FUJ00079219

Document Title: Eposs Attribute Grammar Catalogue

Document Type: Technical Design

Abstract: This document describes the Persistent Objects and Messages used in the Pathway EPOSS Implementation, expressed as attribute grammar

Distribution:

Document Status: Approved

Document Predecessor: Version 5.5

Associated Documents: EPOSS Token Handling
EPOSS Integration Specification
EPOSS Technical Design Overview

Author: Alan Ward (this version)

Approval Authority:

Signature/Date:

Comments To:

Comments By:

FUJ00079219

0 Document Control

0.1 Document History

Version	Date	Reason
1.0	10/08/96	First Issue
2.0	02/09/96	Second Issue - new attributes and incorporation of constants to identify attribute names
3.0	15/01/97	Major revisions to bring up to date with production attribute grammar. Addition of various new record types.
4.0	14/05/97	Final issue by original author, as part of handover activities. Brings up to date with final implementation.
5.0	21/7/97	Complete revision, changing the format to reflect normally accepted formats for the attribute grammar and to reflect the actual data used and generated.
5.4	12/9/97	Substantially modified to reflect implementation of EPOSS produced by Escher.
5.5	17/9	Declaration object name description revised. CutOffs object reverts back to its original use – just for reports.
5.6 & 5.7	30/10	A large number of changes have been introduced in this version. I suggest comparison with a previous version to find out the differences.
6.0	12/12/97	This document has been administratively baselined in order to bring the document under formal change control

0.2 Future Changes

There are a significant number of objects used by EPOSS that are not currently in this document. They will be added piecemeal.

Declarations is being totally re-worked.

0.3 Associated Documents

Version	Date	Title	Source

FUJ00079219

		EPOSS/TD/004 - EPOSS Token Handling	
		EPOSS/TD/003 - EPOSS Integration Specification	
		EPOSS/DES/001 - EPOSS Technical Design Overview	

0.4 Abbreviations

APDM	ARTS Point-of-Sale Data Model document
ARTS	Association for Retail Technology Standards
EPOSS	Electronic Point of Sale Service
POCL	Post Office Counters Ltd
CAP	Cash Account Period
BP	Balancing Period
SU	Stock Unit

0.5

FUJ00079219

CONTENTS

FUJ00079219

FUJ00079219

1 Overview

The Pathway EPOSS Product is driven largely by a set of Persistent Objects that form the EPOSS Reference Data. All the output EPOSS generates, whether it is for external 'consumption' or for internal working purposes is also in attribute grammar format – some of these are persistent objects and some are conventional messages. This document describes both these in attribute grammar format.

FUJ00079219

2 Reference Data

The following sections describe those attribute grammar records that drive the EPOSS applications behaviour.

2.1 EPOSS product

2.1.1 Description

The EPOSS Product object describes the transaction centred attributes of a POCL product or service. Example attributes are Product Name and Retail Price.

The EPOSS Product is made up of 'Core' and 'Extended' attributes of which some attributes or attribute groups are defined as optional. The core attributes apply to all products and the extended attributes apply only to certain product types.

EPOSS has only one set of extended attributes. It is called 'Additional Data' and is used to capture additional data that is specific to a particular class of transaction. Different classes of transaction will expect different sets of additional data. An example is a BT Bill payment.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:EPOSSProducts>
   <ObjectName: >                Unique number identifying Product
   <Suffix: >
   <Depend:bool>                 Identifies whether this product uses the dependency mechanism
                                 to determine whether it is available in a specific outlet.
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_EPOSSProducts>
   <ObjectName: >                Product number plus preceding suffix
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <PN: >                  Product number (again)
         <SN:text(10)>           Short product name
         <LN:text(24)>           Long product name
         <RN:text(16)>           Receipt product name
```

FUJ00079219

<FP:**bool**>	Determines if this product has a fixed price which cannot be overridden
<RP: >	Retail price - This attribute defines the sale price for a single item. For open priced products, the User can override this value.
<MV: >	Multiple value - This value defines the units at which this product can be transacted. I.e. the transaction value must be exactly divisible by this value.
<SE:In|Out>	This attribute indicates the effect this product has on a session. Session Effect is defined below.
<AS:**bool**>	Adopt settlement sense - Indicates whether the value of this product when transacted adopts the sense (Positive / Negative) necessary to balance the current session
<MnV: >	Minimum transaction value allowed
<MxV: >	Maximum transaction value allowed
<MnQ: >	Minimum allowable quantity
<MxQ: >	Maximum allowable quantity
<SR:**bool**>	Indicates if the printing of an Session Receipt is required if this product is sold.
<MP: >	Mandatory Product – contains the product number of a product that must be transacted along with this Product.
<V:0>	unknown
<RV:**bool**>	Indicates whether transactions for this product are reversible
<RT:>	Receipt type - Identifies a Receipt Definition for this Product (Id of report definition)
<RA:>	Reversal authority - Identifies the application that is responsible for authorising a transaction reversal.
<SI:>	Service instructions - Text to display to the clerk if the system prompts for information.
<IA:>	Identifies any integrated application that works with EPOSS to transact this product.
<PM: <L1: > <L2: > <L3: > <L4: > <L5: > >	Maps this product's transactions to the summarisation hierarchy.
<SM: <L1: > <L2: > <L3: > <L4: > <L5: > >	Optionally maps this product's transactions to another leaf on the summarisation hierarchy
<I:True>	***unknown***
<ST:**text(1)**>	Identifies the default Service Type code for any transactions for this product.
<DEP:E>	the Pathway release this product is in
<PreCondition:	Zero or more pre-conditions for this product.
<ProductNo: >	Product No of this product.
<PCProdNo: >	Pre condition product. I.e., preceding product that is looked for
<IsMand:**bool**>	Indicates if rule is mandatory or simply a warning. (Not yet implemented)
<Msg: >	Message to display to user if pre-condition is not met.
<IE:False>	Indicates if rule is exclusive (i.e. this product must NOT exist) or inclusive

FUJ00079219


```
>
<AdditionalData:        Zero or more sections describing any Additional Data that
                        requires capture
   <P: >                        Prompt describing item being captured (displayed next
                                to field on screen).
   <F: >                        Format - Visual Basic Wildcard comparison string used
                                to validate captured data.
   <S: Numeric|Alphanumeric|Date|Currency >
                                Script type - Describes the type of data being captured.
   <D:>                         Default value that User can accept or override.
   <Max: >                      Maximum allowable length of data input
   <Min: >                      Minimum allowable length of data input
   <VM: >                       Validation message to display to user if invalid data is
                                entered (validation of the Format attribute fails).
   <N: >                        Name for the additional data item being captured.
                                (This is used in the transaction record).
   <C: >                        Caption to display when data item is being entered.
   <O: >                        Number determining order in which additional data
                                items are captured (> 50 = prompted for after Core
                                EPOSS data, <50 = prompted for before Core EPOSS)
   <A: None|Display|AllowEdit>
                                Action allowed if this data item is filled automatically
                                by a data capture device
   <SD:                         Script Data - Additional attribute grammar describing
                                the interface characteristics (e.g. Options for
                                OptionList)
      <Option:                  One entry for each option on an option list
         <Text: >               Text to display for option
         <Key: >                Key to select that option
      >
   >
>
>
>
```

2.1.2 Session Effect Attribute

In classifying a products behaviour at point of sale it is necessary to describe the products effect on the session in absolute terms, thereby allowing the system to calculate the session balance in differing functions such as Serve Customer and Refund / Reversal.

The SessionEffect attribute defines “In” as meaning an increase in the balance due to the Post Office. An example of an “In” product is a Stamp sale or a DNS Deposit. An example of an “Out” product would be a Benefit Payment.

The SessionEffect attribute does not drive the accounting and reporting mechanisms; an “In” product may not always appear on the receipts side of the printed balance report for example, Rent Summaries that appear on the same side of the balance as Rent Vouchers, for consistency.

The session effect of a product can be reversed whilst committing transactions in a mode such as Reversal.

FUJ00079219

2.1.3 Pre-conditions

Pre-conditions describe one or more products that must have been transacted before the product describing the pre-condition. An example of this would be TV Stamps redeemed having a pre-condition of TV licence.

Products that describe a pre-condition are not explicitly linked to a given instance of the pre-condition product in the session. For example it would be possible to sell 2 TV licences and have only one TV Stamps redeemed product in a given session.

It should be noted that the voiding of all pre-condition products is not permitted if a transaction exists that references that product as a pre-condition. For example, it should not be possible to void both TV licence transactions without voiding the TV Stamps redeemed.

2.2 Revaluation

Inventoried products (or value stock) can have its value changed whilst it is being held within a stock unit. If this happens additional transactions are written to the stock unit account to reflect the change in the value of the stock unit.

There are a special set of products that are used for this and there is a relationship between the product that is being re-valued and the product that represents the revaluation.

This relationship is held within the Revaluation collection.

The attribute grammar is defined below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:Revaluation>
   <ObjectName: >              Product number that this object defines the revaluation products
                               for. Can also contain 'Default'. The default object is used for
                               any product that does not have its own specific revaluation
                               object.
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_Revaluation>
   <ObjectName: >              Object name + "_" + suffix from previous object
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
```

FUJ00079219


```
            <PUp: >                 Product number for the product that is used for an increase in
                                    price.
            <PDown: >               Product number for the product that is used for a decrease in
                                    price
        >
    >
>
```

2.3 Interface item

Interface items collectively describe the EPOSS Menu hierarchy. Interface item persistent objects are used by the EPOSS application at run-time to build the Desktop Menu Hierarchy. The attribute grammar description follows:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:InterfaceItems>
   <ObjectName: >               It is not clear why the names start with 'IItem'. This id followed
                                by the items identifier
   <Suffix: >
   <Depend:bool>                Identifies whether this interface item uses the dependency
                                mechanism to determine whether it is available in a specific
                                outlet.
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_InterfaceItems>
   <ObjectName: >               Object name + "_" + suffix from previous object
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <SupportedModes:       A list of modes for which this button is supported. By default a
                                mode is supported, hence a mode is only normally shown if it is
                                not supported (value 0). If this entire attribute is missing then all
                                modes are supported.
            <ER:0|1>                    Linked Reversal
            <SC:0|1>                    Serve Customer
            <TI:0|1>                    Transfer In
            <TO:0|1>                    Transfer Out
            <RISD:0|1>                  Remit In – Supplies Division
            <RIOP:0|1>                  Remit In – Other Post Office
            <RICL:0|1>                  Remit In – Client
            <ROSD:0|1>                  Remit Out – Supplies Division
            <ROOP:0|1>                  Remit Out – Other Post Office
            <RODC:0|1>                  Remit Out – Data Centre
            <ROCL:0|1>                  Remit Out – Client
```

FUJ00079219


```
   <RV:0|1>                    Unlinked Reversal
   <REC:0|1>                   Recovery
   <HK:0|1>                    Housekeeping
   <NAD:0|1>
   <PT:0|1>
<IntDef:                       Command issued when interface item is pressed. Below there
                               are a set of real examples:
   <EPOSSImpulse:
      <Cmd:ChangeMode>
      <Mode:SC>
      <ModeTitle:Serve Customer>
      <BlackBoxData:
         <S:1>
      >
      <SessionReceipt:5>
      <AlwaysPrintReceipt:False>
      <ReverseSense:False>
      <DASS:False>
      <PrimaryMappings:>
      <SecondaryMappings:>
      <NavigateString:
         <Menu:IItem194>
         <Menu:IItem1>
      >
   >
>
or <IntDef:                    another example
   <EPOSSImpulse:
      <Cmd:NonEPOSSMode>
      <AppName:BES>
      <ModeName:BC>
      <CmdStr:
         <Cmd:ReaderFailureEntry>
      >
      <StackPic:162>
      <StackPicFile:newicons>
      <StackCaption:Reader Failure>
   >
>
or <IntDef:                    another example
   <EPOSSAppMain:
      <Cmd:>
      <Mode:>
      <ModeTitle:>
      <BlackBoxData:
         <S:>
         <M:>
      <SessionReceipt:>
      <AlwaysPrintReceipt:>
      <ReverseSense:>
      <DASS:>
      <PrimaryMappings:>
      <SecondaryMappings:>
      <NavigateString.Menu:>
      <StackPic:>
      <StackPicFile:>
      <StackCaption:>
      <ShowNoRed:>
      <SettlementProduct:>
      <GetStockUnit:>
      <ProductNo: >
   >
>
or <IntDef:
```

FUJ00079219


```
   <EPOSSStockUnit:
      <Cmd:>
      <StackPic:>
      <StackPicFile:>
      <StackCaption:>
   >
>
or <IntDef:
   <EPOSSReport:
      <Cmd:>
      <PrintDestination:>
      <PreviewDestination:>
      <UseRollOverCutOff:>
      <HomeMenu:>
      <ShowActionButton:>
      <ReportTitle:>
      <ReportID:>
      <Criteria:>
      <SpecificCriteria:
         <Op1:>
         <Comp:>
         <Op2:>
      >
      <StackPic:>
      <StackPicFile:>
      <StackCaption:>
   >
>
or <IntDef:                    another example
   <OBCS:
      <Cmd: >
      <StackPic: >
      <StackPicFile: >
      <StackCaption: >
   >
>
or <IntDef:                    another example
   <APS:
      <Cmd: >
      <StackPic: >
      <StackPicFile: >
      <StackCaption: >
   >
   >
or <IntDef:                    another example
   <Default:
      <StackPic: >
      <StackPicFile: >
      <StackCaption: >
   >
>
or <IntDef:                    another example
   <Unknown:
      <StackPic: >
      <StackPicFile: >
      <StackCaption: >
   >
>
<Hierarchy:>                   A string that defines this interface items place in the menu
                               hierarchy.  Consists of a number of interface item names
                               separated by '|'. (e.g., IItem194|IItem1)
<Depth: >                      Indicates at what depth the Sub Menu (if any) of this interface
                               items should appear.  If null, item assumes default depth.
```

FUJ00079219

<Level: >	For items that have child items in the menu hierarchy (i.e. submenu buttons) this number defines the level of menu titles that will appear on the submenu. Level 0 has no title, level 1 a single title and level 2 and above have two titles.

```
<Sec:
   <Security:
      <Collection:MenuSecurity>
      <ObjectName:EndofDay>
   >
>
```

<BB:**bool**>	Determines if a back button is shown. Only used if this interface item houses a sub-menu
<Cap: >	Caption that appears on interface item
<Pos: >	Position on parent menu that item appears
<Pic: >	Number of icon within picture file
<IntType: >	Style of interface item (defined in the Riposte Desktop API document)
<PicFile: >	File used to retrieve items icon
<Desc: >	Description of interface item – used as title of any submenu below this interface item. Only used if this interface item houses a sub-menu
<Hlp: >	Help Text to display if the user selects help for this interface item

```
      >
   >
>
```

2.4 Token Impulse

Token impulse definitions identify and route data from data capture devices through to interested applications, transforming the data into the elements required by the target application.

The token impulse contains core attributes and a repeating attribute defining each element of the token data. Elements have a defined 'Picture' that is used to identify a token when read. For example, an APS Magnetic Card token may have an element called Issuer Identity with a picture of "63391122".

Tokens are only valid in assigned 'modes', ensuring that token driven transactions only take place in an appropriate context. For example, Automated Payment transactions are not valid in reversal mode.

The attribute grammar description follows:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:EPOSSTokens>
   <ObjectName: >
   <Suffix: >
   <Depend:bool>
>
<Message:
```

<ObjectName: >	unique number identifying the Token (e.g., TOKEN1002)
<Depend:**bool**>	Identifies whether this impulse uses the dependency mechanism to determine whether it is available in a specific outlet.

FUJ00079219

<GroupId: >	
<Id: >	
<Num: >	
<Date:dd-mmm-yyyy>	
<Time:hh:mm:ss>	
<User: >	
<Expiry: >	
<Collection:_EPOSSTokens>	
<ObjectName: >	Object name + "_" + suffix from preceding object.
<StartDate:dd-mmm-yyyy hh:mm:ss>	
<EndDate:>	
<RData:	
<Data:	
<Token:	
<TID: >	Token id. Again!
<Len:0>	Required length of token (if defined as zero, minimum and maximum length attributes are used
<MinLen: >	Minimum required length of token data
<MaxLen: >	Maximum required length of token data
<CDig:>	Definition of the Riposte check digit routine to use (not currently used)
<Inc:>	Definition of any composite fields to build. Format: "<[IncludeName]:<[Element]:><[Element]:>>"
<IntDef:	Interface to call when token is read. Similar (but simpler) to the same attribute in Interface Items.
>	
<PFData:	Attribute grammar to pass to interface defined in InterfaceDef within the <PreFilled:> attribute. Pre-filled attributes are stored in any additional data attributes having a matching name. (Example: <CID:1001><CA:2001><SVC:0><SG:AA><TT:MC><CN:South Wales Electricity><TI:1002><TV:1>
>	
<DecData:	Data describing the non-Riposte decoding method (currently for APS use only). Example: <MinLen:9><MaxLen:9><Flg:2><Div:10><"W:1,3,7,1,3,7,1,3,7,0,0,0,0,0,0,0,0,0,0">><CDs:0>
>	
<SData:	Attribute grammar that defines characteristics of a stack button (only applies to tokens that result in an EPOSS transaction)
<StackPicFile: >	
<StackPic: >	
<StackCaption: >	
>	
<ValApp:>	Not used (included for future proofing)
<ValCmd:>	ditto
<ValName:>	ditto
<PropApp: >	identifies an integrated application that will pre-process this token data
<PT:	Data that denotes that this token should be passed through to an application when read. The attributes are included for BES integration and allow EPOSS to pass token impulses to BES, whilst allowing BES to recognise the token names as those it deems as valid.
<AppName: >	the application name of the application to pass this token through to

FUJ00079219


```
                <TokenName: >          The token name that should be used when passing this
                                       token impulse through to the above application (e.g.,:
                                       BESCardTrack2Type1)
            >
            <BB:                       'Black Box' Data that is included in the token impulse
                                       for passed through token impulses. The following is an
                                       example for BES.
                <PAN:
                    <Include:
                        <IIN:>
                        <NumNINO:>
                    >
                >
                <CardID:
                    <Include:
                        <IIN:>
                        <NumNINO:>
                        <IssueNo:>
                    >
                >
            >
            <DEP:E>                    unknown
        >
        <Element:                      Zero or more definitions of elements within the overall token
                                       data. These element definitions provide the mechanism by
                                       which token data is 'sliced up'
            <Name: >                       Identifier for this element. This name is passed to the
                                           application defined in InterfaceDef within the PreFilled
                                           attribute. For data items that will populate EPOSS
                                           Product Additional Data items, this name must
                                           correspond with the Additional Data name.
            <Start: >                      Position within the token data that this element begins
            <Len: >                        Length of this element
            <Pic: >                        String used to validate this element of the token data
                                           (e.g.: "=", "633200", N(3))
            <CD:                           A Riposte check digit string, describing the check digit
                                           validation that should take place on this element
                <Start: >
                <Len: >
                <Type: >
                <Length: >
            >
        >
        <Mode: >                       zero or more Mode attributes indicating a valid mode for this
                                       Token. (E.g.: BC, PUN, SC)
    >
  >
>
```

2.5 PLU Impulse

PLU Impulses allow a user to initiate a transaction by typing in a PLU number, rather than navigating through a complex menu hierarchy. One or more PLU Impulses can relate to a single product, allowing the behaviour of the product to be modified for each PLU, whilst still transacting against a single product number.

The attribute grammar is defined below:

```
<Message:
   <GroupId: >
```

FUJ00079219


```
      <Id: >
      <Num: >
      <Date:dd-mmm-yyyy>
      <Time:hh:mm:ss>
      <User: >
      <Expiry: >
      <Collection:PLUImpulses>
      <ObjectName: >                Unique number identifying the PLU, prefixed by 'PLU'.
      <Suffix: >
      <Depend:bool>                 Identifies whether this PLU impulse uses the dependency
                                    mechanism to determine whether it is available in a specific
                                    outlet.
>
<Message:
      <GroupId: >
      <Id: >
      <Num: >
      <Date:dd-mmm-yyyy>
      <Time:hh:mm:ss>
      <User: >
      <Expiry: >
      <Collection:_PLUImpulses>
      <ObjectName: >                Object name + "_" + suffix from above
      <StartDate:dd-mmm-yyyy hh:mm:ss>
      <EndDate:>
      <RData:
         <Data:
            <PLUImpulse:
               <PLU: >                    PLU number. Again!
               <SData:                    Attribute grammar defining characteristics of the stack
                                          button that represents the transaction this PLU impulse
                                          is invoking
                  <StackCaption: >
                  <StackPicFile: >
                  <StackPic: >
               >
               <Desc: >                   Descriptive name for the PLU – displayed in the
                                          EPOSS PLU listing
               <IntDef:                   Interface to call when PLU is selected. Similar (but
                                          simpler) to the same attribute in Interface Items.
                  <Cmd: >
                  <ProductNo: >
               >
            >
            <Mode: >                      Zero or more Mode attributes indicating a valid mode for using
                                          this PLU. (E.g.: RIOP, RISD, RODC, ROOP, ROSD, RV, SC,
                                          TI, TO)
         >
      >
>
```

2.6 Message

Messages define how a system message will be presented to a Horizon user. Elements of the message style such as buttons and titles can be defined in this reference data.

The Message text can contain parameters that are populated with run-time values when the message is displayed.

The attribute grammar is shown below:

FUJ00079219


```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:MessageDefs>
   <ObjectName: >                Unique reference identifying the Message, prefixed by 'MSG'
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_MessageDefs>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <MessageDef:
            <MessageID: >                Message reference number, again!
            <MessageRef: >               Name that identifies this message (e.g.:
                                         MSG_PLUNOEXIST)
            <Caption: >                  Caption that appears at the top of this message when
                                         displayed
            <InterfaceName:              Interface that is invoked when this button on the
                                         message is selected. The attributes shown here are
                                         examples.
               <EPOSSMessage:
                  <Cmd: >
               >
            >
            <ButtonID: >
            <Text: >                     Message text. Named parameters for insertion into the
                                         message text at run-time are enclosed in '%' signs, e.g.
                                         '%StockUnit%'.
         >
         <Button:                        attributes describing the available buttons for this message
            <MessageID: >                ***unknown***
            <ButtonID: >                 ***unknown***
            <CmdCaption: >               Button caption (e.g.: OK)
            <CmdHelp: >                  Help text associated with this button
            <CmdPicFile: >               File containing this buttons icon
            <CmdImageIndex: >            Image number within the image file
         >
      >
   >
>
```

2.7 Event

Event definitions define the properties of a system event. The event text, in common with messages, can contain named parameters that are inserted into the event text at run-time.

FUJ00079219

The user, using specific functionality, invokes events. System events include Log on, temporary lock and report production.

The attribute grammar is shown below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:Events>
   <ObjectName: >                Unique number identifying the Event
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_Events>
   <ObjectName: >                Reference + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <ID: >                  reference number, again.
         <C:text>                Reference used to identify this message.
         <Ca: >                  This appears to be the category of the event. Used values are
                                 Accounting, SU/ User, Access, Reports.
         <T: text>               Event text. Named parameters for insertion into the event text at
                                 run-time are enclosed in '%' signs, e.g. '%StockUnit%'.
         <Ti:text>               Descriptive title for this event
         <PM:                    Attributes used for summarisation and reporting of events
            <L1: >
            <L2: >
            <L3: >
            <L4: >
         >
      >
   >
>
```

2.8 Node

Nodes are used to describe the linking and grouping of products to form a hierarchy. Nodes have accumulators associated with them that describe how transactions are totalled for summarisation and reporting.

The principles behind Nodes and Accumulators are described in more detail in the EPOSS Implementation Description document.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
```

FUJ00079219


```
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:EPOSSNodes>
   <ObjectName: >                Unique number identifying the Node
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_EPOSSNodes>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <NID: >                 Number which uniquely identifies this Node
         <NN:text>               Descriptive name for this Node
         <L: >                   Level in the hierarchy at which this Node sits
         <A:                     Repeating attribute describing all accumulators associated with
                                 this Node
            <N: >                      appears to be the same as the node number (NID)
            <AN: >                     Accumulator name
            <Att: >                    Attribute this accumulator works on
            <Pos:bool>                 Indicates if this accumulator totals in a positive or
                                       negative way
            <F: >                      Indicates how totals are derived using this accumulator.
                                       Valid values are 'Sum' or 'Count'
            <Con: >                    unknown
         >
         <C: >                   unknown
      >
   >
>
```

2.9 Dynamic node

Dynamic Nodes describe dynamic groupings for summarisation and reporting purposes. An example of a dynamic grouping would be on transaction price. It is dynamic, as the specific groups will be different depending on the transactions that form the groupings.

The principles behind Dynamic Nodes and Accumulators are described in more detail in the EPOSS Implementation Description document.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
```

FUJ00079219


```
   <User: >
   <Expiry: >
   <Collection:EPOSSDNodes>
   <ObjectName: >              unique number identifying the Dynamic Node
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_EPOSSDNodes>
   <ObjectName: >              Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <DN: >                Descriptive name for this Node
         <GB: >                Identifies the fully qualified attribute this dynamic node will
                               create groupings on (e.g.: EPOSSTransaction.SaleValue)
         <GI: >                Repeating attribute describing all accumulators associated with
                               this Node. In fact, no repeating version of this exist. The only
                               values it takes are: Every, 16 or 23.
         <GID: >               unknown
         <C:>                  Identifies a single child of this dynamic node.  This attribute
                               may be blank
         <A:                   Repeating attribute describing all accumulators associated with
                               this Node
            <N: >                    unknown
            <AN: >                   Accumulator name
            <Att: >                  qualified attribute this accumulator works on
            <Pos:bool>               Indicates if this accumulate totals in a positive or
                                     negative way
            <F: >                    indicates how totals are derived using this accumulator.
                                     Valid values are 'Sum' or 'Count'
            <Con: >                  unknown
         >
      >
   >
>
```

2.10 SCALES

There are a set of objects that control the scales product. Not all the objects use the same attributes.

The attribute grammar is defined below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:EPOSSScales>
```

FUJ00079219


```
   <ObjectName: >                unique name identifying the object
   <Suffix: >
   <Depend:bool>
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_EPOSSScales>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <Position: >
         <Style: >
         <Caption: >
         <FKey: >
         <CardStyle: >
         <Removable: >
         <Depth:>
         <PicFile: >
         <ImageIndex: >
         <Invisible: >
         <InterfaceName:
            <EPOSSScales:            This is an example of an interface it uses.
               <Cmd: >
            >
         >
         <Text: >
         <ScalesProductNo: >
         <ScalesTimeout: >
         <OtherStampProduct: >
      >
   >
```

The object types used are listed below:

i) Abandon

ii) AddSvcs

iii) BestFitMenu

iv) ChangeQty

v) Complete

vi) Override

vii) PrePaid

viii) ScalesDisp

ix) ScalesMenu

x) ScalesParam

xi) Stamps

xii) StampsDisp

FUJ00079219

The EPOSSScales interface supports the following commands:

i) ScalesAbandon

ii) ScalesAdditionalServices

iii) ScalesChangeQuantity

iv) ScalesFinish

v) ScalesBestFitOverride

vi) ScalesPrePaid

vii) BestFit

2.11 SCALES Service Class

Scales service classes relate to one or more services. These records exist purely to group Scales Services into the service groupings currently in place at Post Offices.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:ScalesSC>
   <ObjectName: >                Identifier for this service class. Number prefixed by 'SC'
   <Suffix: >
   <Depend:bool>                 Identifies whether this product uses the dependency mechanism
                                 to determine whether it is available in a specific outlet.
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_ScalesSC>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <SCN: >                 Descriptive name for this service class
         <SCID: >                Numeric identifier for this service class
      >
   >
>
```

FUJ00079219

2.12 SCALES Service

Scales services describe a particular scales service available at the Post Office Counter and its associated list of weight based tariffs. Customer and Clerk side descriptions are included, as are the lists of associated additional services and tariff information. An attribute is included in each record that links each service to a single service class.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:ScalesSVC>
   <ObjectName: >              Identifier of scales service
   <Suffix: >
   <Depend:bool>               Identifies whether this product uses the dependency mechanism
                               to determine whether it is available in a specific outlet.
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_ScalesSVC>
   <ObjectName: >              Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <SeN: >               Determines the order in which services appear on pick lists
         <DS: >                unknown
         <SN: >                Primary service name which appears on pick lists
         <CN1: >               used to display service information on electronic scales line 1
         <CN2: >               used to display service information on electronic scales line 2
         <PN: >                Product number sold when this service is selected
         <SCID: >              ID of the service class this service belongs to
         <AS:                  Attribute grammar string containing one or more additional
                               service Ids applicable to this service
            <A: >                    Additional Service ID
         >
         <PF:>                 Not used (picture file)
         <II:>                 Image number within PF (not used)
         <Tariffs:             Collection of tariffs
            <T:text>                 Each tariff contains an upper weight band and a price,
                                     e.g. (<T:1200,26>). The lowest weight band less than
                                     or equal to the mail items weight is used to calculate
                                     the tariff
         >
      >
   >
>
```

FUJ00079219

2.13 SCALES Additional service

Scales additional services describe an 'Add on' service that can be selected in addition to the base service. Each additional service has a fixed price associated with it.

The Scales Service records link services to their applicable additional services.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:ScalesAddSvc>
   <ObjectName: >                Additional Service ID. Number prefixed by 'ADDSVC'
   <Suffix: >
   <Depend:bool>                 Identifies whether this product uses the dependency mechanism
                                 to determine whether it is available in a specific outlet.
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_ScalesAddSvc>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <SN: >                  Descriptive name for this additional service
         <AN: >                  unknown
         <SE: >                  unknown
         <TC: >                  Cost of this additional service
         <TP: >                  unknown
         <ASID: >                Additional service ID?
      >
   >
>
```

2.14 Scales Countries

This is a list of countries that are supported by the scales product. Each object contains many countries.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
```

FUJ00079219


```
   <Collection:ScalesCountries>
   <ObjectName: >                Sequence number for the object
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_ScalesCountries>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <D:                     Repeats for each country
            <C: >                      Country name
            <S: >                      Region name (e.g., European, Intnl Zone 1, Intnl Zone
                                       2)
         >
      >
   >
>
```

2.15 Scales Denominations

This lists the denominations of stamps. It also provides the combinations of stamps required for the lower values that are not directly supported. There are two variants of the attribute grammar to support this:

2.15.1 Variant 1

This variant is used to represent specific stamps

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:ScalesDenom>
   <ObjectName: >                Sequence letter for the object (A upwards)
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_ScalesDenom>
   <ObjectName: >                Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
```

FUJ00079219


```
   <EndDate:>
   <RData:
      <Data:
         <SP:
            <ProductNo: >            Product number for the stamp
            <Fast:bool>              unknown
            <Caption: >
            <PicFile: >
            <ImageIndex: >
            <UnitPrice:currency >
         >
      >
   >
>
```

2.15.2 Variant 2

This variant is used to represent combinations of stamp that make up an unsupported value.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:ScalesDenom>
   <ObjectName: >              Sequence letter for the object (A upwards)
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_ScalesDenom>
   <ObjectName: >              Object name + "_" + suffix from above
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <SP:
            <Composite: bool>        Indicates that this value requires multiple stamps
            <UnitPrice: currency>
            <C: text>                Repeating object identifier for each stamp that is
                                     needed to make up this value
         >
      >
   >
>
```

2.16 Report

The Report collection contains all EPOSS reports, with links to one or more report sections contained within the report.

The attribute grammar is defined below:

FUJ00079219

<Message:	
<GroupId: >	
<Id: >	
<Num: >	
<Date:dd-mmm-yyyy>	
<Time:hh:mm:ss>	
<User: >	
<Expiry: >	
<Collection:EPOSSReports>	
<ObjectName: >	Unique numeric ID for the report
<Suffix: >	
>	
<Message:	
<GroupId: >	
<Id: >	
<Num: >	
<Date:dd-mmm-yyyy>	
<Time:hh:mm:ss>	
<User: >	
<Expiry: >	
<Collection:_EPOSSReports>	
<ObjectName: >	Object name + "_" + suffix from above
<StartDate:dd-mmm-yyyy hh:mm:ss>	
<EndDate:>	
<RData:	
<Data:	
<RID: >	Unique numeric ID for the report
<N: >	Descriptive Report Name, used as a default title for this report
<SX: >	X spacing. Not used
<SY: >	Y spacing. Not used
<RN: >	The Node ID that forms the hierarchy driving this report. This field may be blank.
<RL: >	The level number of the specified Root Node ID
<DRN: >	The Dynamic Root Node ID used to drive the dynamic groupings for this report
<MA: >	Defines the mappings attribute used to build the accounting hierarchy driving this report. Usually 'PM', but may be different if the report is based on secondary mappings (i.e. Rem/Transfer detail reports)
<L:>	The number of levels of the accounting hierarchy this report should build. Useful if lower levels of grouping are not necessary (i.e. for totals only reports)
<UC:**bool**>	Use collection - indicates whether this report is based on messages or the members of a persistent object collection. Persistent object collections are used for data that must be modifiable, such as declarations and also for reports based on reference data itself (product lists etc.)
<RC:**bool**>	Raw collection - Indicates if the report will retrieve persistent objects through the lookup server or directly from the message store
<CN:**text**>	The collection name, if UC is true
<S:	One or more sections that make up the report detail
<SID: >	Section identifier
<O: >	Order this section appears within the report
<P: >	Section ID of a parent section (if any)
>	
>	
>	
>	

FUJ00079219

There is an additional, single object within this class that defines which reports should be preloaded. It has the following layout:

```
<Message:
<GroupId:123456>
<Id:1>
<Num:12456>
<Date:16-Sep-1997>
<Time:15:06:55>
<Expiry:90>
<Collection:EPOSSReports>
   <ObjectName:PreLoad>
   <Suffix:00>
   <Depend:False>
   <Version: >
   <CRC: >
>

<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_EPOSSReports>
   <ObjectName:PreLoad00>
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <ReportID: >
      >
   >
>
```

2.17 Report Section

Every report consists of basic units called Report Sections. Report sections are linked together to form a report. Report sections can be used in more than one Report. Report Sections contain one or more section items that can be thought of as 'fields'.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:EPOSSRSect>
   <ObjectName: >              Identifier of report section with a suffix of 01
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
```

FUJ00079219

<Time:hh:mm:ss>	
<User: >	
<Expiry: >	
<Collection:_EPOSSRSect>	
<ObjectName: >	Name of the previous object + "_" + its suffix
<StartDate:dd-mmm-yyyy hh:mm:ss>	
<EndDate:>	
<RData:	
<Data:	
<SID: >	Section ID Identifier for this section. Referenced from Report records
<AP:>	***Page number? If so – not used***
<AX:>	Indicates if this section should appear at particular co-ordinates on a page
<AY:>	Indicates if this section should appear at particular co-ordinates on a page
<OX:>	Indicates if this section should appear at a specific offset from it's starting point
<OY:>	Indicates if this section should appear at a specific offset from it's starting point
<ST: >	The section type. Types are listed in the EPOSS Implementation Description document (e.g., Header|Body|Footer|ReportFooter|ReportHeader|RepeatingNode|)
<SN: >	Descriptive name for the section, used only in debugging
<R: >	The number of rows this section occupies
<C: >	The number of columns this section occupies
<DS: >	Indicates the data source this section is based on. Valid values are listed in the EPOSS Implementation description document. (e.g., RepeatingNode|None|Detail|RepeatingParameter|)
<NID: >	NodeID of the Node this section refers to
<V: >	Contains a value, different dependant on the Datasource type, but usually the level of the NodeID referred to in REPSEC_NODEID
<Items:	Contains one or more items (fields) contained in this section
<I:	
<N: >	Descriptive name for this section item
<IT: >	Item type. Available types listed in the EPOSS Implementation Description document (e.g., CurrentNodeName| CurrentNodeAccumulate| PlainText| CurrentNodeValue| NodeAccumulate| Command| LookupItem)
<FN: >	Font name. Can be 'Default'
<FS: >	Font size
<FB:**bool**>	Bold
<FI:**bool**>	Italic
<FR: >	The number of degrees this section item should be rotated
<F:>	A visual basic format string that is applied to the content of this section item before it is output
<MR: >	Not used
<MOX: >	Micro X. Allows fine tuning of positioning
<MOY: >	Micro Y.
<X:1 >	X position within section
<Y:1 >	Y position within section
<O1V: >	Content depends on section item type, generally describes how item is derived
<O2V: >	ditto

FUJ00079219


```
                    <O3V: >                 ditto
                    <O4V: >                 ditto
                    <SID: >                 Why is this here, its a repeat
                    <DL: >                  Maximum length of section item.  Item is truncated if it
                                            exceeds this length
                    <A: >                   Alignment: 0=Left, 1=Right, 2=Centre
                    <V:bool>                Indicates whether this item appears on output
                    <SZ:bool>               If this item evaluates to zero, the whole section is not
                                            printed
                    <SB:bool>               If this item is blank, the whole section is not printed
                    <SLZ:bool>              Not used
                    <DF: >                  Indicates that this items visibility is dependant on the
                                            content of another section item.  This field contains the
                                            dependency section items name
                    <DO: >                  Operator used in comparison with the dependency field
                    <DC: >                  Value used to compare against dependency section item
                                            using the defined operator
                >
            >
        >
    >
>
```

2.18 Cash account mapping

Cash Account mappings describe how figures from the Office Balance are translated to lines on the Cash Account.

2.18.1 attribute grammar

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:CAMappings>
   <ObjectName: >              Appears to be the product's number this mapping applies to
   <Suffix:00>
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_CAMappings>
   <ObjectName: >              Object name + "_" + suffix from previous object
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <ProductNo: >         Product No on the Balancing hierarchy this mapping applies to
         <Leaf:                Repeats for each table this product appears on?
```

FUJ00079219


```
                <P: >                          This appears to always be the same as ProductNo.
                <Tab:text>                     Name of table
                <N: >                          required line (Node) on the Cash Account?
                <PN: >                         Node ID of the parent node that this product must sit
                                               below for this mapping to apply. The system will
                                               check for a mapping that identifies a specific parent
                                               Node and if none is found will use the mapping with
                                               this field blank (the default mapping)
                <PNL: >                        Level of the parent node required for this mapping to
                                               be used
                <M:                            String placed in the Primary Mappings attribute of the
                                               resultant transaction - will point towards the required
                                               line (Node) on the Cash Account
                   <L1: >
                   <L2: >
                   <L3: >
                   <L4: >
                   <L5: >
                   <L6: >
                   <L7: >
                >
             >
          >
       >
    >
```

There is another, simpler variant of this object. I don't currently know what it does. Its layout is below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_CAMappings>
   <ObjectName: >                  Object name + "_" + suffix from previous object
   <StartDate:m/d/yy h:mm:ss am>
   <EndDate:>
   <RData:
      <ProductNo: >                Product No on the Balancing hierarchy this mapping applies to
   >
>
```

2.19 Pick List

Pick lists can be generated at run time or pre-defined. In the latter case there are two persistent objects to define each pick list. These persistent objects can also contain the row items to place into the pick list. Alternatively the rows can be populated at run time.

The attribute grammar is defined below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
```

FUJ00079219


```
   <Collection:PickListEdit>
   <ObjectName: >                Name of pick list
   <Suffix: >
>
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:_PickListEdit>
   <ObjectName: >                Object name + "_" + suffix from previous object
   <StartDate:dd-mmm-yyyy hh:mm:ss>
   <EndDate:>
   <RData:
      <Data:
         <PLD:
            <Cap:text>                 Caption for the pick list
            <ID:integer>               Unique numeric identifier for this pick list
            <ProductNo:integer>        Product number displayed by this pick list. Null is used
                                       where the product type is not fixed (e.g., for stock)
            <Collate:bool>             Sort the items before displaying
            <C1:integer>               Percentage of the available width taken up by column
                                       1.
            <C2:integer>               Ditto for column 2
            <C3:integer>               Ditto for column3
            <EC: null|1|2|3>           Defines which column of the pick list can be edited
            <CS: >                     The standard Riposte calculator style to be used for
                                       entering data.
            <ML: >                     Maximum length for the calculator
            <Item:                     Optional repeating group for pre-populating the entries
                                       within the pick list.
               <N:text>                The text to place in column 1
               <V:number>              The value in pounds to place in column 2
               <PM:                    Primary Mappings
                  <L1: >               The reporting node for this product
               >
            >
         >
      >
   >
>
```

FUJ00079219

3 Working Data

The following sections describe the attribute grammar records generated as a result of EPOSS activity. The record types include transaction and Cash Account data.

3.1 Accounting Structure

One of the fundamental responsibilities of EPOSS is to maintain the accounting information for an entire outlet.

EPOSS maintains the accounts broadly in line with current POCL practice. Essentially there is an account for each Stock Unit and an account for the Outlet as a whole.

The whole account operates over a sequence of accounting periods (Cash Account Periods - CAP) that are defined centrally. Within a CAP a stock unit can be balanced more than once (but not the outlet). The elapsed time between two balance points is called a Balance Period (BP). The choice of BPs is a local affair. Each stock unit must be balanced immediately before starting a new CAP.

The process of moving to the next CAP is called 'Rollover'. This happens independently for each stock unit and some time later for the outlet.

At each stock unit balance the actual content of the stock unit has to be reconciled with the system-derived view of the content. Three classes of stock are reconciled in this way:

- Cash – this is reconciled by value
- General stamps – this is reconciled by total value
- Value (or inventoried) stock. This includes certain specific types of stamps. This is reconciled by quantity of each type.

Discrepancies between the actual content (or holding) and the system-derived position are recorded as additional (discrepancy) transactions that move the system-derived position to be the same as the actual position.

During the course of a CAP one or more fixed price products may change their price (e.g., a first class stamp is increased from 26p to 28p). This has to be compensated for within the account for each stock unit. Special product types exist to record the change in total value of the stock unit due to a price change for any one product. One re-valuation transaction is written to the account for each product that has changed price during the CAP. If a product changes value more than once during a CAP then just one transaction is still written, but with an aggregate value change.

Immediately after roll over a stock unit is assumed empty. It is then re-populated to its initial position by a set of holding transactions, each of which contains the count, unit price and total value for each product line held within the stock unit.

The balance position (i.e., total value) of the stock unit is written as a special type of holding transaction.

A pseudo stock unit that represents the whole outlet is automatically created by EPOSS. This has a reserved name - ##. It exist mainly to support the cash accounting process. It is

FUJ00079219

only updated at office roll over and at that time is populated with the holding transactions and balance position for each of the 'real' stock units. Hence after a roll over it contains the holdings for the entire outlet.

3.2 Implementation within Riposte

Riposte markers delimit Balance Periods and Cash Account Periods. This is to deal with the problem of potentially disconnected counter machines that are still operable. Any transactions that fall between the markers are deemed to be within that period - even if they occurred earlier in time than the first marker, but were not visible due to a network problem.

The revaluation transactions and the holding transactions have each to be written as an atomic set to cater for machine failures and the possibility of two individuals balancing the same SU on two different, but disconnected counters. This is achieved by given all records of the set the same unique key. A trailer message is written that pulls all these sets together and a single persistent object identifies the correct trailer (in case more than one has been written because of machine failure).

The diagram below shoes the relationship between all these message types:

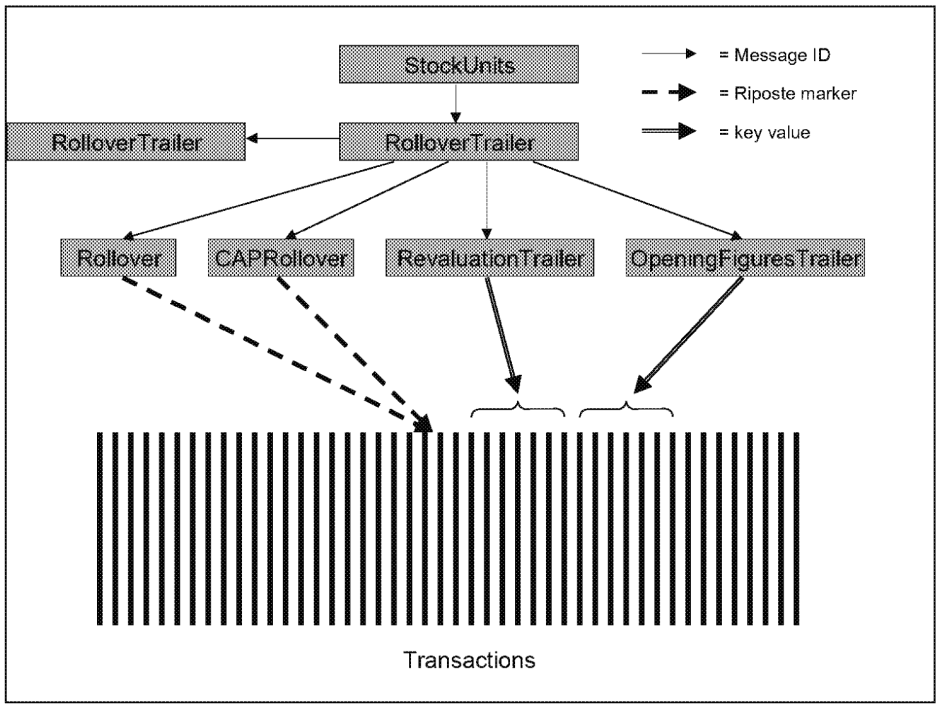

Figure 1 – Stock Unit Account Structure

StockUnits is a persistent object. The other message types are standard Riposte messages.

As stated above a pseudo stock unit is used to represent the entire outlet. This has a similar structure to the above. The diagram below illustrates this,

FUJ00079219

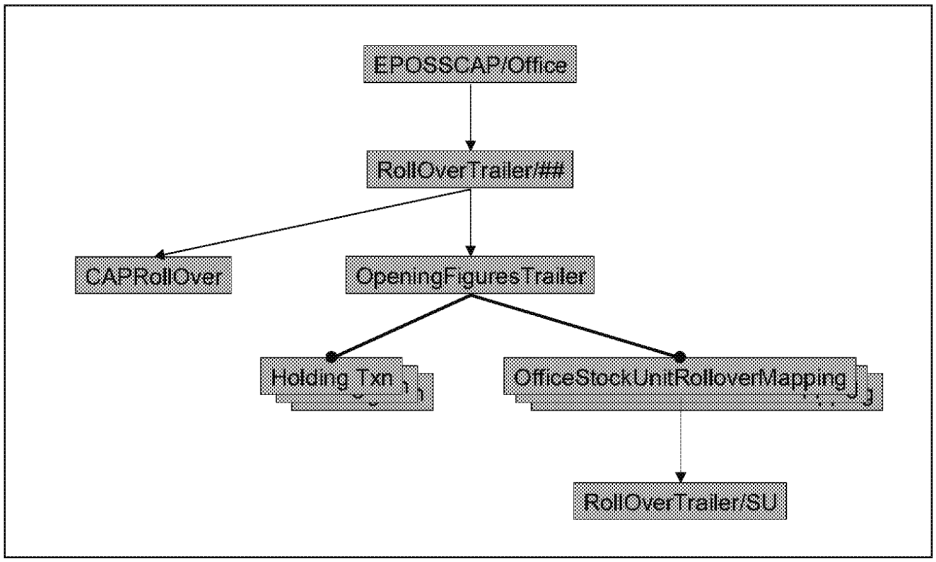

Figure 2 - Office 'Stock Unit' Account Structure

This uses a similar approach to the conventional stock unit in its mixed use of markers, pointers and keys.

The EPOSSCAP persistent object is the head of the structure. It identifies the rollover that occurred at the start of the current cash account period by pointing at the Roll Over Trailer.

The entire office roll over is 'owned' by the Roll Over Trailer for stock unit ##. It identifies that ended the previous CAP; all the holding transactions for the office and all the stock unit data that was used to determine the office position.

Holding transactions and the OfficeStockUnitRolloverMapping messages are identified by the same key value.

The holding transactions are similar to those for a normal stock unit.

One OfficeStockUnitRolloverMapping message exists for each conventional stock unit and identifies the stock unit's rollover trailer that was used to determine the office holding.

The following sections describe the message types that are used to support the accounting mechanism.

3.3 Transaction

3.3.1 Description

A transaction equates to an ARTS line item and is the lowest level of transaction detail recorded. Sale of a stock item and the associated payment from the customer are two discrete transactions.

The data required for transactions was originally derived from 2 main sources:

1. The BA/POCL Interface specification, based on ARTS, which defines the line item information that POCL require from the TMS;

FUJ00079219

2. The additional information necessary to drive the office accounting and EPOSS functionality.

Transactions are described in a similar way to products, containing core and extended attributes. The core data applies to all transactions and the extended data only to transactions of a specific type. The currently described extended transaction type is Additional Data, meaning a transaction where specific extended data has been captured for purposes irrelevant to the core EPOSS and balancing functionality.

3.3.2 Attribute Grammar

<Message:	
<GroupId: >†	
<Id: >†	
<Num: >†	
<Date:dd-mmm-yyyy>†	
<Time:hh:mm:ss>†	
<User: >†	
<Expiry: >*	should be set explicitly (for EPOSS and APS messages to 28)
<TranStartNum: >†	not used by Pathway
<Debit: >*	unsigned pence – used when money is being 'paid' out of the drawer
<Credit: >*	unsigned pence - used when money is being 'paid' into the drawer
<TxnData:	
<SessionId: >‡	unique identifier for all transactions within a customer session. Contains GroupId, Id & Num of one of the messages (normally the first) within the session. The three items are separated by hyphens .
<TxnId:>‡	unique identifier for all the messages within a customer transaction, using a similar algorithm as SessionId.
<Container: *stock unit name* >‡	## represents the entire office
<Start:	
<Date:dd-mmm-yyyy>‡	Date transaction commenced
<Time:hh:mm:ss>‡	Time this transaction commenced
<TF:0-9>‡	Tenths of a second part of the start transaction time
>	
<End:	
<Date: dd-mmm-yyyy >†	Date transaction completed
<Time: hh:mm:ss >‡	Time this transaction completed
<TF:0-9>‡	Tenths of a second part of the completion transaction time
>	
<Mode: >	Contains the mode of the system when this transaction was written. It takes the same value as BlackBoxData.M, below.
>	
<Application: **text**>*	application name generating the transaction. E.g., EPOSSAppMain, SCALES, EOD, etc
<EPOSSTransaction:	
<ProductNo: product reference number >*	
<Qty: >*	Signed count of products. +ve is leaving SU
<PVer: **integer**>*	Product version when this transaction took place.
<SaleValue: **currency**>*	signed value of transaction. +ve is leaving SU
<EntryMethod:0-4>*	Method of data capture: 0 = bar code,

FUJ00079219

	1 = manual 2 = magnetic card 3 = smart card 4 = smart key
<CrossReference: >*	Cross-reference to a previous transaction- using the same format as TxnId. If this transaction is a reversal, this reference points to the reversed transaction
<LkdTxns: **integer**> *	A count of the transactions following this one that are logically linked to this one
<BlackBoxData:	Data provided by extended applications, passed through by EPOSS
<S: >*	unknown but always appears to be 1
<M: **text**>*	Optional attribute to show mode. REC = recovery (ie manual entry of transactions); ER = linked reversal; RV = unlinked reversal; SC = Serve Customer; RISD = Remit in supplies division; RIOP = Remit in other PO; RICL = Remit in client; RIDC = Remit in Data Centre; ROSD = Remit out supplies division; ROOP = Remit out other PO; ROCL = Remit out client; RODC = Remit out Data Centre; TO = Transfer out; TI = Transfer in; HK = housekeeping (adjustments, error notices etc.); DD = ***something to do with discrepancy declarations?***
<LTPN: **integer**>*	Linked Transaction Product Number
<LTSV: **currency**>*	Linked Transaction Sale Value
<HOR:**bool**>*	unknown what this is for – only appears to be used by Scales
<IsLinked:**bool**>*	linked to (a?) previous transaction
<SRC: >	Source stock unit for a transfer
<SRCBP: >	Source balance period? – preceded by a backslash. Only used for transfer out.
<DESTBP: >	Destination balance period? – preceded by a backslash. Only used for transfer in.
<DEST: >	Destination stock unit for a transfer
>	
<AdditionalData:	Zero or more attributes containing data captured as part of this transaction. The number and names of all additional data fields will match those defined in product reference data.
<DETAILS: >*	this is an example
>	
<TranType:C|E|S>*	Flag, used primarily for reporting, to indicate the type of transaction. C = cash account E = event S = serving
<PM:	Primary Mappings - maps this transaction to the EPOSS summarisation hierarchy.
<L1: >*	
<L2: >*	
<L3: >*	
<L4: >*	

FUJ00079219


```
            <L5: >*
        >
        <SM:                        Secondary Mappings - maps this transaction to a second point in
                                    the EPOSS summarisation hierarchy
            <L1: >*
            <L2: >*
            <L3: >*
            <L4: >*
            <L5: >*
        >
        <SUO: bool>*                unknown, but used at roll over of stock
                                    unit with trantype B and O
    >
    <CRC: >†
>
```

Key: * = created by the application

† = created by Riposte

‡ = created by the Retail Broker

Transaction sub-types

Transactions have a common template used throughout EPOSS to track customer service transactions, declarations, Office Summary values, Cash account records and so on.

These transactions can be thought of sub-types as they only make use of a sub-set of the available transaction attributes and are used to drive processes beyond simple customer services.

The following table describes which attributes are used for each transaction sub-type:

AttributeName	Declaration	Balance Rollover	Office Summary	Cash Account
ProductNo:	X	X	X	
Quantity:	X	X	X	X
SaleValue:	X	X	X	X
TranType		X		X
PrimaryMappings:	X	X	X	X
Indicated by:	Mode DD Trantype S	Trantype B	SU ##	Trantype C

Attributes are not shown in this table if they are not used by any of these sub-types

3.4 Cash Account

At office roll over a set of transactions are written that reports the content of each of the cash account lines. Only lines that contain anything are reported on. A subset of the normal transaction format is used – as shown below:

```
<EPOSSTransaction:
    <Qty: >                 negative quantity shown on line
    <SaleValue: >           negative amount shown on line
>
```

FUJ00079219


```
<TranType:C>
<PM:                          node structure for the line
   <L1: >
   <L2: >
   <L3: >
   <L4: >
   <L5: >
   <L6: >
   <L7: >
>
<SnapShot:False>
```

3.5 Event

Events are messages logged in response to particular functionality invoked by the user. The event message is a variant of the normal transaction message. Examples of events include Log on, Temporary Lock and Report production. The format of event messages is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <TxnData:
      <Container:current SU name>
   >
   <EPOSSTransaction:
      <TranType:E>            Flag, used primarily for EPOSS reporting, to indicate the type of
                              EPOSS transaction. In this case it is set to 'E' to signify event
      <ID: >                  Unique Identifier for this event - examples below
      <Ti: >                  descriptive title for this event
      <T: >                   Description of the event, including parameters appropriate to
                              this event type
      <PM:                    Primary Mappings. Allow summarisation and reporting of
                              events
         <L1: >
         <L2: >
         <L3: >
         <L4: >
      >
   >
   <CRC: >
>
```

3.5.1 Example Events

3	Inactive Rollover Failed
4	Inactive SU Rollover
5	Rollover Abandoned
6	Rollover Complete
7	User attached
8	User detached
9	SU Created

FUJ00079219

10	SU Deleted
12	Logon Completed
13	Logoff Completed
14	Office Balance Failed
17	Revluation end
18	Delete SU failed
19	Delete SU failed
20	Delete SU failed
21	Declaration Complete
22	Declaration Abandoned
23	Declaration Complete with Discrepancy
24	Position Locked
25	Position Unlocked
26	Unlock Failed
27	Forced Logoff
28	Report Confirmed
29	Report Printed
30	Report Previewed
31	Inactive Rollover Failed
32	Discrep Committed
33	Balance Checks Failed
34	Balance Checks Failed
35	Revaluation abandoned
40	Cash Acc Created
41	Office CAP rolled
42	Office CAP Roll Abandoned
44	Office Balance Failed

3.6 Print Time Message

```
<Message:
   <GroupId:123456>
   <Id:1>
   <Num:14912>
   <Date:15-Sep-1997>
   <Time:20:03:07>
   <User:ALAN>
   <Expiry:90>
   <PrintInfo:
      <Success:1>
      <Time:00:00:05>
      <Details:>
      <CallerData:>
   >
   <CRC:401443A>
>
```

3.7 Stock unit

Persistent objects are used to represent Stock units in EPOSS.

Stock Unit objects have the following format:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
```

FUJ00079219


```
<Date:dd-mmm-yyyy>
<Time:hh:mm:ss>
<User: >
<Expiry: >
<Collection:StockUnits>
<ObjectName: stock unit name >
<Data:
   <Shared:bool>            is SU shared
   <Locked:bool>            not used (planned for smart card support)
   <LockedBy:>              not used
   <CAP:>                   Cash Account period
   <BP:>                    Balance Period within CAP
   <AttachedUsers:
      <User:                One instance for each user attached to this SU
         <UserName: >
         <IsLoggedOnAt: >          Indicates that user is currently using the SU at this node
                                   number
      >
   >
   <BalanceStatus:Clean|Dirty|Balanced|BalancedBP>
                            Status indicator, used to enforce business rules such as when it is
                            valid to delete a Stock Unit
   <Declaration:            One or more of these according to:
                            a) how many declarations currently exist (3 types at release 1)
                            b) how many declarations have taken place for each declaration
                            type (for shared stock Units)
      <ID:2|3|4>            Declaration type ID
      <DrawerID: >          Always 0 for single stock units. Is this redundant at this level?
      <TotalValue:currency>          The total monetary value of the declaration. This is the
                            sum of the component drawer declarations (see below).
      <LastModified:        Date and time this was last modified.
         <Date: >
         <Time: >
      >
      <Status:Declared|UnDeclared>
                            indicates whether this declaration is outstanding
      <User: >              User who did the declaration. Will be blank until first
                            declaration.
      <ProductNo:0|1|43>    Product this declaration relates to
      <ReqInd:bool>         is this declaration required for individual stock units?
      <ReqShrd:bool>        is this declaration required for shared stock units?
      <Name:Stock|Stamps|Cash>
                            product name
   >
   <RolloverTrailer:        This is the message number of the RolloverTrailer message for
                            the preceding Balance Period.
      <GroupId: >
      <Id: >
      <Num: >
   >
   <Version: >
   <CRC: >
>
```

3.8 Drawer Declaration

A shared stock unit may be comprised of several physical drawers. Hence, during the process of declaring a stock unit each of these drawers has to be declared independently.

FUJ00079219

The declarations for each of the drawers is held within up to 3 transient persistent object that are removed once the stock unit has been rolled over. There is one persistent object for cash, one for stock and one for stamps. Each holds the total value for that class of product within that drawer.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <User: >
   <Expiry: >
   <Collection: >                 The collection name is constructed from the literal 'Drawer'
                                  followed by the stock unit name.(e.g., 'DrawerYYS')
   <ObjectName: >                 The object name is comprised of 2 parts:
                                  one of the literals 'Cash', 'Stock', or 'Stamps'
                                  followed by the DrawerID. (e.g., 'Cash1' or 'Stock2').
   <Declaration:
      <ID:0>                      unknown
      <DrawerID: >                The drawer id that the user chose to enter when declaring
      <TotalValue: >              The total value for this class of stock for this drawer
      <LastModified:
         <Date: >
         <Time: >
      >
      <User: >
   >
   <Version: >
   <CRC: >
>
```

3.9 Rollover Trailer

This message is the last message that is written during a rollover. It identifies all the other messages involved in the roll over. It is identified from the EPOSSStockUnit persistent object. This mechanism has been developed to protect the system from parallel roll-overs being undertaken on disconnected counters. Once the counters are re-connected only one roll-over will 'win' – as identified by the surviving persistent object

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <User: >
   <Expiry: >
   <Data:
      <TranType:RolloverTrailer>
      <Container: >
      <CAP: >
      <BP: >
      <Rollover:                  Message number of the BP rollover trailer message
         <GroupId: >
         <Id: >
         <Num: >
      >
      <CAPRollover:               Message number of the CAP rollover trailer message
         <GroupId: >
```

FUJ00079219


```
            <Id: >
            <Num: >
         >
         <OpeningFigures:          Message number of the opening figures trailer message
            <GroupId: >
            <Id: >
            <Num: >
         >
         <Revaluation:             Message number of the revaluation trailer message
            <GroupId: >
            <Id: >
            <Num: >
         >
         <Previous:                Message number of the previous rollover message. Using this,
                                   balance periods can be chained together
            <GroupId: >
            <Id: >
            <Num: >
         >
      >
      <CRC: >
>
```

3.10 Balance Period Rollover Marker

This message contains a marker that is used to delimit the end of a balance period.

Note that the CAP and BP attributes within this message are for diagnostic purposes only and should not be used for retrieval. Retrieval of these messages should only be through the message hierarchy defined at 3.1 Accounting Structure.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <User: >
   <Expiry: >
   <Data:
      <TranType:RollOver>
      <Container: >          Stock unit identifier
      <CAP: >                Cash account period for transactions that follow this marker
      <BP: >                 Balance period for transactions that follow this marker
      <Mark:                 Riposte marker
         <1: >
      >
   >
   <CRC: >
>
```

3.11 CAP Rollover Marker

This message contains a marker that is used to delimit the end of a cash account period.

Note that the CAP and BP attributes within this message are for diagnostic purposes only and should not be used for retrieval. Retrieval of these messages should only be through the message hierarchy defined at 3.1 Accounting Structure.

```
<Message:
```

FUJ00079219


```
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <User: >
   <Expiry: >
   <Data:
      <TranType:CAPRollOver>
      <Container: >          Stock unit identifier
      <CAP: >                Cash account period for transactions that follow this marker
      <BP: >                 Balance period for transactions that follow this marker
      <Mark:                 Riposte marker
         <1: >
      >
   >
   <CRC:F55A75FD>
>
```

3.12 Revaluation Trailer

This message is used to terminate the set of revaluation transaction that may follow a balance period. All the revaluation transactions have an attribute 'RevalautionId' that has the same value as that attribute in this message. These revaluation messages actually 'belong' to the preceding balance period.

Note that the CAP and BP attributes within this message are for diagnostic purposes only and should not be used for retrieval. Retrieval of these messages should only be through the message hierarchy defined at 3.1 Accounting Structure.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <User: >
   <Expiry: >
   <Data:
      <TranType:RevaluationTrailer>
      <Container: >          Stock unit identifier
      <CAP:2>                Cash account period for transactions that follow this marker
      <BP:1>                 Balance period for transactions that follow this marker
      <RevaluationId: >      A key value that is used in all the revaluation transactions that
                             precede this trailer.
   >
   <CRC:CA427AFA>
>
```

3.13 Opening Figures Trailer

This message is used to terminate the set of opening figure transactions that will follow a balance period. All the opening figure transactions have an attribute 'OpeningFiguresId' that has the same value as that attribute in this message.

Note that the CAP and BP attributes within this message are for diagnostic purposes only and should not be used for retrieval. Retrieval of these messages should only be through the message hierarchy defined at 3.1 Accounting Structure.

FUJ00079219


```
<Message:
   <GroupId:123456>
   <Id:1>
   <Num:14940>
   <Date:15-Sep-1997>
   <Time:20:09:42>
   <User:ALAN>
   <Expiry:90>
   <Data:
      <TranType:OpeningFiguresTrailer>
      <Container: >            Stock unit identifier
      <CAP: >                  Cash account period for transactions that follow this marker
      <BP: >                   Balance period for transactions that follow this marker
      <OpeningFiguresId: >     A key value that is used in all the opening figures transactions
                               that precede this trailer.
   >
   <CRC:C5281559>
>
```

3.14 Outlet Status

The outlet status persistent object is a pair of objects containing outlet status information used to enforce business rules such as when it is valid to produce a Cash Account or rollover the office

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:EPOSSOutlet>
   <ObjectName:AccStatus>
   <Data:
      <Status: Clean | Dirty >
   >
   <Version:>
   <CRC: >
>
```

3.15 Cut-Offs

The Cut-off object contains a marker and is used to mark the point at which a report is produced

The object name is constructed from two components: the first is the SU name and the second contains the numeric report type.

It is used to reduce to a minimum, the part of the message store that must be searched to compile a report. Cut-offs are created when the user indicates a report to be accurate and complete[1], in order to ensure the confirmed transactions do not appear on a subsequent report.

```
<Message:
```

[1] By pressing the cutoff button on the desktop

FUJ00079219


```
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection:CutOffs>
   <ObjectName: >                Unique Cut-off ID, as described above.
   <TideMark:                    An EPOSS specific string used to determine if transactions for
                                 this cut-off have occurred since the cut-off was last created or
                                 updated
      <Data:
         <DateTime:17/09/97 21:36:59> not 2000 conformant
                                 This is the time that the first message that contributed to this
                                 report was written to the message store. It is not the actual time
                                 of the transaction.
         <Seq: >                 This is a construction of node number (Id) concatenated with
                                 message number (Num) for the first message that contributed to
                                 this report. There is no delimiter between Id and Num and hence
                                 this cannot support node numbers greater than 10.
      >
   >
   <Mark:
      <1: >                      A Riposte marker representing the point in the message store
                                 this cut-off applies to
   >
   <Version: >
   <CRC: >
>
```

In addition to the above persistent object, a 'Cutoff' message is also written that summarises the content of the particular report.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <User: >
   <Expiry: >
   <CutOffID: >                  The numeric report type this cutoff applies to
   <TranType:C>
   <Summary:
      <SU: >
      <CAP: >
      <BP: >
      <SV: >                     The total value reported
      <Qty: >                    The number of transactions contributing to the report
   >
   <PM:
      <L1: >
      <L2: >
      <L3: >
      <L4: >
      <L5: >
   >
   <SM:>
   <CRC: >
>
```

FUJ00079219

3.16 Declaration

A series of persistent objects is written for each product declared for a specific stock unit. It is not used for cash declaration or stamps declaration. There is a separate collection for each stock unit. The object name reflects the product number.

The attribute grammar is defined below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date:dd-mmm-yyyy>
   <Time:hh:mm:ss>
   <User: >
   <Expiry: >
   <Collection: >                 The collection name is constructed from three components:
                                  "Dec" followed by the stock unit name followed by the number
                                  02. (E.g., DecAAA02)
   <ObjectName: >                 The object name is constructed from two components: the
                                  declaration id, followed by a four numeric character
                                  representation of the product number. (E.g., 010041)
   <TxnData:
      <Container: >               Contains the stock unit name
   >
   <EPOSSTransaction:
      <ProductNo: >               Product number
      <Qty: >                     The (negative) quantity of the product declared
      <SaleValue: >               The (negative) total value of the product declared (e.g., ten 26p
                                  stamps would result in –2.60)
      <DSM:                       Declaration secondary mapping?
         <L1: >
         <L2: >
         <L3: >
         <L4: >
         <L5: >
      >
   >
   <Version: >
   <CRC: >
>
```

3.17 Cash Account Snapshot Object

When a cash account is produced the results are written as a set of persistent objects within the CASnap collection. The object name is the node number for the line within the cash account. Objects are only produced for those lines that have an entry.

The attribute grammar is described below:

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <Expiry:28>
   <Collection:CASnap>
   <ObjectName: >
```

FUJ00079219


```
   <EPOSSTransaction:
      <Qty: >                    negative quantity on cash account line
      <SaleValue: >              negative value on line
   >
   <TranType:C>
   <PM:                          Node mapping of cash account line
      <L1: >
      <L2: >
      <L3: >
      <L4: >
      <L5: >
      <L6: >
      <L7: >
   >
   <SnapShot:True>
   <Version:>
   <CRC: >
>
```

3.18 EPOSSStockUnit

This collection is used internally by EPOSS and is of little interest elsewhere.

There are a set of static configuration objects in this collection:

- CashAccount
- Declarations
- FinalBalance
- IgnoreProducts
- Parameters
- Revaluation

Also there is a dynamic object 'CASnapShot' that contains a signal that is set after all the CASnap objects have been produced. This signifies that the snap shot is complete. After the roll over the signal is unset.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <Expiry: >
   <Collection:EPOSSStockUnit>
   <ObjectName:CASnapShot>
   <StartDate: >
   <EndDate: >
   <RData:
      <Data:
         <Produced:bool>
         <RolloverTrailer:       Only present when Produced is true, it identifies the rollover
                                 trailer for stock unit ## that is being used by this rollover.
            <GroupId: >
            <Id: >
            <Num: >
         >
      >
```

FUJ00079219


```
   >
   <Version >
   <CRC: >
>
```

3.19 EPOSS Outlet

This object is used internally by EPOSS and is of little interest elsewhere.

This is used to indicate the status of the office 'account'. It is normally set to 'Dirty' except immediately after a cash account has been produced, when it is set to 'Rollable'

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <Expiry: >
   <Collection:EPOSSOutlet>
   <ObjectName:AccStatus>
   <StartDate: >
   <EndDate: >
   <RData:
      <Data:
         <Status:Dirty|Rollable>
         <RolloverTrailer:
            <GroupId: >
            <Id: >
            <Num: >
         >
      >
   >
   <Version: >
   <CRC: >
>
```

Only present when Staus is 'rollable', it identifies the rollover trailer for stock unit ## that is being used by this rollover.

3.20 EPOSS CAP Object

This object is written to record the current Cash Account period for the office. It is written after the office roll-over. The attributes are self explanatory.

```
<Message:
   <GroupId: >
   <Id: >
   <Num: >
   <Date: >
   <Time: >
   <Expiry: >
   <Collection:EPOSSCAP>
   <ObjectName:Office>
   <StartDate: >
   <EndDate:>
   <RData:
      <Data:
         <RolloverTrailer:
            <GroupId: >
            <Id: >
            <Num: >
```

Identifies the rollover trailer for stock unit ## that was created by the last CAP rollover for the office.

FUJ00079219


```
                >
                <PreviousCAP: >
                <CAP: >
                <Year: >
                <StartDate: >
                <EndDate: >
            >
        >
        <Version: >
        <CRC: >
    >
```