), the stock unit is not rolled over to a new balancing period.

An employee can make corrections to stock levels by entering adjustment transactions.

Adjustments to suspense accounts can also be input.

Stock declarations can be made at any time during a stock unit balancing period, however, one must be made as part of the final balancing process. From a declaration, any loss or gain in stock is calculated and compensating adjustment transactions are input.

Cash declarations can be made at any time during a stock unit balancing period, however, one must be made as part of the final balancing process and it must be after the corresponding stock declaration. From a cash declaration, any loss or gain in cash value is calculated and an adjustment transaction is input.

The final balance is produced (as for trial balance) and the stock unit is rolled over to a new balancing period and optionally cash account week.

2.8.2.2 OFFICE BALANCING

Office Balances can be produced at any time during the cash account week. This requires input data to be prepared such as opening balances and the values and volumes of business transacted for the current cash account week plus various aggregates of this raw information. The base data is classified by transaction type and product. The accounting and reporting hierarchy described above is used to drive the aggregation process.

As part of end of cash account week process, once balanced, a office can be rolled over into a new balance period (cash account week). All stock units must have been rolled over before this can occur. Stock units designated as dormant are rolled over automatically. Value stock item and suspense account outlet level balances are rolled over creating opening balance transactions. Non-value stock items do not have outlet level balances.

An office snapshot is an office balance that isn't rolled over.

2.8.3 TRANSACTION TYPES

2.8.3.1 SALES

A sales transaction is started from a menu selection or input from another peripheral, eg. a token swipe. Whatever the route, the required transaction is identified and the **Transaction Line** is created. For token input, transaction details are defaulted using token data.

If the **Register** is in a ready state, a **Session** is automatically started before the **Transaction Line** is created..

The **Employee** may input transaction details either manually or through another peripheral modifying the **Transaction Line** as appropriate. Some details may have been already input from the initial peripheral activity. Some items may have a mandatory links with another which may result in several Transaction Lines being generated. **Transaction Line** details default to **Product** data and those input by the user are validated against the same source.

Further **Transactions Lines** can be created in the same fashion.

[Drafting Note] Do any Transaction Lines need to be committed immediately?

Further transactions can be started within the session.

Transaction Lines can be voided if the **Product** allows.

A **Session** can be voided, in effect voiding each **Transaction Line** within it. The rules governing voidable transactions apply.

Finally a **Session** is completed - it must balance before this can occur. Completion commits all the **Transaction Lines** within it.

2.8.3.2 REFUNDS/RETURNS

Refund/Reversal Sessions are started from the menu. Products can be selected as for Sales (above).

Whilst refunds don't refer to a previous transaction, some reversals do. An operation to retrieve a previous **Transaction Line** is provided to support this.

[Drafting Note] What is the scope of transaction retrieval (stock unit v outlet, current balance period v cash account week, current client reporting period)?

2.8.3.3 REMITTANCES

Remittance Sessions are started from the menu. The direction of the remittance session (In or Out) is selected. Individual Value Stock Item **Products** are selected and the amount and/or value being remitted is entered generating a **Transaction Line** within the **Session.** This process may be repeated for further **Products** adding **Transaction Lines** to the **Session.**

2.8.3.4 TRANSFERS

Transfer Sessions are started from the menu. The type of transfer session (Request, In or Out) is selected. Individual Value Stock Item **Products** are selected and the amount and/or value being transferred is entered generating a **Transaction Line** within the **Session.** This process may be repeated for further **Products** adding **Transaction Lines** to the **Session.**

Transfer Requests do not generate **Transaction Lines** as they do not effect the stock unit contents and only generate printed output.

2.8.3.5 ADJUSTMENTS

Adjustments are used to input corrections and in the final analysis to account for losses and gains. They affect the contents of a Stock Unit and can apply to any Product.

Adjustment Sessions are started from the menu. Individual **Products** are selected and the amount and/or value of the adjustment plus its effect (In or Out) is entered generating a **Transaction Line** within the **Session.** This process may be repeated for further **Products** adding **Transaction Lines** to the **Session.**

3 CLASS DIAGRAMS

Relevant Class Diagrams are appended.