WaveWorks Portable Shopping System Integration Guide

February 1, 2000
Version 1.10

Integration Services Group
Pittsburgh, Pennsylvania
USA

Revision History

Revision	Date	Author	Comments
1.00	December 31, 1999	Mary Wroniak Garry Simmons Brian Reed Tom Pike Mike Tierney Luke Petrozza	Initial version
1.10	February, 1, 2000	Luke Petrozza	Added PST 008 and PST 038 transactions

Table of Contents

1. Introduction

1.1 Purpose of Document

This document is an aid to be used in the configuration, customization and installation of the WaveWorks Portable Shopping System (PSS).

1.2 Assumptions

This document assumes that the reader is familiar with the functionality of Symbol's WaveWorks architecture, the WaveWorks Client, and the Server Enabler. In addition, the reader should be familiar with the Windows NT Operating System, and third-party "browser" software.

1.3 Reference Documents

- Portable Shopping System Functional Specification
- Portable Shopping System Design Document
- Portable Shopping System Users Reference Guide
- ScreenMaker Users Reference Guide

2. Architecture Overview

The Portable Shopping System utilizes Symbol Technologies' proprietary WaveWorks three-tiered architecture, which provides clear separation of the user interface, business logic and database portions of the system. For the PSS, WaveWorks provides application messaging functions and manages distributed objects on the Microsoft Windows NT platform.

2.1 Introduction to WaveWorks

WaveWorks is a light weight, high performance, three-tiered, application messaging and distributed object management architecture which is designed to run on the Windows NT system platform. As a horizontal system component, WaveWorks can serve as the backbone architecture for a wide variety of multitiered client/server applications.

The WaveWorks system provides the following application services:

- Thin client radio terminal environment (reduces system administration costs)
- Radio Terminal display and client state management
- Client-to-Server and Client-to-Client messaging service
- Business Service object broker
- Configurable Business Servers
- Distributed business service objects
- Load-based object instancing
- Load balancing
- Service based instancing
- DBMS connection sharing
- Client specific object instancing
- Web-based system administration.
- Application user password validation and security service

2.1.1 WaveWorks Design Objectives

Three-tiered Architecture—WaveWorks provides a platform for implementing applications that maintain a clear separation between the user interface, business logic, and database layers.

Light Weight—WaveWorks is designed to run on P200 systems or better.

Adaptable—COM object-based design allows easy addition of new business service objects to support new applications. The encapsulation of data access inside business objects makes it easy to change data sources in the future, without rewriting the entire application.

Business Object Reuse—COM object-based design allows easy reuse of existing business service objects to support new applications.

Scaleable—A distributed-object architecture makes it easy to add additional business service capacity, based on changing business demands.

Distributed—WaveWorks system components are designed so they can be deployed across LAN or WAN networks. Network utilization can be optimized by placing the WaveWorks Business Server components near or on the DBMS server, while running the Display Server / Message Server components near or on the client connection point server.

High Performance—Multithreaded business service execution, avoidance of persistent message queues, event driven implementation (no polling), and automatic load balancing permit message rates of up to 500 per second (on a single system configuration).

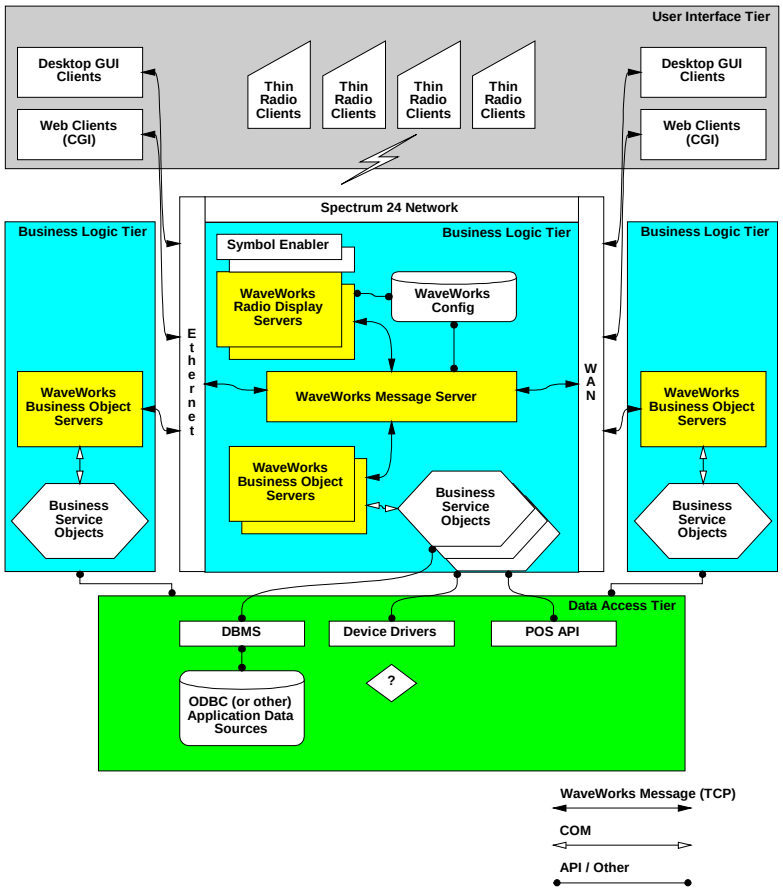
Reliable—WaveWorks supports the ability to configure multiply redundant parallel Business Server components. In addition to load sharing, the redundant servers will automatically hot-swap / load-shift to the remaining servers if one or more server systems suffer a complete or partial failure.

Platform Independence—Dependency on NT specific features are minimal and are internally well isolated.

Minimum Dependency on Third-party Components—To improve platform portability and reduce licensing costs, WaveWorks depends on only generally available add-on components, such as: an ODBC compliant DBMS, Winsock 1.1, an http server, and a third-generation (or better) Web browser. A suitable version of all of these components is included with NT 4.0 server.

2.1.2 WaveWorks Components

The WaveWorks components are shown in the following diagram and described below.



Display Server—Manages the user interface and screen flow for radio-based, thin client applications. Maintains client data variables to record the state of the client application. Invokes business services via WaveWorks messages.

Message Server—Provides light weight message routing and load balancing services between the various WaveWorks system components. Supports Command / Response style communication for the execution of business services, and unsolicited client-to-client messaging.

Business Server (Object Broker)—Manages the creation, destruction, and invocation of COM-based business service objects. Automatically creates and destroys object instances based on system demand. Permits sharing of database connections by multiple clients. The WaveWorks architecture supports multiple Business Servers, running on either local or remote systems.

WaveWorks Messages—The WaveWorks messaging service uses a common message format for all WaveWorks messages. This format contains a fixed header, followed by zero or more application dependent data fields. TCP is used as the transport protocol for all WaveWorks messages. This permits the business logic tier to reside on a remote system, if desired. The messages form the "glue" that cements the client interface tier of the application to the business logic tier. The messaging service can also be used to implement direct "client-to-client" messaging.

Business Service Objects—Contain the vertical business application logic, implemented as Microsoft COM automation server objects. The term "Service Object" is used to describe the COM interface subset to which all WaveWorks-managed objects must adhere.

Transaction Model—WaveWorks uses a transaction Id scheme to provide namespace services for purposes of automatically routing request messages to a capable Business Server for execution.

Development Tools (*Optional Component*)—An application interface development tool (*Screen Maker*) supports rapid development of thin client radio applications. A radio terminal simulation tool (*Screen Runner*) permits testing of Screen Maker applications without the need for an actual radio network. A WaveWorks C++ foundation class library is also available, which includes classes that support ODBC data access, WaveWorks messaging, INI file access, Date / Time manipulation,and more.

2.2 PSS Software Overview

Built upon the WaveWorks core software, the PSS software consists of

- COM Objects
- NT Services
- Display Server Scripts
- Database

The COM Objects and NT Services perform the business logic. That is, they are responsible for the data manipulation and system sequencing that makes the system a portable shopping system. The methods contained in the COM Objects and the NT Services are invoked via WaveWorks transactions. The COM Objects are dynamically created and deleted by the Business Server. The NT Services are started at system startup and remain resident in the system.

The Display Server scripts provide the user interface mechanism for the hand held terminals and entrance units. The scripts define the screen layout and provide the handling of user actions for those devices. When necessary, the scripts invoke business methods via WaveWorks transactions.

The database contains all of the persistent data of the system. The database is used to store all information about the devices, shoppers, items, and system configuration as required.

2.3 Major Subsystems

2.3.1 Unit Management Subsystem

The Unit Management Subsystem (UMS) manages the Symbol hardware devices that comprise the PSS system. Those devices include hand held terminals, cradles, entrance unit devices, ticket printers, and power supplies. UMS ensures that the devices work in concert with one another to provide a seamless integrated system.

2.3.2 Shopping Trip Subsystem

Once the shopper retrieves the proper hand held terminal from the dispenser, the Shopping Trip application allows the shopper to add and delete items from their basket and view the “totals” information for their current shopping trip. Shopping trips can be ended by returning the hand held terminal to an empty dispenser slot or by scanning an “End of Trip” barcode. Each activity during the shopping trip is logged and available for review from the Service Terminal.

2.3.3 Quick Order Subsystem

While shopping, a user may choose to place a quick order through the optional Quick Order subsystem provided. The shopper identifies, through the quick order mechanism, the products and the quantities/weights of those products they wish to order. The quick order can then be sent to the appropriate department, where a web screen displays to store personnel the orders placed by shoppers. After they have finished processing the order, store personnel can send a message to the shopper that their order is ready to be picked up.

2.3.4 Queue Busting Subsystem

The Queue Busting subsystem allows the PSS System to be used in a slightly different manner than normal, but also helps retailers and shoppers reduce the amount of checkout time. Any hand held terminal which has been released can be used for queue busting. This is normally done by store personnel to alleviate long checkout lines which may have formed.

Transactions in this mode are initiated by scanning a special (configurable) barcode. More than one barcode can be configured for use in this manner. This barcode is expected to be on a plastic token. Once the special barcode has been scanned, the items in the customer basket are scanned. A PSS transaction is created in the same manner as a normal PSS transaction. After all items in the basket have been scanned, the special barcode, which began this mode, is scanned again.

This transaction is sent to the POS System. The plastic token containing the barcode is then handed to the customer with instructions to give it to the cashier at the checkout register. When the customer reaches the register, they present the token and their loyalty card (if used) to the cashier who then performs a standard non-audit self scan checkout. After the customer checks out, the token can be re-used for other customers.

2.3.5 Express Shopper Subsystem

The Express Shopper subsystem also allows the PSS System to be used in a slightly different manner than normal, and also helps retailers and shoppers reduce the amount of checkout time. A self-scanning kiosk is installed near the express checkout lanes in the store. A customer picks up a terminal, scans their items, and replaces the terminal into the dispenser slot. The customer then proceeds to the PSS checkout lanes and presents their ticket or loyalty card to the cashier who then performs a standard non-audit self scan checkout.

2.3.6 Electronic Marketing Subsystem

The PSS system also allows for a basic electronic marketing facility through the Electronic Marketing subsystem. This application allows a retailer to send messages to a shopper's hand held terminal anytime that a given item is scanned.

2.3.7 User Messaging

Also, through the User Messaging application, the store has the option to send messages to shopper's hand held terminals. The messages can be any text, and can be made to be repeated, to be displayed at fixed times during the day, and/or to be directed to any set of shoppers currently in the store. In addition, messages can be "pre-loaded" such that they are displayed to the user the next time they enter the store and use the PSS system.

2.3.8 Transaction Ticket Printing

The Transaction Ticket Printing application prints a transaction ticket for the shopper at the conclusion of their self-scan shopping trip. The ticket is used to direct the shopper to the self-scan or quick pay lanes. The ticket also provides a mechanism, when scanned, for notifying the POS System that this is a PSS transaction. Using a transaction ticket is optional.

2.3.9 POS Interface

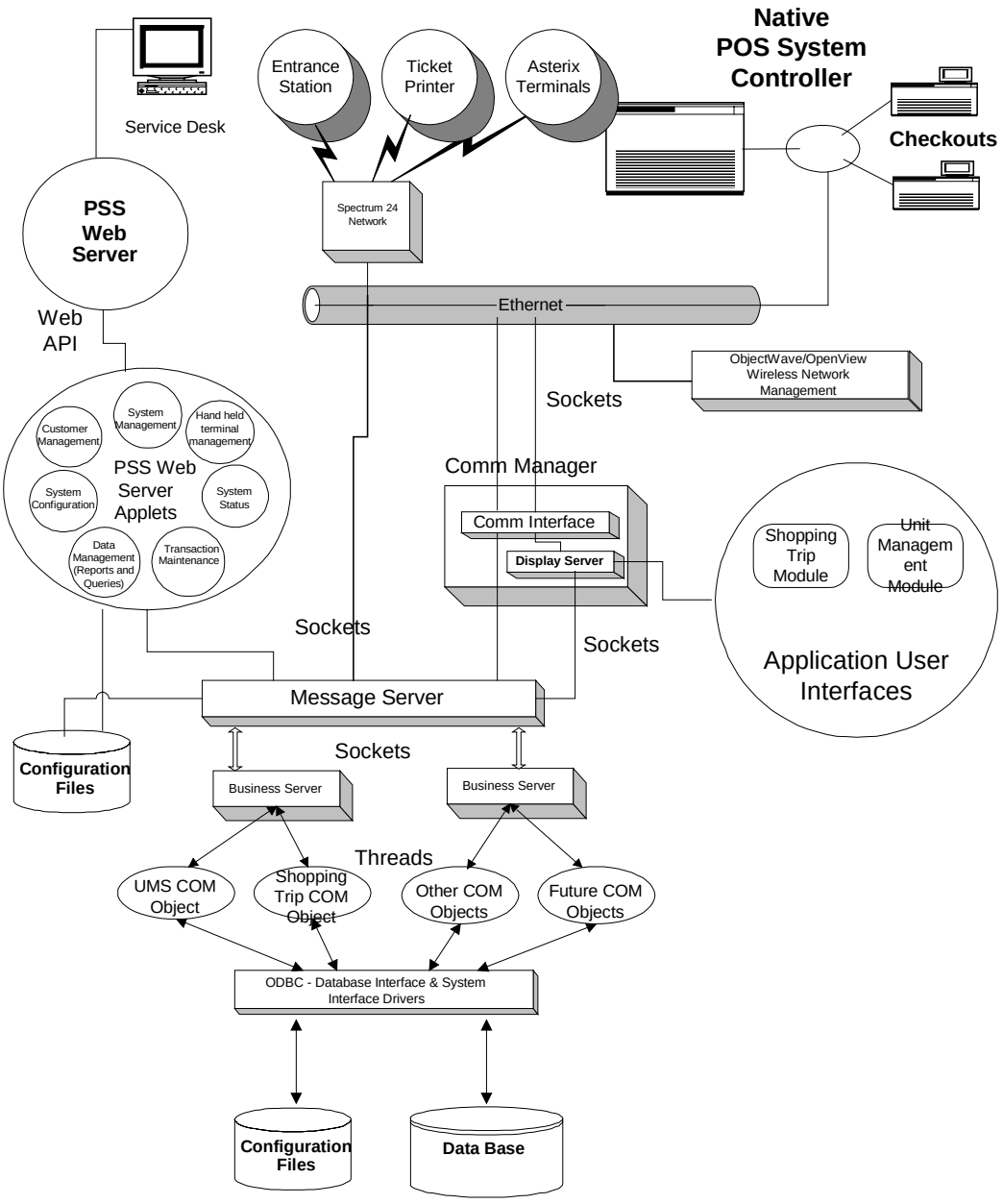
The POS Interface application provides the mechanism by which the PSS System can be fully integrated into a store's environment. It handles price file updates and Scan-In/ Scan-Out file mechanisms.

2.3.10 System Administration

Further, a system administration application is provided which allows store personnel to adjust and monitor the PSS system and data.

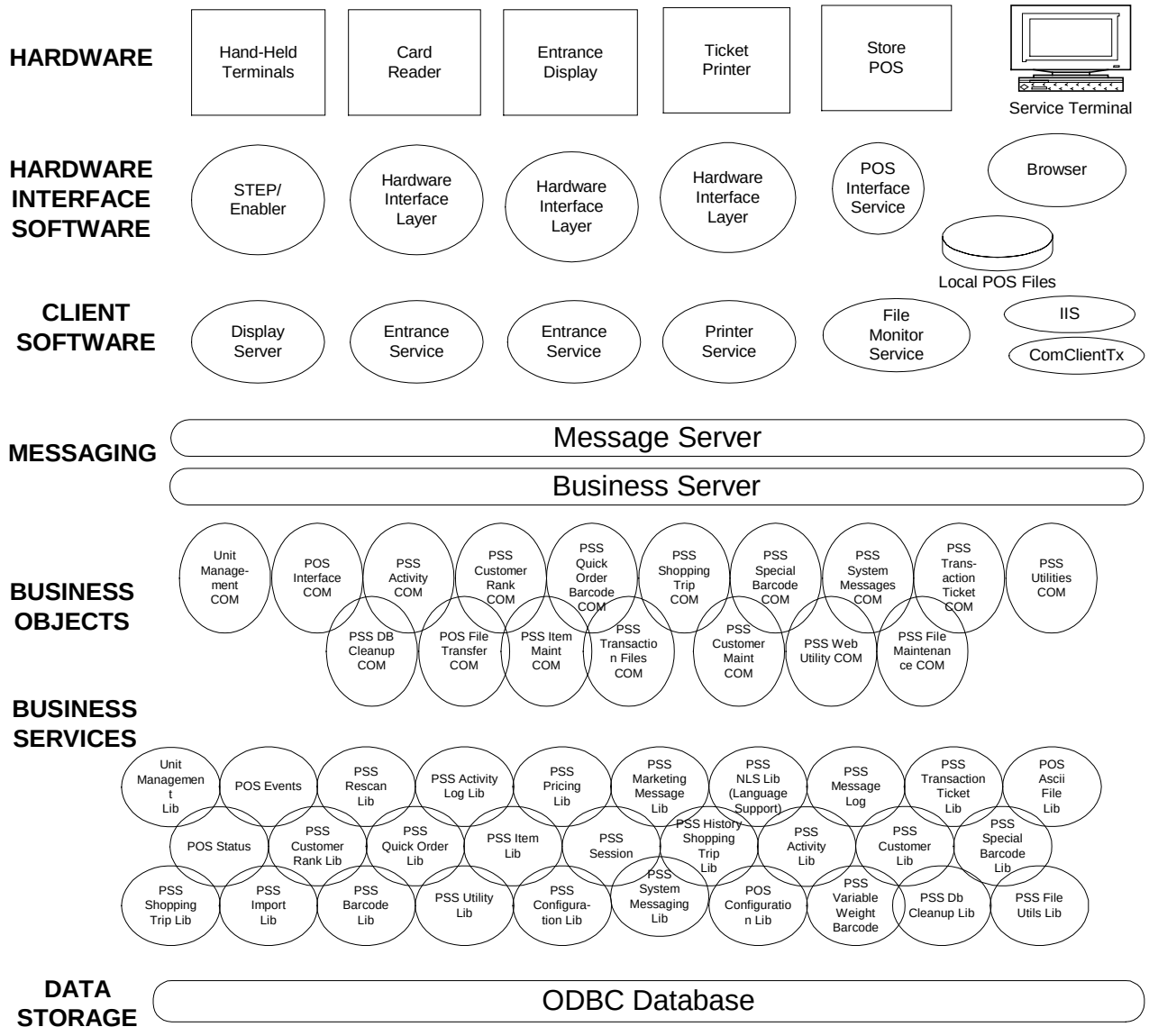
2.4 PSS Overview Diagram

The following diagram depicts the PSS Software in the context of a WaveWorks system. The system hardware interfaces occur via the Spectrum 24 network. The hand-held terminal user interfaces are managed as shown in the Application User Interfaces. The Web screens used at the Service Desk Terminal interface via the Web Server. The Message Server routes messages through the Business Server to COM objects that perform the PSS business logic and provide any necessary database manipulations.



2.5 PSS Component Diagram

The following diagram gives a more complete listing of the software components included in the WaveWorks PSS system. Further details of these components are provided in subsequent chapters.



3. Hardware Requirements

3.1 Windows NT Computer

A Windows NT Machine is required with approximately the following configuration:

- Intel-based PC (minimum 233 MHz Pentium)
- Windows NT Server version 4.0 (NT Server needed to perform DHCP functionality; if some other host in store is a DHCP server where PSS files can be written, NT Workstation is sufficient.)
- VGA monitor
- Mouse
- xx Mb of free space on the hard disk—disk sizing is dependent on length of time historical data is to be stored in database
- minimum 128 Mb RAM
- CD drive
- Ethernet adapter card
- Token ring adapter card (optional)

The build environment expected is:

- IBM RCO Version 2.3 NT Client Support
- ODBC-compliant database (SQL Server is recommended)

3.2 Hand Held Terminals

- Symbol Technologies' CST 2040
- Asterix 2 RF; 5-key model
- Asterix 3, RF

Note: See Symbol product literature for available terminal types and options.

3.3 Terminal Dispenser/Entrance Unit

- Symbol Technologies' CPD 20XX Dispenser - 96, 64, or 32 slot (for Asterix 2 terminals)
- Symbol Technologies' Asterix 3 Standard Charging Cradle
- Entry Station barcode reader or magnetic stripe reader
- Standard furniture
- Optional ticket printer

Note: See Symbol product literature for available dispenser types and options.

3.4 Spectrum 24™ Radio Network

The hand held terminals communicate to the Windows NT computer via Symbol's wireless radio system. The terminals send and receive radio messages via multiple "access points" located throughout the store.

4. PSS Software Installation

This installation guide is intended to provide a roadmap for installing and configuring the PSS system and all supporting software. The reader should be familiar with the Windows interface and installing software.

This guide assumes the software is being installed on a new machine. It is strongly recommended that the target computer be dedicated to running PSS and that you install the software onto a freshly formatted disk.

Before Starting

The following media will be needed during the installation process:

- Bootable Microsoft NT Server Version 4.0 Installation CD
- NT "Getting Started" book (with the Certificate of Authenticity on the front)
- NT 4.0 Service Pack 5 CD
- NT 4.0 Option Pack CD
- Video and Network Card drivers
- Microsoft SQL Server 7.0 (Standard) CD
- Microsoft Access CD or Microsoft Office 97 Professional Edition CD
- Symbol PSS Setup CD (includes WaveWorks, PSS, and customer changes)

The following information will be needed during the install:

- Administrative account and password for the computer
- The IP address and hostname of the target computer. (It is suggested that the target computer use a private IP subnet. The 172.16.N.N addresses are typically used with a subnet mask of 255.255.255.0; this allows for 256 hosts on the subnet**).**
- The range of IP addresses to allocate for DHCP use. If you need to have the target computer on the office/store network, make sure that there are no other DHCP servers on the same subnet. If that is not possible, then install a second network adapter and use it to connect to the outside world, leaving the private IP subnet for PSS use.
- Access Point ESS ID (aka Net ID)

4.1 Install NT 4.0 Server

The first step in the installation process is to install the NT 4.0 Server operating system. It is strongly recommended that you do a clean installation of the entire system. Many of the more expensive server PCs come with customized installation programs that walk the user thru the NT installation process. If your PC comes with a special installation CD, you should use it, but read this section first to identify the places where you will have to change settings from the default values. This section describes the NT installation process using the standard Microsoft NT Server installation CD.

Below are the specific steps to follow:

Turn the computer on. During the boot-up process, enter the CMOS setup—however that process occurs on your machine. Look for a message that says which key to press to enter Setup (Delete and F2 are popular choices).

- Set the CMOS to boot from the CD-ROM drive.

- Insert the NT Server CD Installation in the CD-ROM drive, then exit from the CMOS settings. The system should now boot from the CD.

- The blue "Windows NT Setup" screen is displayed. Wait while various system files and drivers are loaded.

- The blue NT boot screen is displayed, then it goes back to the "Windows NT Server Setup" screen.

- Press Enter to install Windows NT now.

- The Mass Storage screen is now displayed. This section of the setup process is hardware specific. Most corporate PCs do not have SCSI interfaces installed in them. Many server PCs do come with SCSI interfaces, so you'll need to know what kind of hard disks your system has before continuing. Press Enter to skip searching for SCSI devices. Press "S" to check for them.

- The Licensing Agreement is displayed. Read the Agreement (press the Page Down key to advance to the next page) until end of End User License Agreement (EULA) verbiage is reached.

- Press F8 to accept the Licensing Agreement.

- The Server Setup screen is displayed which lists any previous installations of NT. Select "N" to install a fresh copy of NT Server.

- The Components list is displayed. Press Enter to accept the list of standard hardware/software components.

- The Partition options screen is displayed. It is very important to pay attention to what you are doing here. This section is also hardware specific.

- Start off by deleting all existing disk partitions. Select each existing partition (use the arrow key to move the highlighted partition) and following the on-screen instructions.

- The screen should now only show "Unpartitioned space" since you just deleted all existing partitions. The next step is to create a partition that NT will be installed onto. Depending on the size of your hard drive, you'll want to select a value between 2GB (2047 MB) and 4GB (4095 MB). You will now create a system (C:) partition. Move the highlight to "Unpartitioned Space" with the arrow keys. Press "C" to create a new partition and then enter the partition size (a number between 2047 and 4095, consult your system administrator if you need help).

- If your disk is 8GB or smaller, you can create a D: partition here. If your disk is greater than 8GB in size, the NT setup program can't access all of it, so it's best to wait until later to create the D: partition. To create a D: partition, follow the same steps as for the C: partition, with the exception that the size will be all the remaining space on the drive (the number is already filled in for you by default). Typical installations might evenly divide a 6 or 8 GB disk between the C and D partitions. If you expect the database to be quite large, then give the D: parition more space than the C partition (but give C at least 2GB!).

- Put the highlight on the C: partition and press Enter to install NT on the C: drive

- The File System selection screen is displayed. The FAT file system is the default, but FAT partitions are limited to a maximum size of 2GB and are not as efficient as the NT File System (NTFS). Select the NTFS file system and press Enter.

- Wait while the system formats the C: drive. Note that the D: partition (if you created one) does NOT get formatted at this step. Formatting a 4GB partition can take a while (10 minutes perhaps).

- The File location screen displays. Press Enter to accept installing NT in the \WINNT folder.

- The Hard disk examination screen displays. Press ESC to skip the exhaustive hard disk check. The exhaustive check physically checks the entire hard disk surface and can take a very long time, so we skip it.

- Wait while setup does a quick disk check.

- Wait while setup copies files to the disk.

- The Setup Successful screen is displayed. Remove the CD from CD-ROM drive.

- Press Enter to restart the computer.

- Wait while the system reboots and prepares to continue the installation. You will be prompted to insert the NT server CD when it is ready to continue. Note that the PC is now using the mouse and is displaying graphical screens.

- Wait while more files are copied.

- The Windows NT Setup Wizard is displayed. Click the Next button to gather information about your computer.

- The Name screen displays. Enter a name and organization in the fields provided, then click on the Next button.

- Enter the Product ID (use the tab key to jump between fields) then click on the Next button. The Product ID code is found on the cover of the NT Server "Getting Started" book with the "Certificate of Authenticity" on it.

- The Licensing Modes screen is displayed. Select "Per Seat", then click on Next.

- The Computer Name screen is displayed. Enter the name of the computer. Note the name must be unique to avoid confusion within your network domain. Then click on Next. Contact your network administrator if you need help selecting a Name.

- The Server Type screen is displayed. Select the "Stand Alone Server" option, then click on Next.

- The Administrator Account screen is displayed. Enter the Administrator Account password (and the confirmation to ensure you typed it correctly), then click on Next.

- The Emergency Repair Disk screen is displayed. Select "No", since one will be created later (once the the system is completely configured). Click on Next to continue.

- The Select Components screen is displayed. In general, we try not load any "fluff" on the server, so we will limit the amount of extra software we install.

    - De-select all the options (no check mark) except for the "Accessories" and "Communications" options.

    - Highlight the Accessories option and click the Details button. Select all the components listed. Select the OK button.

    - Highlight the Communications option. Click on the Details button. Deselect the Chat and PhoneDialer options. Hyperterminal should be the only component selected. Click on the OK button.

  - Click on the Next button on the Select Components screen to continue.

- The Windows NT Setup screen displays again, click Next to Install Networking.

- The Networking screen is displayed. The "will participate" and "wired to the network" options should be checked by default, click Next to continue.

- The Microsoft Internet Info Server (IIS) screen is displayed. The "Install" MS IIS box is checked by default. Uncheck the box since we will be installing a newer version of IIS later in the procedure. Click Next to continue.

- The Network Adapter screen is displayed. Click on the "Start Search" to look for network cards. Depending on the type of PC you have, the search may or may not find the network interface in your computer. The procedure assumes the search does NOT find the network card in the PC or built into the motherboard.

- Click on the "Select from list" button, then Click on the "Have disk…" button.

- Insert the floppy disk containing the NT drivers for your network card then click on OK to search the disk.

- Select the proper driver from the list, then click OK to copy the driver files to the hard disk.

- The network card you just installed the drivers for should now be listed. Click Next to continue.

- The Network Protocols screen is now displayed. Deselect the "NWLINK IPX/SPX" protocol. Select the NETBui protocol. TCP/IP is required by the PSS system. NETBUI is used by NT to access disk drives on other computers. Click Next to continue.

- The Network Services screen displays. Click on the "Select from list.." button.

- Select the "Microsoft DHCP Server" service from the list, then click OK. The DHCP service is a central resource that gives out IP addresses to other computers on the same network. The terminals do not store IP addresses, so they ask DHCP for an IP address when they first connect to the system.

- Note that DHCP Server is now added to the list of Network Services to install. Click Next.

- Click Next again to install the networking components.

- Wait while files are copied. Click OK on any status boxes that display.

- The TCP/IP Setup dialog box displays and asks "Do you wish to use DHCP?" Click the "No" button. Just to clarify, this dialog box is asking if we want to assign a hard-coded IP

address to the computer or to let the computer use DHCP (running on some other computer) to supply us with an IP address. We do NOT want to use DHCP to get an IP address.

- The TCP/IP Properties dialog is now displayed. Enter the following information in the fields provided, then click OK:
  - IP address
  - Subnet mask
  - Gateway

- The following entry may or may not display: "At least one of the adapter cards has an empty primary WINS address. Do you want to continue?" Answer "Yes."

- The Bindings screen is displayed. Click Next to skip past Bindings screen.

- The Network Start screen is displayed. Click Next to start the network.

- The Computer Name screen is displayed. Click Next to accept the computer name and workgroup selection.

- Click Finish to complete the network setup.

- The Time screen displays. Select the appropriate time zone and system date and time, and then click on Close.

- A Message box is displayed concerning the video adapter. Click OK. Then click OK on the Display Properties dialog box. These selections just confirm that you are using the default 640x480 VGA video driver. We will update the video driver later in this procedure.

- Wait while Setup copies files.

- The Windows NT Successfully Installed message is displayed. Remove CD and floppy, then click "Restart Computer" button.

Congratulations! Windows NT Server is now installed.

4.2 Install the NT 4 Service Pack

NT has bugs, just like any other software. Microsoft releases "service packs" to add new features and fix bugs on your NT installation. This section of the installation guide documents how to install Service Pack 5.

- Log in as Administrator (using the password you defined in section 1).
- Close the "Welcome to Windows NT" window.
- Insert the NT Service Pack 5 CD. Service Pack 5 will autostart in a few seconds.
- Click on the "Install Service Pack 5" link (on the left).
- The screen jumps to the Service Pack 5 options. Click on the "Install Service Pack 5 for Intel-based processors" link.
- A message box asks how to open the file. Click on "Open" to run the setup program.
- The License window displays. Select the "Accept License " option, deselect the "Backup files" option, then click on "Install" to continue.
- Wait while files are copied.
- Click on the "Restart" button to reboot the PC.

4.3 Install Internet Explorer 4.01 Service Pack 2

The PSS system uses the Internet Explorer web browser by default. Installing IE4 also provides some useful desktop updates. We will upgrade our IE4 installation to IE5 later in the procedure (primarily to install the Task Scheduler that comes with IE5), but we install IE4 here to get the desktop update and other Windows enhancements.

- Log in as Administrator.
- The "Welcome to Window NT" Screen is displayed. Uncheck the "Show Welcome Screen Next Time" option, then click "Close."
- Open, then close, the CD drive to restart the Service Pack 5 installation program.
- Click on the "Internet Explorer 4.01 Service Pack 2" link on the left (scroll down if necessary).
- Click on the "Install IE4.01 SP2 for Intel-based Systems" link.
- Click Open on the security message.
- The Internet Explorer 4.01 Service Pack 2 screen is displayed. Click Next.
- The License screen is displayed. Select the "I accept" option, then click Next.
- The Installation Option screen is displayed. Click Next to accept a "Standard Installation".
- The Windows Desktop Update screen is displayed. Click Next to install the desktop update.
- The Active Channel Selection screen displays. Click "Next".
- The Destination folder screen displays. Click "Next" to accept the default destination folder.
- Wait while files are copied.
- Click OK to complete the install.
- Wait while the system is configured.
- Click OK to restart the PC.
- Wait while the PC reboots.
- Log in as Administrator again.

- The IE4.01 setup program automatically starts when you log in and finishes configuring the system.

- Wait for the desktop to appear.

- Close the Active Desktop Window (click the “x”)

- Click on Start button, then Settings, then Active Desktop and uncheck the “View as Web Page” option.

4.4 NT Option Pack Install

The NT Option Pack installs software that is not part of the standard NT installation. Our purpose for installing the Option Pack is to install the Microsoft Internet Information Server (IIS). IIS is the web server used by PSS for the Service Terminal screens.

- Insert the Option Pack CD, click Install
- Click on Step 5, Install the Windows NT Option Pack.
- Select the "Run this program from current location" option, then click the OK button.
- Click "Yes" on the security warning screen.
- The message "Not tested with Service Pack 4 or greater. Do you wish to proceed?" message displays. Click Yes.
- The NT Option Pack Setup screen is displayed. Click Next to continue.
- The License Agreement displays. Click on the "Accept" button.
- The Options screen is displayed. Click on the "Typical" button.
- Click on the Next button to confirm the default file locations.
- Click on the Next button to confirm the default "mailroot" location.
- Wait while files are installed.
- Click "Finish" to complete the setup.
- Click "Yes" to restart the computer.

4.5 Install the Video Driver

The normal NT installation uses a generic video driver. This section leads you through the steps to installing a video driver for the specific video card in your computer. Installing the correct video driver for your hardware allows the system to run at higher resolutions, refresh rates and numbers of colors. You will need the floppy disk containing the video drivers in this section.

- Right Click on the desktop and select the Properties option from the menu.
- The Display Properties window is displayed. Select the Settings tab. Click on the "Display Type" button.
- Click the "Change…" button.
- Click on the "Have Disk…" button.
- Insert the floppy disk containing the NT video drivers. Click OK.
- Select the appropriate driver from the list, then click OK.
- A third party driver message is displayed. Select the "Yes" button.
- Wait while files are copied from the floppy disk to the hard drive.
- The "Successful Installation" message is displayed. Click OK.
- Select the Close button on Display Type window.
- Select the Close button on Display Properties window.
- Remove the floppy disk from the drive.
- Click "Yes" to reboot the computer.
- Wait while the computer reboots.
- Log in as Administrator.
- A message about a new graphics driver is displayed. Click OK.
- Set the Desktop Area to 1024 x 768 (suggested value)
- Set Color Palette to 65536 colors (suggested value)

- Set Refresh Frequency to 85Hz (or the highest refresh rate offered)

- Click the “Test” button, then the OK button to view the test pattern.

- The test ends after 15 seconds. Click “Yes” if you saw the test patten properly.

- Click OK to exit the Display Properties window.

- Click on the Start button, then Control Panel, then System, then Performance tab. Set the Virtual Memory min and max sizes to twice the RAM installed in the computer, then click the “Set” button. Click OK. Select Start Up/Shutdown tab. Change time in “Show list for…” option to 3 seconds. Click OK.

4.6 Create and Format the Database partition

You may remember the discussion about the creation of a D: partition on the hard drive from the early steps of this procedure. If you created a D: partition then, you may skip to the Format steps. If you did not create a D: partition earlier, we will create (and then format) one here. The D: partition is where we store all the database data files. Run the Disk Administrator utility to create and format the D: partition:

- Click on the Start button, then Programs, then Admin Tools, then Disk Administrator
- Click OK to update the configuration

Creating the D: partition

<< TO DO: add steps on creating a Logical drive in an Extended partition. Include info about possibly needing to move the CD-ROM drive letter to E first, so that you can make the new partition the D: drive>>

Formatting the D: partition

- Click on the D: drive to select it (make it the active partition)
- Right click on then D: drive, then select “Format”
- Change the file system to NTFS (NTFS is more efficient than FAT)
- Change the Volume label to “Database”
- Select the “Quick Format” option
- Click on the “Start” button
- A warning message about losing data displays. Click OK.
- A “Format complete” message displays. Click OK.
- Click on “Close” to exit the Format window.
- Close “Disk Admin” window (click on the “X” in the upper right corner of the screen).

4.7 Desktop Cleanup (optional)

This section includes OPTIONAL changes to the desktop environment. We've found that these changes make life a little easier. The intent is clean up the desktop and to set up the Explorer window to show the data we want to see.

- Delete the Outlook Express, My Briefcase, Internet Explorer, and Inbox Shortcuts by right-clicking on the desktop icon and then selecting the Delete option.

- Right click on the "Recycle Bin" icon. Select Properties. Select the "Do not move files… " option. Click OK. This step basically turns off the Recycle Bin and causes all files to be deleted immediately. We find ourselves creating (and later deleting) large log files. If the Recycle Bin is active, you won't actually delete the file (and free up the disk space) until you empty the Recycle Bin. So we turn it Off.

- Right click on the Recycle Bin icon. Select the "Empty Recycle Bin" option, then click "Yes". This step cleans out any files that were previously deleted and moved to the Recycle Bin.

- Right click on the desktop and select the Arrange Icons submenu. Select the Auto Arrange option. This step makes the icons arrange themselves automatically (top to bottom, left to right). You may want to skip this step if you prefer to group desktop shortcuts all over the place.

- Click the Start button. Select Settings, then Taskbar, then Small Icons, then OK. This step keeps the Start menu smaller.

- Click the Start button, then Programs, then Windows NT Explorer. Select View Menu, Folder Options, View Tab. These changes maximize the amount of information displayed in the Explorer window.
  - Select Display full path in the title bar.
  - Select "Show Attributes" in Detail View.
  - Select "Show Map Network Drive" button in toolbar.
  - Uncheck Hide file extensions for known file types.
  - Click on "Show All Files"

4.8 Install Microsoft SQL Server 7

This section installs Microsoft SQL Server 7, the database application used to store PSS data. This section only installs the software, it does not configure SS7, or create the PSS database. Those steps are in the next section.

- Insert the MS SQL Server 7 (Standard Edition, not Enterprise) CD. The setup program will autostart.
- Click on the Install SQL Server 7.0 Components link.
- Click on the "Database Server- Standard Edition" link.
- Click on Next to accept "Local Install"
- The Welcome screen is displayed. Click on Next to continue.
- The License screen displays. Click on Yes to accept.
- Click on Next to accept the default Name and Company.
- Enter CD key at window
- The Setup Type screen is displayed. Select "Typical" install (default). Click "Browse" button for Data files. We want to put the SS7 data files on a different disk than the application itself. Change C:\ to D:\, then click OK. Click on Next to continue.
- The Services Accounts window displays. Select "Use the Local System Account," then click on Next.
- Select Next (again) to continue.
- The Choose Licensing Mode screen displays. Select "Per Seat" then "Continue"
- The Licensing screen displays. Check the "I agree that.." box, then OK.
- Click on "Continue" again (screen change might be sluggish)
- Wait while files are installed (several minutes)
- The Setup complete screen is displayed. Click Finish.
- Click "Exit" to exit SQL Server 7 install program.

- Reboot the PC manually. (Start | Shutdown | Restart)

4.9 Create and Configure the PSS Database

The PSS database is stored in a few large files. This section creates the (empty) PSS data files. We will still have to create database tables and populate them (later in the procedure). After creating the PSS database, we will configure SQL Server and set up the automated database maintenance tasks.

4.9.1 Create the (blank) PSS Database

This step will create the initial storage for the PSS database.

- Select Start button, then Programs, then Microsoft SQL Server 7.0 Enterprise Manager.
- Expand the "MS SQL Server" branch.
- Expand the "SQL Server Group" branch.
- Expand the <*local machine name*> branch.
- Under the <*local machine name*> branch, right click on "Databases." Select "New Database."
- Enter "Name" as "PSS". WARNING: If you enter a different name, various PSS database scripts will no longer work. It is strongly recommended that you use the default database name of "PSS". Use something else at your own risk!
- Set the initial database size to 512MB.
- Click on the Transaction Log tab.
- Set the initial size transaction log to 512MB.
- Click OK to create and prepare the database
- Wait while the database is created (about 5 minutes)

4.9.2 Configure SQL Server

This step will configure SQL Server to function properly, and allow sufficient security access to the database.

- Expand the “MS SQL Server” branch.
- Expand the “SQL Server Group” branch.
- Expand the <*local machine name*> branch.
    - Right-click on the local machine name. Select “Properties”
        - In the General tab, at the bottom, make sure that in the AutoStart policies, both ‘Autostart SQL Server’ and ‘Autostart SQL Server Agent’ are checked.
        - Press OK.
- Expand the security branch
- Right click on “Logins” and select “New Login”
- Select SQL Server authentication
- In the “Name” field, type: IUSR_<machine name>
- Click on Database Access tab
- Click on Pss database “Permit” column
- Select db_datareader and db_datawriter
- Click on OK
- Under the local machine name icon, drop down and expand ‘Management’ and then expand ‘SQL Server Agent’.
- On the SQL Server Agent icon, if a green arrow does not appear, right-click on the SQL Server Agent icon, and select Start.

4.9.3 Configure the Clear Transaction Log job

This step will schedule a clearing of the SQL Server Transaction Log, which provides temporary space for all data modification activities in the database. The Transaction Log must be cleared occasionally to remove old inactive database transaction entries.

- Under SQL Server Agent, click on ‘Jobs’.
    - In the right pane, right-click, and select ‘New Job...’
        - General tab:
            - In Name, enter 'Clear Transaction Log'.
            - Under Category, select 'Database Maintenance'.
        - Steps tab:
            - Click New...
            - In Step name, enter 'Clear Transaction Log'.
            - Under Database, select the name of the PSS database.
            - In Command, enter 'BACKUP TRANSACTION <dbname> WITH TRUNCATE_ONLY, where <dbname> is the name selected under the Database field.
            - Press OK.
        - Schedules tab:
            - Press New Schedule...
            - In Name, enter 'Every 2 hours' (or if you decide to use a different recurrence, a similarly descriptive name)
            - Make sure Recurring is selected, and press Change...
                - In Occurs, press Daily.
                - In Daily Frequency, select Occurs Every, and change the value to the right to 2.

- In Starting At, change the time to 1:55am (or 5 minutes before backups will be scheduled to start below).

- Press OK.

- Press OK.

- Press OK.

4.9.4 Configure the Extensive Database Check and Backup job

This step will configure a database backup, and an extensive check of the integrity of all of the structures in the database, including tables and table indexes, which will run once a week. The integrity check takes a relatively long time because the structure of all of the table indexes must be verified, and because of this, this activity must be scheduled for the time of the week with the least shopping activity.

- Under 'Management', click on 'Database Maintenance Plans'.

- In the right pane, right-click, and select 'New Maintenance Plan...', and the Database Maintenance Plan Wizard should appear. Click "Next."

  - This maintenance plan will include an exhaustive optimization and validation of the database, as well as a backup. It should run once a week when the system is unused, or lightly used. The default schedule for these activities is Sunday in the early morning hours. You should change this schedule only if necessary.

  - On the Select Databases page, make sure 'These databases' is selected, and check the box next to the PSS database name. Click 'Next'.

  - On the 'Update Data Optimization Information' page, click on 'Reorganize data and index pages', and click on 'Remove unused space from database files'.

    - This operation may take some time, and slow down the system, so if you have a reason to use a different schedule, then next to 'Schedule' press the 'Change' button, and make the appropriate changes.

    - Click 'Next'.

  - On the 'Database Integrity Check' page, click on 'Check database integrity'. Click on 'Perform these tests before doing backups'.

    - If you have a reason to use a different schedule, you can change it by pressing the 'Change' button next to 'Schedule'. Make any changes, and press 'OK".

  - Click ‘Next’.

- On the ‘Specify the Database Backup Plan’ page, make sure the ‘Back up the database’ and ‘Verify the integrity’ options are selected, and under the ‘Location to store’, that ‘Disk’ is selected.

  - If you have a reason to use a different schedule, you can change it by pressing the ‘Change’ button next to ‘Schedule’. Make any changes, and press ‘OK”.

  - Click ‘Next’.

- On the ‘Specify Backup Disk Directory’ page, make sure ‘Use this directory’ is selected, and enter the location where your backups should go (usually a different physical drive than the drive on which the database resides). The default directory is the ‘Backup’ directory under your SQL Server 7 installation directory, such as C:\MSSQL7\BACKUP. In any case, you should make sure that the specified directory exists in Windows Explorer, and create it if it does not (using File, New, Folder). Through normal operation, the database, and therefore the backups, can eventually grow to be very large, so make sure the space available will be sufficient.

  - Click on ‘Remove files older than’, and change the data to the right to ‘2’ ‘Days’. This value can be varied based on the amount of space available, and the number of days of backups you wish to keep.

  - Click ‘Next’.

- When the ‘Specify the Transaction Log Backup Plan’ page appears, click ‘Next’.

- When the ‘Reports to Generate’ page appears, click ‘Next’.

- When the ‘Maintenance History’ page appears, click ‘Next’.

- When the ‘Completing the Database Maintenance Plan Wizard’ page appears, select the entire contents of the ‘Plan Name’ field, and type over it: ‘DB Extensive Check and Backup’. This may not paint correctly, but your changes should appear in the list once you are finished with this step. Click ‘Finish’.

- Click OK.

4.9.5 Configure the Nightly Database Check and Backup job

This step will configure a database backup and a brief check of the integrity of all of the tables in the database, which will run every day of the week, except for the night on which the extensive check and backup is run. This activity should be scheduled for the time of the day with the least shopping activity.

- Once again, in the right pane, right-click, and select 'New Maintenance Plan...', and the Database Maintenance Plan Wizard should appear. Click 'Next'.

  - This maintenance plan will include a quick validation of the database, and a backup. It should run once a night (except for the night on which the exhaustive checks are scheduled) when the system is unused, or lightly used. The default schedule for these activities is each day in the early morning hours. You should change this schedule only if necessary.

  - On the Select Databases page, make sure 'These databases' is selected, and check the box next to the PSS database name. Click 'Next'.

  - When the 'Update Data Optimization Information' page appears, click 'Next'.

  - On the 'Database Integrity Check' page, click on 'Check database integrity', and then click on 'Exclude indexes'. Click on 'Perform these tests before doing backups'. Next to 'Schedule' press the 'Change' button.

    - Unless you have a reason to use a different schedule, make sure 'Weekly' is selected, and to the right, check every day except for the day on which the exhaustive backup is scheduled. If the default, Sunday, is used for exhaustive backups, only every other day from Monday to Saturday should be checked. Press "OK".

    - Click 'Next'.

  - On the 'Specify the Database Backup Plan' page, make sure the 'Back up the database' and 'Verify the integrity' options are selected, and under the 'Location to store', that 'Disk' is selected. Next to 'Schedule' press the 'Change' button.

    - Unless you have a reason to use a different schedule, make sure 'Weekly' is selected, and to the right, check every day except for the day on which the exhaustive backup is scheduled. If the default, Sunday, is used for exhaustive backups, only every other day from Monday to Saturday should be checked. Press 'OK".

    - Click 'Next'.

- On the ‘Specify Backup Disk Directory’ page, make sure ‘Use this directory’ is selected, and enter the location where your backups should go (usually a different physical drive than the drive on which the database resides). The default directory is the ‘Backup’ directory under your SQL Server 7 installation directory, such as C:\MSSQL7\BACKUP. In any case, you should make sure that the specified directory exists in Windows Explorer, and create it if it does not (using File, New, Folder). Through normal operation, the database, and therefore the backups, can eventually grow to be very large, so make sure the space available will be sufficient.

    - Click on ‘Remove files older than’, and change the data to the right to ‘2’ ‘Days’. This value can be varied based on the amount of space available, and the number of days of backups you wish to keep.

    - Click ‘Next’.

- When the ‘Specify the Transaction Log Backup Plan’ page appears, click ‘Next’.

- When the ‘Reports to Generate’ page appears, click ‘Next’.

- When the ‘Maintenance History’ page appears, click ‘Next’.

- When the ‘Completing the Database Maintenance Plan Wizard’ page appears, select the entire contents of the ‘Plan Name’ field, and type over it: ‘DB Backup’. This may not paint correctly, but your changes should appear in the list once you are finished with this step. Click ‘Finish’.
-

- Click OK.

- Exit Enterprise Manager.

4.10 Install Microsoft Access

There are times that you may want direct access to the WaveWorks database. You do not need to install the whole MS Office Suite or even any of the Access extras, just Access. This procedure assumes you are using the Office 97 Professional CD.

- Insert the Microsoft Office Professional CD.
- Click on Install MS Office.
- The Welcome screen is displayed. Click on the Continue button.
- Enter Name and Organization. Click OK.
- The default install folder is displayed. Click OK to accept the default location.
- The installation type window is displayed. Select “Custom”.
- Deselect everything except Microsoft Access, the click Continue to install
- A success message is displayed. Click OK to complete the setup.
- Click on the “X” to exit setup.

4.11 Internet Explorer 5.0 with Task Scheduler install

This section installs the latest version of the Internet Explorer browser. The main reason we install IE5 is to also install the new Task Schedule program. PSS uses the Task Scheduler to periodically trigger transactions. We will configure the periodic tasks later in this procedure.

- Insert the PSS CD.
- Open an NT Explorer window.
- Double click on the “symie5.exe” icon in the root folder of the CD.
- Wait while lots of files are copied to the disk (takes a while). The PC will automatically reboot when done.

- Log on as Administrator.
- The final setup of IE5 will continue. The normal desktop will be displayed when setup is complete.

4.12 Install WaveWorks

ObjectWave has been renamed "WaveWorks." The name change is only partially complete, so you may still see references to "Objectwave" on the system. WaveWorks is the foundation that the PSS system is built on.

- Insert the PSS CD (if it's not still in the CD-ROM drive from the previous step)
- Run the WaveWorksSetup.exe file via Explorer to install WaveWorks.
- The Welcome Screen is displayed. Click on the "Install" button.
- The Components Screen is displayed. Click on the "Next" button.
- The Destination Directory is displayed. Click Next to accept the default location. NOTE: You will break several PSS scripts and will have to perform extra configuration steps to correct these problems, if you install to a non-default location.
- Click Next to accept the default network parameters.
- Uncheck the "Autostart Services" option. You will need to make the WaveWorks services autostart before placing the system into production, but it MUST be unchecked until you have completed the entire PSS installation and verified that things are working properly. Click Next to continue.
- You may accept the default value of 35 days to retain logs, but if you are installing on a test/lab system, then perhaps you'll want to change this value to 7 days to avoid using large amounts of disk space. Click Next to continue.
- Enter Domain (If part of a larger network). Select Next.
- The Install screen is displayed. Click Install, then OK.
- Wait while files are copied.
- The install is completed.

- Click Finish

- Reboot the PC to make new environment variables and registry changes take effect

4.13 Install PSS Runtime System

This section installs the PSS software. It is strongly recommended that you use the default folder for the files. This section installs the generic/base system.

- Run the PssWaveworksSetup.exe file to install the PSS system.

- PSS requires several Microsoft and WaveWorks components. If you have skipped any steps in this installation guide, then you will be notified by a dialog box of any missing components which may be required. The installation process may stop if a missing component is required. Please note the missing component name and follow the instructions in this document for installing that component.

- The Welcome screen displays. Click Next

- The PSS Components dialog displays. Choose the appropriate type of handheld terminal, entrance unit, printer and POS (Point of Sale) system interface which you will be using. Click Next.

- Select Destination Directory dialog displays. Click Install. Do NOT change the default installation directory for PSS. The current installation script will not be able to successfully complete the installation if this directory is changed.

- Wait while files install.

- Several DOS windows will be displayed as various batch file scripts are run. Most of these scripts are updating various WaveWorks and PSS database tables. Be sure to read the messages in the windows. If you see any error messages, write them down. After each script completes, the message “Press any key to continue” displays. Press a key to allow the install procedure to continue.

- The supplied PSS COM objects are registered at the end of the installation process. For a first time installation, you may get errors indicating that some Microsoft dll files are missing. These files have been delivered, but are not yet active, so the system reports errors. In subsequent installations, these messages should not appear.

- The Limited License Notice screen displays. Read its instructions and then press OK.

- The Installation Completed screen displays. Press the Finish button. The installation process is now completed

- Reboot the PC to make PSS environment variables and registry changes take effect.

4.14 Install Waveworks Development Studio

This section installs the Waveworks Development Studio system used for modifying the scripts that drive the handheld terminal applications and that link terminal actions to WaveWorks transactions. It is strongly recommended that you use the default folder for the files.

- Run the ScreenMakerSetup.exe file on the root directory of the CD..

- The Welcome screen displays. Click Next if you have installed the base WaveWorks system, or click Cancel and install that component first.

- The Select Components dialog displays. It defaults selection to the most likely components to be used. If you will be modifying handheld terminal displays, the ScreenMaker and Hardlock components must be checked. If you will be modifying Pss User Exit routines, the C++ Development Toolkit must be selected.

- The Select Destination directory dialog displays. Click Next.

- The Ready to Install dialog displays. If you are sure of your selections, click Next.

- The installation progress bar now displays showing the progress of the installation.

- If all goes well, the Installation Completed dialog box displays. Click Finish.

This installation delivers the ScreenMaker application which is used for development of applications that run on Step compliant terminal devices. It also delivers WaveWorks libraries and header files and sets up environment variables needed to compile and link C or C++ WaveWorks transaction components.

4.15 Install PSS Development System

This section installs the PSS Development system software required for modifying handheld terminal displays and providing other client-specific customizations. It is strongly recommended that you use the default folder for the files.

- Run the PssDeveloperSetup.exe file.

- The “PSS Development System Installation” screen displays identifying the prerequisite WaveWorks and PSS components. Click Next if you have installed these components, or click Cancel and install those components first.

- The PSS Development System requires several Microsoft and WaveWorks components. If you have skipped any steps in this integration guide, then you will be notified by a dialog box of any missing components which may be required. The installation process may stop if a missing component is required. Please note the missing component name and follow the instructions in this document for installing that component.

- The “Install New User Exit Sources” dialog displays. It defaults to installing the sources. If this is a first time installation you should just click the Next button. If this is not the first time installation and you have modified the User Exit sources, you may choose to skip this step. Before choosing, please read the release notes to see if any existing User Exit call formats have changed or if any new User Exits are supplied in this release before you decide on whether or not to install the User Exit sources. If you choose to install the User Exits, the files in the User Exit directory will be moved to a backup directory before the new files are installed. After making your selection, click Next.

- “Select Destination Directory” dialog displays. Click Install.

- Wait while files install.

- The “Installation Completed” dialog displays. The installation process has finished. Click Finish.

This installation delivers PSS libraries and header files and sets up environment variables needed to compile and link the User Exit sources into a dll. It also contains a project file for version 6 of Microsoft Developer Studio. If you are using that development platform, you may use that file to build the dll. If you are using another development platform, you will need to develop your own make file or equivalent project file. Once you have built a new dll file, copy it to the Pss\Bin directory to make it part of the PSS runtime system.

4.16 TFTP Configuration

TFTP (Trivial File Transfer Protocol) is the mechanism that is used to transfer files between the host and the terminal when the terminal boots up and connects to the host. This section explains how to install the Symbol TFTP service.

- Click on Start button, then Programs, then Command Prompt to open a command window.

- Type cd c:\tftpserver to change to the "TftpServer" folder.

- Type tftpsvr install c:\tftpserver\tftp.ini to install the TFTP server as a service.

- Type Exit to close command window.

4.16.1 Create STEP hex images for the terminals to download

This section explains how to create the hex image files that are transferred to the terminal. Asterix2 terminals have a somewhat different procedure than Asterix3 terminals, so follow the steps for the type of terminals you have. The net result of these steps is that new hex image and configuration files will be created and copied under the C:\TftpBoot folder for use by the terminals.

Asterix2 Terminals

- Go to Start | Programs | Command Prompt to open a command prompt window.
- Change to the C:\Asterix2Step\Control folder
- Edit the “hosts.x” file. The file should have a single line with the IP address and hostname and port id of the DHCP server (typically the target computer). Save the file and exit.
- Change to the Asterix2Step\Hex1 folder
- Edit the “Net.cfg” file. Change the "net_id" to match the net ID of the access point(s) the PSS system is using. Close and save the file.
- Change to the Asterix2 folder.
- Run MakeStep.bat to create/copy terminal files to the TftpBoot folder
- Exit the command window.

Asterix3 Terminals

- Go to Start | Programs | Command Prompt to open a command prompt window.
- Change to the C:\Asterix3Step\Control folder
- Edit the “hosts.x” file. The file should have a single line with the IP address and hostname and port id of the DHCP server (typically the target computer). Save the file and exit.
- Change to the Asterix3 folder.
- Run SetupStp.bat to create/copy terminal files to the TftpBoot folder
- Exit the command window.

4.17 DHCP Configuration

DHCP is the service that gives out IP addresses to Asterix terminals when they boot up. The DHCP service is allocated a range of IP addresses that it can give out. This section describes how to set up that range of addresses and make them active.

- Run the "DHCP Manager" program by selecting Start button, then Programs, then Administrative Tools.
- Select "Local Machine" (double click)
- Run Scope | Create... to create the range of IP addresses to manage.
- Enter the start and end addresses, and subnet mask.
- Change Lease Duration to Unlimited. In Name field at bottom of screen, enter description of terminals. Select "Yes" when asked to activate the addresses.
- Acknowledge status message.
- Close DHCP window (x at top).

4.18 Create an ODBC Data Source

This section creates the ODBC data source that is used by the PSS software to access the database. It is strongly recommended that you use the default name “PSS” for the data source. Using a different name will require you to manually edit the PSS.INI file. The PSS software uses the data source name in the PSS.INI file and if a matching data source is not found, then nothing will work.

- Run the ODBC32 control panel applet (Select Start button, then Settings, then Control Panel, then ODBC)
- Click on the "System DSN" tab
- Click on the "Add..." button
- Select the SQL Server driver, then click the Finish button.
- Enter the following DSN information, then click on the Next button:
    - Name: PSS
    - Description: PSS Data
    - Server: (local)
- Enter the following information on user IDs, then click the Next button:
    - Select the "With SQL Server authentication..." radio button
    - Change the Login ID: to "sa". Leave the Password blank. Click on Next.
- The Create New Data Source window is displayed.
- Change the default database to “PSS”, then click the Next button.
- Select the "Use regional settings" checkbox.
- Click the Finish button
- Click on the "Test Data Source..." button to verify everything is OK.
- Click on the OK button to return to the System DSN tab.
- Click OK to close ODBC Data Source Administration.

4.19 Load Initial Data

This step will populate the PSS database with the initial data needed for the system to run. The PSS system is very much a data-driven system. It is CRITICAL that these scripts be run correctly if you want the system to run properly. You can run a batch file that will run all the scripts for you (the "Automatic" method), but it requires that the database is named "PSS". If you chose to use a different name for the database, you will have to run each script manually using the Query Analyzer program or edit the BAT file to use the database name you used. The Automatic method is the preferred method.

Automatic Method

- Open a command window (Start | Programs | Command Prompt)
- Change to the Pss\Database folder
- Run the Pss_Create_And_Init.bat file to create the PSS tables and fill them with data.
- Exit the command window

Manual Method

- Run the Query Analyzer program (Start | Programs | Microsoft SQL Server 7.0|Query Analyzer)
- Connect to SQLServer (click on the "…" button).
- Select the "Start SQLServer if stopped" button, then click OK.
- Select the PSS database from the dropdown list.
- Run the following SQL files, IN ORDER, by opening them (File | Open), then executing them by clicking on the green triangle. Pay attention to the status messages at the bottom of the window. The message should read "Query completed without errors" if the script was successfully executed. The following SQL files are found in the C:\Pss\Database folder (by default):
    - Pss_Drop_Db.sql (deletes all existing PSS tables, so backup any existing data first)
    - Pss_Create_Db.sql (creates empty database tables)
    - Pss_Init_Data.sql (loads default data)
    - PosInterface_Dbinit.sql (loads POS interface settings)
    - PosInterface_Text_Engus.sql (loads POS text strings in US English, the default language)

- Ums_System_Settings.sql (loads Unit Management settings)
- Ums_Text.sql (loads Unit Management settings in US English)

- Exit Query Analyzer.

4.20 MS IIS Configuration

This step will configure the web server to recognize the PSS Service Terminal as a valid Active Server Page application, and will allow access to the system from a web browser.

- Select Start button, then Programs, then Win NT Option Pack, then Microsoft Internet Information Server, then Internet Service Manager.

- Expand Internet Information Server.

- Expand <your machine name>

- Expand default Web Site

- Right click on PSS Interface - Properties. In the Virtual Directory tab, click on Create button.

- Click OK.

- Exit

- Save console setting message displays. Click No.

4.21 Obtaining Access to PSS System Administration Screens

This step will configure Internet Explorer for easy access to the Portable Shopping System Service Terminal.

- Open Internet Explorer (double-click on the icon on the desktop)

- If the Internet Explorer connection Wizard displays, select the "connect through Local Area Network" option.

- When the browser window displays, select Tools, Internet Options. In the 'General' tab, in the 'Home page' box, next to 'Address:' type in the address: http://localhost/pssinterface/index.asp, and press OK. Press the 'Home' button, which has an icon that looks like a house.

- The System Administration Main Menu screen displays. Login with the default username ADMIN, password ADMIN.

- You can then access the System User account screen and setup user accounts specific to your store, as described in the PSS User Reference Guide.

4.22 Configure Task Scheduler

This step creates processes that must run automatically at various intervals for the proper function of the Portable Shopping System. If these tasks are not configured correctly, certain features of the system, such as timed messaging, will not work, and the system will eventually cease functioning.

- Double-click "My Computer" on the Windows Desktop. Once the "My Computer" window appears, locate and double-click "Scheduled Tasks".

- There should be several tasks listed in the window ("Add Scheduled Task" is not a task, but will allow you to add tasks to the scheduler, if desired).

- For each task:

    - Double-click the task, and the task dialog should appear. Make sure the Task tab is selected.

    - In the "Run:" and "Start in:" fields, scroll across and make sure that all of the path names that appear point to the correct PSS installation directory (C:\Pss).

    - In the "Run as:" field, enter the local Administrator account (e.g. MyMachineName\Administrator). Click the Set Password button, enter the Administrator password in each of the two fields, and press OK.

    - Press OK on the task dialog.

    - To verify that the scheduled task can run correctly, right-click on the task, and select Run. Scroll to the right in the Scheduled Tasks window, and make sure that under the Last Run Time column, the current time appears, and that under the Status column, no error messages appear. The task can then run successfully as scheduled.

- Once the procedure above has been run on all scheduled tasks, close the window.

4.23 Configure Unit Management

This step establishes the configuration of the hardware needed for PSS. The exact number of entries is specific for each installation, so you'll need to know what your hardware set up is. Use the Service Terminal screens to add entries or just directly type them into the database using the SQL Server Enterprise Manager. If the hardware configuration is a "standard" setup, then perhaps it would be wise to have an SQL file that can be run from the Query Profiler instead of manually entering the data at each store. Follow the order of data entry as shown. There are relationships between the tables that require this order of creation:

- Create Power Supplies
    - Each power supply can handle about a dozen devices. The Unit Management logic uses the relationship between cradles and power supplies to prevent unlocking more than one cradle per power supply. Enter one row for each power supply in the PSS system.
- Create Entrances
    - Create a row in the UMS_Entrance table for each entry station. The Entrance ID is just a sequential number that is used to identify the Entry Station once it has logged in. The Entry Station logs in using its MAC address (RF Entry stations) or its service name (Ex. PssEntrance01) for serial entry stations. The MAC address (or service name) needs to be entered in the Hardware ID field. If you have multiple entrances you can set the backup entrance. The system will automatically failover to the backup entrance when the entrance is set to Out of Service.
- Create Printers
    - Create a row in the UMS_Printer table for each printer in the system. The Printer ID is just a sequential number that is used to identify the Printer once it has logged in. The Printer logs in using its MAC address (RF printers) or its service name (ex. PssPrinter01) for serial printers. The MAC address (or service name) needs to be entered in the Hardware ID field. The system will automatically failover to the backup printer when the printer is set to Out of Service.
- Create Dispensers
    - Create a row in the UMS_Dispenser table for each dispenser in the system. Dispensers are simply a logical group of cradles that are (usually) physically close to each other. A dispenser is NOT a piece of hardware, although all the cradles mounted on a particular piece of furniture may be called a dispenser. It is up to the discretion of the integrator on how to best group cradles into dispensers. You

want at least one dispenser per entry station, but you may have multiple dispensers per entry station. Customers are directed to a dispenser, so keep this in mind when deciding how to group cradles into dispensers. Each dispenser is controlled by a single entrance.

- Create Cradles

    - Finally, create a row in the UMS_Cradle table for each cradle in the system. This can be a time consuming process if you have 100 or more terminals and is a fine reason to have a standard SQL script for configuring the UMS tables. Each cradle has a location barcode sticker. Make sure that the cradle ID's you enter match the barcodes exactly (i.e. if the barcode is a "0100", the cradle ID entered must be 0100). The stickers MUST be unique, but they do not need to be sequential. Be careful to assign the correct power supply to each cradle.

Note: you do not need to create rows in the UMS_Terminal table for the hand-held terminals. As terminals login to the system, rows are automatically created.

4.24 Configure Licensing

This step makes the number of licenses bought for specific Microsoft products known to the system, to avoid warnings in the NT Event Log about violations of software licenses. To do this, perform the following:

- Select Start, Programs, Administrative Tools, License Manager. The License Manager window should appear.
- From the License Manager menu, select License, New License.
- Next to Product, select “Microsoft SQL Server 7.0”.
- Next to Quantity, enter the number of licenses you have purchased.
- Press OK.
- Read the license agreement, and click on 'I agree that'.
- Press OK.
- From the menu, select License, New License.
- Next to Product, select “Windows NT Server”.
- Next to Quantity, enter the number of licenses you have purchased.
- Press OK.
- Read the license agreement, and click on 'I agree that'.
- Press OK.
- Close the License Manager.

4.25 Configure the NBQMAIN Service (IBM 4690 POS Only)

If you chose the IBM4690 POS as your Point of Sale system, NBQMAIN service, which is part of IBM's Retail Connectivity Option(RCO), is a critical link between the PSS system and the POS. The NBQMAIN service should be configured to startup automatically when WindowsNT starts up. It is NOT controlled by the WaveWorks System Controller. This section explains how to set up the service to automatically start and how to configure the service.

You will need to make the NBQMAIN service auto-start at bootup. Go to Start→Settings→Control Panel→Services and click on the "nbqmain" entry in the list. Now click on the "Startup…" button (NOT the "Start" button). Select the Automatic startup option and click on the OK button. Close the Services window.

The startup parameters, such as the POS controller name, LANA number and heartbeat interval are contained in the NBQMAIN.CFG file in the PSS\Bin folder. You will need to edit the NBQMAIN.CFG file to set these parameters to match your specific installation. Default values are provided in the file.

The most likely parameter to be modified is the LANA number, which is specified with the –a switch in the NBQMAIN parameters list. This number identifies which netbios port to use for communication to the IBM4690 POS. The correct value for this parameter is determined as follows:

From the Windows NT Start button follow the path Start→Settings→ControlPanel→Network.

Select the Services tab, then double-click the NetBIOS Interface

Note the lana number for the "Nbf" entry for the network card connected to the POS network.

For more information on configuring parameters for the NBQMAIN service, start a DOS command prompt session, set your path to C:\Pss\Bin and type the command:

nbqmain -?

Additional information is also available in the document Retail Connectivity Option Version 2.2 , IBM product reference 5764-054.

4.26 Setup NT System Log

Select Start, Program, Administrative Tools, Event Viewer.

Click on "Log" tab.

Select "Log Settings" from pulldown

Chnage settings for "System"

In the "Event Log Settings" Window, in the event Log Wrapping Area, select "Overwrite events as needed"

Click OK

Click on Log Tab

Select Exit

5. Configuration of the PSS System

5.1 Set Up System Setting Constants

All system configuration data is stored in the database. A System Administration screen is included to provide you with the ability to adjust the configuration data for the system. You can navigate to this screen by selecting System Settings under the System Management heading on the main System Administration screen.

The Name of each configuration item relates this particular configuration item to a subsystem of the PSS System. The Subname provides a descriptive name for this particular item. The Value field displays the current setting for this field. The Description provides all the pertinent information explaining the use and/or settings for this item. The Unit gives the allowable range of values. The Default Value is the value used if no changes have been made to this item. And the Actions field limits the allowable actions for this particular item.

Name – Subname
4POSSinterface – CopyTaxTables
4POSSInterface – NumberPriceBytes
4POSSInterface – POSTransFileVersion
Activity – InactiveTime
DisplaySize – ItemPrice
Display Size – Total Price
ImportItemData – CheckDigitExists
Import Item Data – ValidateCheckDigit
POSInterface – ExcludedDepartments
POSInterface – LanAdapterNumber
POSInterface – Local_TransferArea.
POSInterface – POSControllerName
POSInterface – POSControllerType
POSInterface – ProcessedFileArea
POSInterface – PSSItemBarcodeLength
POSInterface – PSSTransactionType
POSInterface –Remote_TransferArea
POSInterface – Trace_Level
POSInterface – UsePSSPrice
POSItemFile – AddItemChkDigit
POSItemFile – FileFormatVersion
POSItemFile – FileName
POSItemFile – FullItemFileName
POSItemFile – MonitorInterval

Name – Subname
POSItemFile – ProcessTX
POSItemFile – PSSExpansionLength
POSItemFile – RemoveItemChkDigit
POSItemFile – TransferEnabled
POSItemFile – TransferTime
POSItemFile – TransferTX
POSItemFile – TransferType
POSItemFile – UserExpansionLength
POSItemUpdate – MonitorInterval
POSTaxFile – FileFormatVersion
POSTaxFile – FileName
POSTaxFile – MonitorInterval
POSTaxFile – ProcessTX
POSTaxFile – TransferEnabled
POSTaxFile – TransferTime
POSTaxFile – TransferTX
POSTaxFile – TransferType
POSTransFile – AddCustChkdig
POSTransFile – AddItemChkdigit
POSTransFile – FileFormatVersion
POSTransFile – FileName
POSTransFile – MonitorInterva
POSTransFile – ProcessTX
POSTransFile – PSSExpansionLength
POSTransFile – RemoveCustChkdigit
POSTransFile – RemoveItemChkdigit
POSTransFile – TransferEnabled
POSTransFile – TransferTime
POSTransFile – TransferTX
POSTransFile – TransferType
POSTransFile – UserExpansionLength
PSS_Global – Barcode_Type
PSS Globar – Default Currency
PSS_Global – Default_Language
PSS_Global – LoyaltyCardLength
PSS Global – PssVersion
PSS_Global – PSS_Name
PSS Global – Store Name
PSS Global – Store Number
PSS Global – Trace Level
PSS Global – Transaction ID
PSS File Monitor – Report Startup Configuration
PSS File Monitor – Trace Level
PSS TransFile – AddCustCheckDigit
PSSTransFile – AddItemCheckDigit

Name – Subname
PSSTransFile – FileFormatVersion
PSSTransFile - File Name
PSSTransFile – Monitor Interval
PSSTransFile – ProcessEvent
PSSTransFile – PSSExpansionLength
PSSTransFile – Transfer Enabled
PSSTransFile – TransferTime
PSSTransFile – TransferTX
PSSTransFile – TransferType
PSSTransfile – UserExpansionLength
RESCAN – ConsecNExcptBelowLvl
RESCAN – DefaultDiffUnit
RESCAN – DefaultInitLvl
RESCAN – GlobalAllTimeoutl
RESCAN – GlobalNoneTimeout
RESCAN – GlobalRescan
RESCAN – IgnoreScanTooMuch
RESCAN – MaxTripAmt
RESCAN – MaxWeeksWithout
RESCAN – MinItemCnt
RESCAN – MinTripAmt
RESCAN – MinItemCntNoRescan
RESCAN – NewUserNeverNumTrips
RESCAN – NewUserNeverSecond
RESCAN – NewUserRescan
RESCAN – OptionsChangeData
RESCAN – UseConsecNever
RESCAN – UseConsecNeverExcept
RESCAN – UseGlobalAllTimeout
RESCAN – UseGlobalNoneTimeout
RESCAN – Use GlobalRescanOpts
RESCAN – UseMaxTripAmt
RESCAN – UseMaxWeeksWithout
RESCAN – UseMinItemCnt
RESCAN – UseMinItemCntNoRescan
RESCAN – UseMinTripAmt
RESCAN – UseNewUserOpts
SHOPPING TRIP – CreateScanIn
SHOPPING TRIP – FS SUPPORT
SHOPPING TRIP – FX TAXED
SHOPPING TRIP – LOYALTY
SHOPPING TRIP – MAXIMUM ITEMS
SHOPPING TRIP – MAXIMUM VALUE
SHOPPING TRIP – PRINT TICKET
SHOPPING TRIP – ROUNDING METHOD
SHOPPING TRIP – SYSTEM CURRENCY
SHOPPING TRIP – SYSTEM LANGUAGE
SHOPPING TRIP - Trace Level.
SHOPPING TRIP – TRANSACTION BARCODE

Name – Subname
SHOPPING TRIP – TRANSACTION TYPE
SPECIAL BARCODE – CUSTOMERENDOFTRIP
SVCTERMINAl – Card Prefix
SVCTERMINAL – CustomStatusString
SVCTERMINAL – UsePrinters
SVCTERMINAL – UseQuickOrder
TRANSTICKET – DefaultDir
TRANSTICKET –DefaultPrinter
TRANSTICKET – PrinterColumnWidth
TRANSTICKET - Ticket Cut Type
UMS – BadScanFactor
UMS – BaseFactor
UMS – ChargingFactor
UMS – GoodScanFactor
UMS – LaserOnFactor
UMS – MaxBatteryLevel
UMS – MinBatteryLevel
UMS – PopupTimeout
UMS – QueryTimeout
UMS – RadioXmtFactor

5.2 Perform Initial Loading of Customer and Item Data

Both the IBM 4690 POS and the FileBased POS interfaces require the creation of flat ASCII files of information for PSS to process. The formats and default filenames for the Item and Customer files are documented in Appendix D of this document. Ultimately the POS should create both of these file types for PSS to process. An Example Customer file and Item file are provided in the PSS\PosTools directory for reference. See the readme.txt file in the directory for more information on the files provided in that directory.

After other configuration steps have been completed and the PSS System has been started, a set of test Customers and test items can be loaded into the system by placing the example Customer and Item files into the configured Transfer directory(C:\Pss\Transfer). The PssFileMonitor service will detect them and trigger the WaveWorks transaction to process them and load the information into the PSS database.

6. Validation of System Operation

6.1 Testing an Installation

Tests	Notes
On Windows NT Controller: Start ObjectWave System Controller Services or check that they are running.	
Selecting /opening browser software displays PSS Main menu.	
Valid username / password allows login and PSS Main Menu is displayed.	
System Summary screen shows system available and correct hardware status.	
Physically check that terminals are in "ready to shop" state.	
Physically check that Entrance Station displays "Welcome to PSS / insert card."	
Valid shopper card can check out terminal (note location terminal taken from).	
System Admin / Terminals screen shows that location as "empty."	
System Admin / View Current Shoppers screen displays that shopper name / card number.	
Begin Shopping Trip:	
Scanner Welcome screen displayed for *n* seconds.	
Opening messages display if defined.	
Press "+" button and scan a normal item that exists in Item File. Item should be successfully added and the Item Entry Screen displayed. Displays correct item description / quantity / price. Increments total quantity and dollar amount of all items.	UPC #1
View Shopping List Screen.	
Add normal item that exists in Item File (Scan barcode while pressing "+" key).	UPC #2
Add price-embedded item that exists in Item File.	UPC #3
Add normal item that exists in Item File.	UPC #4
View Shopping List Screen.	
On Shopping List Screen, scroll up and down through list.	
Delete normal item just added.	UPC #4
Add Item not in Item File (Exception Item). (Scan barcode while pressing "+" key) Displays "Exception Item" screen for *n* seconds, then Item Entry screen is redisplayed.	UPC #5
Add "restricted" item that exists in Item File.	UPC #7
View Shopping List Screen.	
Add multiple quantities of the same normal item that exists in Item File.	UPC #8
End Shopping Trip; verify on scanner. total value of all items total number of items	
Return scanner to cradle.	
Verify that scanner reads barcode location and that System Admin / Terminal screen updates that terminal status / location.	
If using tickets, verify ticket total matches display on scanner.	
Check System Administration Screens for shopper's Item List and Activity List; verify that lists are correct.	
Check customer trip through POS register.	
Check System Administration Screens that shopper's trip moves from Current to Historical Status.	

6.2 Obtaining System Status Information

6.2.1 Viewing The System Log

The System Log screen allows you to view detailed information on events that have occurred in the system; for example, to determine the cause when the system is down.

From the Main Menu, select System / Settings / Status / System Log.

This screen displays the following information:

- The dates and times that events occurred. The pages are ordered from the most current event to the least current.
- The facility (part of the software) that performs a function. Examples are Unit Management, Transaction Ticket, PSS File Processor, etc.
- Actions performed that are specifically related to the facility, such as Terminal Returned, Print Receipt / Stub, POS transFile, etc.
- Severity code from 1-4 and 10, in ascending order of severity.
- Messages.

This System Log screen has several filter options for limiting the data display as follows:

- Display all events in a particular date / time range.
- Display all events associated with a particular facility within a date / time range.
- Display all events associated with a particular action and date / time range.
- Display all events with a particular severity code and date / time range.
- Display all events with messages containing typed-in words or phrases.
- You can also display data with all of the above filter options or select a date / range and any other one or more of these options.

Note: The data in the log can be limited or expanded by setting the "trace level" of a particular system facility (function) to a particular level. *Trace levels* are set in the PSS Settings screen. If the trace level for the Quick Order function is set to 2, for example, the system will save information at level 2 and above, but will not save errors to the log that are beneath that level. The lower the trace level, the more details on the function are saved in the log.

Trace levels are displayed on the log under the title Severity Level. If problems occur with a facility of the system, you can view the severity level for that facility and decide to lower the trace level so that future events relating to that facility will be saved in the log.

6.2.2 Viewing POS Status

The POS Interface screen displays POS status messages and activities. These can be sorted by item name or start date.

To access the POS Status screen, from the Main Menu, select Hardware / POS Interface. The following information is displayed:

- Item name (i. e., customer file, item file, etc.)
- Instance name
- Detection time
- Start and end time
- Instance value—(online, restarting, open, etc.)
- Instance description—(errors while processing, successfully processed, etc.)

7. Directory Listing of PSS Folders/Files

This section provides a detailed listing and description of the Folders and Files that are created as a result of the PSS installation process.

C:\

Folders	SubFolder	Files	Description
Asterix2Step			This folder contains the software that is loaded on the Asterix 2 hand held scanners and the batch files which place those files into the tftpBoot directory for retrieval by the scanners during initial introduction to the system or during reboot of the terminal.
		Bios236.hex	Bios hex file used for manual download.
		Makebios.bat	Used by makestep.bat to create .bin version of bios.
		Makectrl.bat	Used by makestep.bat to copy files to tftpboot directory.
		Makehex1.bat	Used by makestep.bat to create hex1 hex and .bin files.
		Makehex2.bat	Creates hex2 .hex and .bin files.
		Makestep.bat	Transfers A2 files to the tftpBoot directory.
		Romdisk1.hex	Hex1 .hex file used for manual download.
		Romdisk2.hex	Hex2 .hex file used for manual download.
		Sendbios.bat	Manual download batch file for bios.
		Sendhex1.bat	Manual download batch file for hex1.
		Sendhex2.bat	Manual download batch file for hex2.
	Astdiag		Contains diagnostic files for A2.
	Bldtools		Contains .exes used by batch files.
	Control	Host.x file	Defines which host the terminal connects to.
	Hex1		Contains all files associated .hex1 files.
	Hex2	Net.cfg	Contains access point id number and other files part of .hex2 files.
Asterix3Step			This folder contains the software that is loaded on the Asterix 3 hand held scanners and the batch files which place those files into the tftpBoot directory for retrieval by the scanners during initial introduction to the system or during reboot of the terminal.
		MakeRFLD.bat	Batch file to create .LD file from .hex.
		RFburn.hex	hex file for RF loader (Ver 1.4) (obsolete)
		SendRF.bat	Manual download of RF loader.
		Setupstp.bat	Transfers A3 files to the tftpBoot folder.
	Bldtools		Executables used by batch files.

Folders	SubFolder	Files	Description
	Control	Host.x	Defines which host the terminal connects to.
ObjectWave			Location of all ObjectWave-related architecture files (Refer to WaveWorks product documentation.)
PSS			The Master directory for the PSS software.
		Install.log	Created by installation program to document the placement of files.
		Prs_tags.000 Prs_tags.001	When tickets are printed to serial printers, these files convert the software commands into printer commands.
		Pss.ini	Initialization file which defines the path the database, as well as the login authorization.
	Backup	*varies*	Directory used by the installation process to keep previous versions of files (in the event that regression is needed).
	Bin		
		EhtService.exe	STEP translator for RF entrance units.
		EntryStationService.exe	Controls serial entrance units.
		nbq.bat	4690 POS install script.
		nbqmain.cfg	4690 POS configuration file.
		nbqmain.exe	4690 POS NT service.
		nxg.cfg	Library used by IBM 4690 POSService.
		Nxgco.dll	Library used by IBM 4690 POSService.
		Nxghil.dll	Library used by IBM 4690 POSService.
		Nxgue.dll	Library used by IBM 4690 POSService.
		posapi.dll	Library used by IBM 4690 POSService.
		POSInterFaceCOM.exe	COM object – processes POS item, transaction files.
		PosService.exe	4690 POS – NT service that transfers information between POS and PSS.
		PrinterService.exe	Controls serial ticket printers.
		PssActivityCom.exe	Adds shopping trip activities to activity table.
		PssCustomerMaintCom.exe	COM object – processes POS customer file.
		PSSCustomerRankCom.exe	COM object – performs customer ranking based on amount spent over a specified number of shopping trips.
		PssDBCleanupCom.exe	COM object – performs database data maintenance functions, such as removing old shopping trips, or message log data.
		PssFileMaintCom.exe	COM object – performs file housekeeping functions.
		PSSFileMonitorSvc.exe	File monitor service.
		PssQOBarCodeCom.exe	Processes quick order barcodes.
		PssShoppingTripCom.exe	Performs shopping trip functions.
		PssSpecialBarCodeCom.exe	Performs special barcode lookups.
		PSSSystemMessage Com.exe	COM object – provides PSS System Messaging functionality for sending messages at specified times.

		PssTransactionTicket Com.exe	COM object – provides PSS transaction ticket printing services, in conjunction with the PrinterService.
		PSSUserExits.dll	Stub user exit routines.
		PssUtilsCom.exe	COM object – contains PSS utility functions.
		PssWebUtilityCom.dll	COM object – provides WaveWorks / PSS utility services for the Service Terminal.
		Qexec.exe	"Quietly" executes a command (no command window).
		UnitMgmtCom.exe	Provides unit management business services.
	Database		
		InsertA2EntranceServices.sql	Adds serial entrance services to the WW daemon table.
		InsertA2PrinterServices.sql	Adds serial printer services to the WW daemon table.
		InsertA3EntranceServices.sql	Adds RF entrance service to the WW daemon table.
		InsertAdmProcesses.sql	Adds startup / shutdown entries to the WW process table.
		InsertPssServices.sql	Adds standard PSS services to the WW daemon table.
		InsertServiceTerminalPrinter Service.sql	Adds the service terminal printer service to the WW daemon table.
		POSINTERFACE_DBINIT.SQL	Initializes POS interface system settings.
		POSINTERFACE_TEXT_ENGUK.SQL	UK English version of messages written by the POS interface to the system log (loads tables PSS_Text).
		POSINTERFACE_TEXT_ENGUSA.SQL	USA English version of messages written to system log by POS interface.
		pss_count.sql	SQL script that lists a count of records for each table in the PSS database.
		PSS_Create_And_Init.bat	Batch file that runs all of the scripts necessary to create and initialize the database for a generic PSS system.
		pss_create_db.sql	SQL script to create the PSS database tables.
		pss_delete.sql	SQL script to delete all of the data from the PSS database tables.
		PSS_Delete_And_Init.bat	Batch file that runs all of the scripts necessary to delete the existing data and reinitialize the database for a generic PSS system.
		pss_drop_db.sql	SQL script to delete all of the PSS database tables.
		pss_init_data.sql	SQL script to initialize most of the necessary settings and text for the generic PSS system.
		pss_init_text_2.sql	SQL script containing text for the English – UK language.
		pss_init_text_3.sql	SQL script containing text for the French language.
		pss_init_text_4.sql	SQL script containing text for the Dutch language.
		PSS_SYMBOL.ER1	ERwin database model of the PSS database.
		UMS_System_Settings.sql	Adds UMS constants to system settings.
		UMS_Text.sql	Adds UMS text strings to text table in US English.

Folders	SubFolder	Files	Description
		UmsCleanup.sql	Sets all UMS devices to "logged out" at startup / shut down.
		UpdateOwave.bat	Generic update batch file that runs an SQL script on WaveWorks database.
	Logs		
		Ess01.tr1	Trace file for entrance units which use serial communications.
		Prs01.tr2	Trace file for printers which use serial communications.
	Print		Temporary holding location for transaction ticket print files.
	Processed		Holding directory for files received from POS after processing by PSS. (Note: this directory name and the filenames identified below are configurable to accommodate POS systems; the names listed are the *default entries.)* **Shopping transaction files:** *cardnumber_yyyymmddtttt*_PSS completed where *yyyy* = year, *mm* = month, *dd* = day and *tttt* = time; this is the PSS shopping transaction as scanned by the customer. *cardnumber_yyyymmddtttt*_POS. completed where *yyyy* = year, *mm* = month, *dd* = day and *tttt* = time; this is the POS shopping transaction as completed at the register. **Item files:** PLUNW.PSS - a full item (price look up) file transfer. PLUMT*nn*.PSS – a partial (maintenance) item file update, where *nn* is a sequential number reset each day. **Customer files:** CUSTMTnn.DAT – a maintenance update to the customer data file.
	StepDev		Files with an .sws extension are links between COM Objects and transaction Ids. Files with an .swv extension are ScreenMaker interface files.
		DiManager.swv	Display interface manager interface.
		EHT.swv	RF entrance interface.
		ExpressShopper.swv	Interface for express shopper.

		install.bat	Batch file run to install scripts to (base) WaveWorks database.
		MainMenu.swv	Initial interface that launches either UMS or shopping trip.
		PosInterface.sws	Associates WaveWork transactions with COM methods that perform POS interface functions.
		Pss.swp	Used by ScreenMaker.
		PssActivityCom.sws	Trans ID—shopping activity log functions.
		PssCustomerMaintCom.sws	Trans ID—customer download function.
		PssCustomerRank.sws	Trans ID—customer ranking function.
		PSSDbCleanup.sws	Trans ID—database cleanup function.
		PssFileMaintCom.sws	Trans ID—COM methods that perform file housekeeping.
		PssQOBarcodeCom.sws	Trans ID—quick order barcode functions.
		PssShoppingTripCom.sws	Trans ID—shopping trip functions.
		PssSpecialBarcodeCom.sws	Trans ID—special barcode functions.
		PssSystemMessage.sws	Trans ID—process system messages function.
		PssTimerCom.sws	Debug tool for shopping trip.
		PssTransactionTicket.sws	Trans ID—transaction ticket printing function.
		PssUtilsCom.sws	Trans ID—utility function.
		QuickOrder.swv	Quick order interface.
		release.ini	List of multiple .sws and .swv files to install.
		ShoppingTrip.swv	Shopping trip interface.
		sminstall.exe	Installs .sws and .swv files.
		UMS.sws	Trans ID—UMS COM methods.
		UMS.swv	UMS interface.
		update.bat (customer specific data)	Batch file used to change base script or add new ones.
		update.ini (customer specific data)	List of files to update for a customer.
	Transfer		The holding location used by the PSS and POS for exchanging files (i.e. both systems read from and write to this directory). Under normal conditions, files reside here only temporarily.
	Web		All *.asp files are Active Server Pages.
		admincradles.asp	Administer cradle hardware.
		admindispensers.asp	Administer dispenser hardware.
		admineditterminal.asp	Edit terminal settings / status.
		adminentrances.asp	Administer entrance unit hardware.
		administer.asp	Presents the service terminal menuing system.
		adminpageactions.asp	Administers authenticated service terminal page actions.
		adminpages.asp	Administers authenticated service terminal pages.
		adminpos.asp	View and administer point-of-sale system status.
		adminpowersupplies.asp	Administer power supply hardware.
		adminprinters.asp	Administer printer hardware.
		adminservices.asp	Administer PSS NT service components.
		adminterminals.asp	Administer terminal hardware.
		adminusers.asp	Administer service terminal users.

Folders	SubFolder	Files	Description
		fielddate.inc	Script file to present a standard date input field.
		fieldtime.inc	Script file to present a standard time input field.
		global.asa	Active Server Page application file that sets up database connections for the PSS service terminal.
		index.asp	Redirects the service terminal to the appropriate first page.
		index.html	Redirects the service terminal to the appropriate first page.
		issueterminal.asp	Issues a hand-held terminal to a customer and displays the location of the terminal.
		keyboard.asp	Displays an on-screen keyboard for use with terminals without keyboards (currently unused).
		login.asp	Presents a user login prompt and performs user authentication.
		lookupuser.asp	Looks up a customer by account details.
		pss.css	Cascading Style Sheet file to create consistent positioning and font formatting throughout the PSS service terminal.
		pssdatatableend.inc	Active Server Pages script file to manage the portion of a data table after the data is displayed.
		pssdatatablestart.inc	Active Server Pages script file to manage the header and retrieval of data for a data table.
		pssfunctions.inc	Active Server Pages script file containing numerous functions for use throughout the service terminal.
		psstatus.inc.	Active Server Pages script file to calculate the current status of the Portable Shopping System.
		quickorder.asp	Displays an individual quick order in the quick order frames page.
		quickorderframes.asp	Displays multiple quick orders.
		quickorderheader.asp	Manages display of multiple orders on the quick order frames page.
		registeruser.asp	Creates or administers PSS customers.
		sendmessage.asp	Sends ad hoc messages to PSS customers, or defines and schedules system messages.
		systemproblems.asp	Views any current problems with PSS.
		systemsummary.asp	View an operational summary of PSS.
		unabletoissue.asp	Displays errors in issuing a terminal to a customer.
		viewdepartments.asp	View the configured item departments.
		viewitemdetails.asp	View the details of a particular item.
		viewitems.asp	View the currently configured items.
		viewmarketing.asp	View and administer the currently configured marketing messages.

		viewmessagelist.asp	View the currently configured system messages.
		viewmessagelog.asp	View the PSS system log.
		viewnlstext.asp	View the currently configured language-dependent text.
		viewopeningmessages.asp	View the currently configured opening messages.
		vieworderitems.asp	View the items selected for a particular quick order.
		vieworders.asp	View the currently entered quick orders.
		viewreportrescan.asp	View the Shopper Rescan Report.
		viewreportsales.asp	View the Sales Report.
		viewreportshoppersummary.asp	View the Shopper Summary Report.
		viewreportterminals.asp	View the Hand-Held Terminal Report.
		viewreportunknownitems.asp	View the Unknown Item Report.
		viewrescanresults.asp	View the results of rescans in a particular customer's historical shopping trips.
		viewrescansettings.asp	Administer the PSS Rescan system configuration.
		viewshoppers.asp	View and administer the customers currently using PSS.
		viewshoppingactivities.asp	View the activities that occurred during a particular customer's current or historical shopping trip.
		viewshoppingitems.asp	View the items selected for purchase during a particular customer's current or historical shopping trip.
		viewshoppinglists.asp	View the list of a particular customer's current, historical, and incomplete shopping trips.
		viewsystemsettings.asp	View and administer PSS system settings.
		viewtaxtableentries.asp	View the entries for a particular PSS tax table.
		viewtaxtables.asp	View the list of PSS tax tables.
		viewterminallist.asp	View a terminal issue list of terminals available by dispenser.
		viewusersummary.asp	View a particular customer account summary.
	Web – Images	*various*	Contains graphics files that are used throughout the Service Terminal application (e.g., the red, yellow and green traffic light on the Main Menu.)
	Working		Where files reside while the PSS software is processing them.
TftpBoot			Location of software files which are downloaded to hand held terminals when they boot.
TftpServer			Service that provides for file transfers.

D:\ (database repository)

Folders	SubFolder	Files	Description
Mssql7	*varies*		Location of all database tables, in Microsoft SQL Server format.

7.1 Scheduled Tasks

Folders	SubFolder	Files	Description
C:\Winnt\Tasks			Repository for time-based tasks to be defined and scheduled.

8. Software Description

The subsystems included in the WaveWorks PSS system may consist of any combination of the following:

COM Objects	Contain methods invoked via WaveWorks transactions. Managed by the Message Server.
NT Services	Contain methods invoked via WaveWorks transactions. Usually started at system startup.
StepServer Scripts	Manage the user interface and screen flow for the hand held terminals. Maintain client data variables. Invoke business services (COM object methods) via WaveWorks transactions.

8.1 PSS Transaction IDs / COM Object Method Listing

The following tables list the transactions used to perform business logic, the methods which are invoked, and the input and output parameters.

POSInterface

Transaction ID	Method Name	Input Parameters	Return Parameters
P2P002	LoadPOSTransaction	FilePath	Result
P2P003	LoadPOSItemFile	FilePath	Result
P2P005	LoadTaxTable	FilePath	Result
POS100	POSstartup		Result
P2P103	POSSItemFileTransfer		Result
P2P199	FileTransferNoOP		Result
P2P304	EnableItemUpdates		
P2P404	DisableItemUpdates		
POS300	POSLogin		PosVersion TransactionCheckInterval ItemUpdatesCheckInterval CopyTaxTablesWithItems PSSTransactionType NumberPriceBytes LoyaltyCardnumLength LANAdapterNumber TransferAreaPath POSControllerName Result
POS301	SetCommLinkStatus	LinkStatus	Result

PSSActivity

Transaction ID	Method Name	Input Parameters	Return Parameters
PSA010	PssActivityAdd	CustomerId ShoppingListId ActivityId DetailText	RetValue

PSSSpecialBarcode

Transaction ID	Method Name	Input Parameters	Return Parameters
PSP001	Refresh		RetValue
PSP010	PssGetBarCodeType	InputBarcode TerminalId	ItemId Action BarcodeType RetValue

PSSQOBarcode

Transaction ID	Method Name	Input Parameters	Return Parameters
PSQ010	GetMaxQONumber	CustomerId ShoppingListId	QONumber RetValue
PSQ020	PssQOBarCode	CustomerId ShoppingListId PreviousQONumber PreviousItemId CurrentBarcode Action PreviousDepartmentId	ThisItemId This ItemName ThisItemUnit ThisItemDepartmentId ThisItemQty ThisQONumber RetValue
PSQ030	PssQOSend	CustomerId ShoppingListId QONumber	RetValue
PSQ031	PssQOQuit	CustomerId ShoppingListId QONumber	RetValue
PSQ040	PssQOGetTotalQty	CustomerId ShoppingListId QONumber	TotalQuantity RetValue
PSQ050	PssQOListItems	CustomerId ShoppingListId QONumber	ItemNames ItemUnits ItemQty RetValue

PSSShoppingTrip

Transaction ID	Method Name	Input Parameters	Return Parameters
PST001	CreateShoppingList	TerminalId CustomerId	RetValue
PST003	SwapTerminal	TerminalId CustomerId	RetValue
PST004	ValidateCustomer	CardNumber	Output CustomerName LanguageId Greeting CustomerId Result
PST005	PssSessionSwitch	TerminalId InputBarcode Customer Id ShoppingListId	NewCustomerId NewShoppingListId SessionStatus RetValue
PST007	GetSessionData	TerminalId CustomerId ShoppingListId	Result
PST008	GetShoppingTripType	TerminalId TripType	RetValue
PST010	StartShoppingTrip	TerminalId	CustomerId ShoppingListId CustomerName CustomerLanguageId CustomerCurrencyId RescanLevel StoreNumber RetValue
PST020	AdjustShoppingItemData	InputBarcode CustomerId ShoppingListId ItemQuantity ListValue Action CustomerCurrencyId TerminalId CMSName CMSApplication TerminalGroup	ItemDescription ItemPrice ItemCount TotalValue TotalCount Msg EmbeddedPriceItem RetValue
PST025	SetShoppingListStatus	CustomerId ShoppingListId ShoppingListStatus	RetValue
PST030	GetShoppingList	CustomerId ShoppingListId CustomerCurrencyId	ItemPrice ItemQuantity ItemDescription ItemId EmbeddedPrice TotalValueString TotalValueInt ItemCount RetValue

PST035	GetDisplayTotalByCurrency	CustomerId ShoppingListId CustomerCurrencyId CurrencyIndex	CurrencyIndex DisplayTotal Result
PST038	AdjustShoppingList	ItemId CsutomerId ShoppingListId ItemQty Action TerminalId CmsApplication StartIndex NumberOfItems	ItemDescription ItemPrice ItemCount TotalValue TotalCount Msg EmbeddedPriceItem ItemId MsgFlags RetValue
PST040	GetStoreInformational Message	CustomerId ShoppingListId	Text Message RetValue
PST041	GetCustomerMessage	CustomerId ShoppingListId	Text Message RetVal
PST042	GetElectronicMarketingMes	CustomerId ShoppingListId ItemId TerminalId CMSName CMSApplication TerminalGroup	TextMessage Result
PST043	GetCustomerCardNumber	CustomerID	CardNumber Result
PST100	EndShoppingTrip	CustomerId ShoppingListId TerminalId TerminalLocation RescanLevel	Rescan RetValue
PST102	DeleteShoppingList	TerminalId	RetValue
PST104	DetermineEndShoppingTrip	CustomerId ShoppingListId TerminalId TerminalLocation RescanLevel	Rescan Result
PST200	GetExpressCustomer		CustomerId Result
PST999	Refresh		

TransactionTicket

Transaction ID	Method Name	Input Parameters	Return Parameters
PST101	PrintTicket	CustomerID ShoppingTripID TransactionType PrinterService	RetValue

PSSSystemMessage

Transaction ID	Method Name	Input Parameters	Return Parameters
PST103	ProcessMessages		RetValue

PSSUtils

Transaction ID	Method Name	Input Parameters	Return Parameters
PSU001	GetText	TextId CustomerLanguageId	TextMsg RetValue
PSU002	GetTextWithParms	TextId CustomerLanguageId TextParameters	Text Result
PSU003	GetInActivityTimeoutString	InactivityIndex	InactivityString Result

UnitMgmt

Transaction ID	Method Name	Input Parameters	Return Parameters
UMS001	ReserveBestTerminal	EntranceId	TerminalId DispenserId CradleId DispenserName Result
UMS002	TerminalIssued	TerminalId	Result
UMS003	TerminalReturned	TerminalId Location IssueTime ReturnTime PlusScans MinusScans FailedScans	Result
UMS004	TerminalTimeout	TerminalId	Result
UMS005	TerminalLogin	TerminalId Location TerminalType	Result
UMS006	TerminalLogout	TerminalId	Result
UMS007	GetTerminalStatus	TerminalId	StatusCode StatusText StatusTimestamp Result
UMS008	SetTerminalStatus	TerminalId StatusCode Reason	Result
UMS009	ResetTerminalStatistics	TerminalId	Result
UMS010	GetTerminalTypeInfo	TerminalType	TerminalClass DefaultInterface DisplayRows DisplayColumns KeypadType Result
UMS011	GetCurrentInterface	TerminalId	InterfaceName Result
UMS012	SetCurrentInterface	TerminalId InterfaceName	Result
UMS013	UpdateBatteryLevels		Result

UMS014	SetBatteryStatus	TerminalId BatteryStatus	Result
UMS020	ReleaseTerminal	TerminalId	Result
UMS021	RebootTerminal	TerminalId BootType	Result
UMS022	UnlockTerminal	TerminalId	Result
UMS023	QueryTerminal	TerminalId	Location Result
UMS024	ResetTerminalSession	TerminalId	Result
UMS025	SetTerminalUnlockMode	TerminalId UnlockMode	Result
UMS026	TestCardReader	EntranceId	CardData Result
UMS027	ReportBatteryCondition	TerminalId BatteryCondition	Result
UMS028	UpdateTerminalLocation	TerminalId Location	Result
UMS100	EntranceLogin	HardwareId StatusCode	EntranceId Mode Result
UMS101	EntranceLogout	EntranceId	Result
UMS103	SetEntranceStatus	EntranceId StatusCode	Result
UMS104	GetEntranceMode	EntranceId	Mode Return
UMS105	UpdateEntranceModes		Return
UMS106	GetEntranceConfig	EntranceId	ReaderType ReaderPort DisplayType DisplayPort PopupTimeout CardLength Return
UMS107	ProcessCardData	EntranceId RawData	CardNumber CardStatus Return
UMS108	GetDispenserCounts	DispenserId	TotalCount IssueCount ChargingCount UnavailCount OtherCount Return
UMS200	PrinterLogin	HardwareId PrinterStatus	PrinterId Result
UMS201	SetPrinterStatus	PrinterId PrinterStatus Reason	Result
UMS202	GetPrinterConfig	HardwareId	PrinterType PrinterPort Return
UMS999	Refresh	N / A	Return

PSSCustomerMaint

Transaction ID	Method Name	Input Parameters	Return Parameters
P2P006	LoadFromFile		

PSSFileMaintenance

Transaction ID	Method Name	Input Parameters	Return Parameters
P2P201	CleanupPssTransFile		
P2P298	CleanupArchiveArea		
P2P299	ConvertSltoSO		

PSSDbCleanup

Transaction ID	Method Name	Input Parameters	Return Parameters
PST105	PerformDBCleanup		

PSSCustomerRank

Transaction ID	Method Name	Input Parameters	Return Parameters
PST106	RankCustomers		

8.2 Unit Management Subsystem

The Unit Management Subsystem (UMS) manages the Symbol hardware devices that comprise the PSS system. Those devices include hand held terminals, cradles, entrance unit devices, ticket printers, and power supplies. Since UMS manages hardware, it is important to understand each of the devices being managed in both the Asterix2 and Asterix3 versions of PSS.

8.2.1 Hardware Overview

Terminals

The PSS system only uses RF terminals. PSS can use either Asterix2 or Asterix3 terminals. The Asterix2 terminal has five buttons and a 4 x 20 display. The Asterix3 terminal has four buttons and a trigger (in lieu of the "+" key on the Asterix2), and an 8 x 20 display. Both terminal types charge their batteries while in the cradle, but since the Asterix3 uses a newer lithium battery, the battery-related constants are different for the two terminals.

Cradles

The cradles are completely different between the Asterix2 and Asterix3 systems. The Asterix2 terminals are placed into an integrated eight-slot rack, and as a result, the entire system is comprised of terminals in multiples of eight. When the Asterix2 is unlocked, a small light on the face of the terminal is lit. The Asterix3 cradles are individual units, and the entire system can include any desired number of terminals. When the Asterix3 is unlocked, the cradle housing lights up. The Asterix2 cradle locks the terminal tightly against the contacts. The Asterix3 cradle does not--which causes a number of issues, since the terminal can be lifted off the contacts before the latch stops it.

Dispensers

In a typical Asterix2 system a *module* was a large plastic device that had eight integrated cradles. Four modules were a *unit.* A *dispenser* in the Asterix2 world consisted of one or more modules. The Asterix3 system uses individual cradles that can be physically arranged however the customer wishes. An Asterix3 *dispenser* is simply a logical group of cradles (that are usually physically adjacent to each other).

The concept of a dispenser being a logical group of cradles also works for Asterix2 systems as long as the cradles in the same module are part of the same dispenser (to avoid confusing the customer). So, in this document, a *dispenser* is really a collection of cradles, not a physical device.

Entrances

There are two types of entrance stations: serial and RF. Either type can be used with either terminal type, but the older serial entrance units are typically used on Asterix2 systems and the newer RF entrances are used with the Asterix3 terminals. The primary difference between them is how they are controlled from the host.

The serial entrance units are controlled by NT Services (one service per entrance unit). These services are named "PssEntrance01," "PssEntrance02,", etc. These services are WaveWorks clients and directly control the entrance display and process input. The RF entrance units communicate with a single instance of an NT service (EhtService) that translates STEP commands into low-level commands the hardware can understand. DisplayServer scripts handle the screen display and input processing.

Printers

Transaction tickets are (optionally) printed at the conclusion of a self-scan shopping trip. The ticket is used to direct the shopper to a quick pay or rescan lane and also as a means of identifying the shopper to the POS system. Transaction tickets are an optional feature of the PSS system. PSS supports three types of ticket printers: RF rack, serial rack, and serial service terminal. For installations that do print tickets, a printer is needed at the return rack and at the service terminal.

Power Supplies

The PSS system also manages power supplies. There isn't much to manage aside from ensuring that only one terminal per power supply is unlocked at a time. Unlocking more than one terminal at a time could cause the power supply to fail. Power supplies normally have a status of Ready (status code = 0). If a terminal that is powered by the power supply is being unlocked, the status is set to Reserved (status code = 1). If all the terminals attached to a power supply report that they are on battery power, then the power supply status is set to "Check Hardware" (status code = 6) to indicate that perhaps the power supply has failed or is unplugged.

8.2.2 Software Components

The Unit Management subsystem consists of the following software:

Unit Management COM object—provides all the UMS business services. The services maintain data on hardware status, battery level, and terminal location. The services also send commands to the hand held terminals while they are in the dispenser. The UMS transaction model (description of all UMS transactions) is found in the Appendix.

Entry Station Service—uses PSA's Hardware Isolation Library (HIL) to communicate with the Entrance Station hardware devices (serial entrance stations only).

Printer Service—Uses PSA's Hardware Isolation Library (HIL) to communicate with serially connected printers.

UMS DisplayServer Interface—controls the display on the hand held terminal and also provides procedures for handling input from the terminal and commands the UMS COM object.

EHT DisplayServer Interface—controls the display on RF entrance units and provides procedures for handling input from the entrance.

8.2.3 Database Access

The Unit Management COM object maintains the following database tables. Refer to the Data Dictionary in the Appendix for a description of the tables:

UMS_Entrance—maintains state information on entrance unit devices

UMS_Printer—maintains information on printer devices

UMS_Terminal—maintains information on hand held terminals

UMS_Dispenser—maintains information on dispensers (groups of cradles)

UMS_Status_Code—list of valid status codes for UMS devices

UMS_Terminal_Types—list of valid terminal types

UMS_Cradle—maintains information on terminal cradles

UMS_Power—maintains information on power supplies

UMS_Card_Reader_Type—maintains a list of supported card reader types and properties

8.3 Shopping Trip Subsystem

Once the shopper retrieves the proper hand held terminal from the dispenser, the Shopping Trip Subsystem allows the shopper to add and delete items from their basket and view the total amount and number of items scanned during their current shopping trip. Shopping trips can be ended by returning the hand held terminal to an empty dispenser slot or by scanning an "End of Trip" barcode. Each activity during the shopping trip is logged and available for review from the Service Terminal.

8.3.1 Start of Shopping Trip Processing

When a customer "swipes" their card to reserve a scanner to go shopping, the system will perform the following:

1. ValidateCustomer—verifies the following:
    - that the customer is in the Pss_Customer database table and is not suspended;
    - that the customer has no outstanding transactions (in the Pss_Shopping_List table), and
    - that the card has no current shopping trips (in the Pss_Session table).
2. CreatShoppingList—initializes the tables Pss_Shopping_List, Pss_Session, and Pss_Shopping_Activity.
3. StartShoppingTrip—occurs at removal of the scanner from the dispenser.
4. GetText—retrieves all text necessary for display to the shopper (in the shopper's language).

8.3.2 Shopping Trip Processing

For Asterix 2 terminals, the "+" key is a physical key on the scanner; for the Asterix 3 terminals, the "+" key is the trigger. The "-" key is a physical key on either terminal.

After the shopper scans a barcode, the following steps are taken in the StepServer Script procedure ProcessInputData to process the barcode:

1. Parse the barcode
    - A call to PssSpecialBarcode to break up the scanned input field into its disparate parts
    - Modifications to the barcode from what was reported as scanned to what is needed to compare into the item table
    - Determination of whether a barcode is a "special" barcode that requires special handling (e.g. the End of Trip barcode)
2. If, in the output from the above method, a "special" barcode is encountered, it is specifically handled

3. If, in the output from the above method, an item barcode is scanned, then the AdjustShoppingList method is called to add or delete the scanned barcode from the customer's shopping list. Any anomalous output from this method is handled via error screen displays (i.e., exception items, invalid barcodes).

The following is the database activity for a "+" or "-" key event:

1. Retrieve customer data from Pss_Customer table.
2. Verify input barcode.
3. Determine if it is a variable weight barcode using the Pss_Barcode_Variable_Weight

   - if variable weight, get the price and item Id

   - else use the scanned barcode and data from the Pss_Item table to execute the pricing algorithm

4. Add the item to the shopping list ("+") or remove the last scanned match for this item id ("-") in the Pss_Shopping_Item.
5. Calculate the total tax from the Pss_Tax_Table and Pss_Tax_Table_Entry and write the value to Pss_Shopping_Tax.
6. Read for marketing messages from the Pss_Marketing_Item, Pss_Marketing_Department, and the Pss_Marketing_Mfg tables and write to the Pss_Marketing_Sent table any messages sent to the shopper.
7. Convert pricing values using the Pss_Currency table.
8. Write the total to the Pss_Shopping_List table.
9. Record the event into the Pss_Shopping_Activity table.

Anomalous conditions encountered in this process will be logged to the Pss_Message_Log table. If the scanned item Id is not in the Pss_Item table, then record this in the Pss_Uknown_Item table.

All text used throughout the shopping trip is stored in the Pss_Text table and is accessed through a language Id associated with the customer or store.

Describe what happens when a customer presses the "=" key to view the summary.

The shopper presses the "=" key, and in the StepServer script procedure ProcessInputData, a call to one of the ShoppingTripList functions is made. This will return character strings with item Ids, descriptions, and prices (see COM object definitions for full parameter list). These strings will contain all items currently in the customer's basket in a pipe-separated format. The lists are controlled through the pss_system_setting database table. Based on these settings, display field size, list order (ascending or descending), and contents of the list are established for consistent display to the shoppers. All database access is "read only," with the exception of an insert into the Pss_Shopping_Activity table to record the event.

The following libraries and COM Objects are available for the shopping trip subsystem to use:

PssActivityCom	COM object interface that allows activities to be added to the Pss_Shopping_Activity database table to record shopper key presses and any other event.
PSS ActivityLib	This library handles the database insert, deletion, and retrieval of Pss_Shopping_Activity database table rows.
PssBarcodeLib	Library that interprets the barcodes scanned by the shopper. Has methods to calculate and strip off check digits, determine barcode type (i.e., UPC, EAN13), etc.
PssConfigurationLib	Library used to access the database table Pss_System_Settings that contains the client-selected configuration options.
PssPricingLib	This library provides the core of the pricing method algorithms as defined in the appendices.
PssShoppingTripLib	Library that handles the database insert, deletion, and retrieval of Pss_Shopping_List and Pss_Shopping_Item database table rows.
PssShoppingTripCom	This is the COM object interface allowing • shopping trips to be started, • items to be added and removed from shopper's lists, • lists to be formatted for shopper review, • shopping trips to be ended, • as well as a series of general support functions (such as validating the customer, retrieving shopper messages of various sorts, and maintaining the status of the current shopping trips).
PssSpecialBarcodeCom	This is the COM object interface to allow access to the special barcode functions, determining if the scanned barcode requires modification or requires special handling methods.
PssSpecialBarcodeLib	Library to access the Pss_special_Barcode table that contains definitions of all special barcodes, descriptions, and return types for StepServer script statement handling.

In general, the COM Objects handle the interface to the StepServer Scripts and perform the business logic and the libraries provide access to the persistent data.

The following user exits are called during shopping trips:

UE_PreRescanDetermination	Allows modification of the rescan calculation data prior to the calculation, or allows wholesale replacement of standard rescan calculation with a customized algorithm
UE_PostRescanDetermination	Allows modification of the result of a standard rescan calculation
UE_PreProcessMarketingMessage	Allows additional item marketing messages to be allowed / disallowed prior to the standard method's retrieval
UE_PrintReceipt	On End of Trip, allows modification to the receipt file to be printed on the ticket
UE_PostProcessIsVariableWeight	After an item has been determined to be an embedded price / weight item (or not), custom code can be added here to modify the decision.

If using the End of Trip barcode feature, configure any type of barcode with a check digit to be the End of Trip barcode. The Pss_Special_Barcode table should contain the End of Trip value.

8.3.3 End of Shopping Trip Processing

This can happen in one of two ways depending upon the Pss_system_Settings database table values. Either the shopper scans a predetermined End of Trip barcode (as specified in the Pss_Special_Barcode table) or places the scanner back into the dispenser.

If an End of Trip input is recognized (either an "E" from the dispenser, or and End of Trip barcode), the StepServer script procedure EndOfTrip is executed. This simply calls the PssShoppingTripCOM object method EndShoppingTrip and then handles the resetting of the shopping list status (if the scanner was in Queue Buster mode), then proceeds to the final Thank You screen.

Within the EndShoppingtrip method, the following is done:

1. The customer data and shopping trip data are examined to determine if the trip is to be rescanned or not.
2. The transaction barcode is calculated.
3. If enabled, the PssTransferFile is written with the list of items scanned by the shopper.
4. If enabled, the printer is requested to print the transaction ticket.
5. Customer and Shopping Trip data and status are updated.
6. Session data is deleted or modified if in queue buster mode.
7. Data in the database tables related to the shopping trip are deleted, such as Pss_Order_List_Item, Pss_Order_List, Pss_Marketing_Sent and Pss_Shopping_Tax.

Database tables associated with the shopping trip are:

Pss_Currency
Pss_Customer
Pss_Language
Pss_Text
Pss_Shopping_List
Pss_Session
Pss_Shopping_Item
Pss_Shopping_Activity
Pss_Shopping_Tax
Pss_Tax_Table
Pss_Tax_Table_Entry
Pss_Unknown_Item
Pss_Item
Pss_System_Setting
Pss_Message_Log

Other objects such as classes need to be defined:

PssStoreInformation
PssSpecialBarcode
Session
Language Text
Activity
Customer
Customer Rescan
Event Scheduler

8.3.4 Shopping Trip Message Log Entries

This describes the log messages found in the Pss_Message_Log table that originated within the Shopping Trip. The text shown here is in English, since the inclusion of all language text for all messages would be excessive. This text is configurable in the database table Pss_Text, so if a message is not here or not exactly as stated here, then the database table probably has changed. All text is accessed through the GetText method, which accepts the Text_Id as an input parameter.

8.3.5 Fatal Messages:

"No Session for this Terminal (%s)" ST_TEXT_NOSESSION	- Attempted to start the Shopping trip without associating a customer with a terminal. Verify that CreateShoppingList completed successfully prior to calling StartShoppingTrip.
"No Customer Data" ST_TEXT_NOCUSTOMER	- Customer's card does not exist in the database.
"Initialization FAILED(%s)" ST_TEXT_INITFAILED	- Could not initialize Pss_System_Setting configuration data from database – Methods may not be executing along desired paths.
"No Express Customers Available" ST_TEXT_NO_XCUSTOMERS	- No more Express Shopper customers available for express shopper to use. Add more to the database.
"Unable To Access Express Customers Data" ST_TEXT_NO_ACCESS_XCUSTOMERS	- Database error accessing Express Shopper customers.

8.3.6 Error Messages:

"Activity Add" ST_TEXT_ACTIVITY_ADD	- Failure to add an activity to the Pss_Shopping_Activity table. Generally when the shopping trip does not start properly, there is an invalid key constraint problem.
"Invalid Customer Data associated with terminal (%s)" ST_TEXT_INVALIDCUSTOMER	- Invalid Customer Id, not found in database. Verify customer exists in database.
"Invalid Get Rescan data for customer (%s)" ST_TEXT_INVALIDRESCANDATA	- Failure getting customer rescan data. Verify customer exists in the database and has valid rescan level and accurate trip counters and rescan counters in the Pss_Customer table.
"Invalid Sub Total for Customer (%s)" ST_TEXT_INVALIDSUBTOTAL	- Had an error calculating total for customer"s shopping trip. Verify values for this customer in the database.
"Invalid Customer (%s) Save Recalced Prices" ST_TEXT_BADSAVEPRICES	- Database error saving pricing data to the customer"s shopping list.
"UE_PreRescanDetermination GENERATED TRAP!!!do NOT rescan" ST_TEXT_USEREXITERROR	- Error in the user exit for determination of rescan. Rewrite the user exit without the problem causing the abnormal execution.
"Invalid Rescan Determination for Customer (%s)" ST_TEXT_RESCANDETERMINATION	- Error in determining whether to rescan a customer. Verify customer rescan data in Pss_Customer and Pss_rescan tables are consistent.
"UE_PostProcessRescanDetermination GENERATED TRAP!!! – do NOT rescan" ST_TEXT_USEREXITERROR	-Error in the user exit for determination of rescan. Rewrite the user exit without the problem causing the abnormal execution.
"UE Post CalculateRescan" ST_TEXT_USEREXITERROR	- Error in the user exit for determination of rescan. Rewrite the user exit without the problem causing the abnormal execution.
"UE Pre CalculateRescan" ST_TEXT_USEREXITERROR	- Error in the user exit for determination of rescan. Rewrite the user exit without the problem causing the abnormal execution.
"Invalid Customer (%s) Rescan Update" ST_TEXT_UPDATERESCAN	- On end of trip, could not update customer"s rescan data in the Pss_Customer table with latest data.
"Invalid Calculate Transaction Id for Customer (%s)" ST_TEXT_TRANSACTIONID	- Could not calculate correct transaction Id. Verify Pss_System_Settings (for Shopping_Trip Loyalty Card and Transaction barcode) are proper for this customer.
"Invalid Saving End of Trip Data for Customer (%s)" ST_TEXT_SAVE_EOT	- Error saving shopping trip and/or customer data for end of trip.

"Invalid Writing SCAN-IN %d" ST_TEXT_WRITESCANIN	- Error writing scanIn (PssTransferFile). Check for disk space availability, existance of proper directories and permissions to create files there.
"Cannot unload list items for Customer (%s)" ST_TEXT_UNLOAD	- Error-freeing memory used to store local copyof shopping list.
"Invalid Getting Printer CMSID ReturnCode, Customer %s" ST_TEXT_INVALIDPRINT	- Error printing transaction ticket. Verify printer and connections.
"Could Not delete Shopping Tax Table %d" ST_TEXT_BADDELETETRIP	- Error as specified in text deleting shopping data on end of trip
"Could not format item list (%s)" ST_TEXT_INVALIDFORMAT	- Error formatting shopping trip data for list display.
"Customer (%s) Could not get Qty" ST_TEXT_GETQTY	- Error retrieving shopping trip data for item barcode.
"Customer (%s) has less than zero items !!" ST_TEXT_LESSZEROITEMS	- Deleting of an item from a customer"s list yielded a less-than-zero number of items in the list. Ooops.
"Customer (%s) BAD Saving Item To Shopping List" ST_TEXT_BADSAVEITEM	- Could not save a item to the Pss_Shopping_Item table for a customer"s shopping list. Verify that the Pss_Session and Pss_Shopping_list tables have rows for this customer.
"UE_PreProcessMarketingMessage" ST_TEXT_USEREXITERROR	- Error value returned from user exit code. Fix the code.
"Customer (%s) Could not set Total Value" ST_TEXT_TOTALVALUE	- Error saving total value to Pss_Shopping_List table. Verify that the customer is valid
"No Session for this Terminal (%s)" ST_TEXT_NOSESSIONCUSTOMER	- Attempted to queue bust shopper without an associated exployee with a terminal. Verify that CreateShoppingList completed successfully prior to calling StartShoppingTrip.
"No Customer Data" ST_TEXT_NOCUSTOMER	- The customer barcode scanned by the employee for queue busting does not exist in the Pss_Customer table.
"No New Shopping List for this card (%s)" ST_TEXT_NOSHOPPINGLIST	- Could not get Pss_session table data for the terminal trying to do the queue busting
"Could Not initialize For card number (%s)" ST_TEXT_SESSIONBAD	- Could not exchange from employee to shopper for the terminal trying to do the queue busting.
"Could not Get Queue Buster session Data (%s)" ST_TEXT_QUEUEBUSTER	- Could not find queue buster Pss_session table entry for the shopper.
"Could not Swapping queue buster session for card (%s)" ST_TEXT_SWAPQUEUEBUSTER	- Could not exchange from shopper to initiating employee for the terminal trying to do the queue busting
"Could not delete shopping list just created for (%s)" ST_TEXT_BADDELETE	- Error deleting shopping data Pss_Session table after an error. Attempting cleanup
"Error Retrieving Customer Message for (%s)" ST_TEXT_NOCUSTOMERMESSAGE.	- Database error retrieving message for this customer. Verify existance of customer and message in PSS_Customer_Message table.
"Customer (%s) Could not create session" ST_TEXT_NOSESSIONSTART	- On starting a shopping trip, added a row to Pss_Shopping_List but could not add to the Pss_Session TABLE. Check for already existing row for this terminal in the Pss_Session table.
"Customer (%s) Could not delete shopping list" ST_TEXT_DELETESHOPPINGLIST	- Database error trying to cleanup after some other error the Pss_Session, Pss_Shopping_List and Pss_Shopping_Activity tables.
"Customer (%s) could not Set Shopping Status" ST_TEXT_SETSHOPPINGSTATUS	- Error setting the Pss_Shopping_List table status. Verify the customer has a valid row in this table.

"SwapTerminal Error for customer (%s)" ST_TEXT_SWAPTERMINALERROR	- Error updating the Pss_Session and/or Pss_Shopping_List tables while attempting a swap terminal command.
"Customer, Card Number (%s) is suspended" ST_TEXT_CUSTOMERSUSPENDED	- Logging the fact that a suspended customer has tried to release a scanner. From ValidateCustomer on card swipe.
"Customer for card is not found (%s)" ST_TEXT_NOCUSTOMERFORCARD	- Logging the fact that a nonexistant customer (as per Pss_Customer database table) has tried to release a scanner. From ValidateCustomer on card swipe.
"Error Validating Card; Customer (%s) has a Session" ST_TEXT_HASSESSION	- Logging the fact that this customer already has a scanner currently issued to them. From ValidateCustomer on card swipe.
"Error Validating Customer (%s)" ST_TEXT_VALIDATECUSTOMERERROR	- There was a Bad database problem on a card swipe trying to validate a customer.

8.3.7 Informational Messages

"No Session, or Swapped terminal (%s) - No trip to end" ST_TEXT_NOTRIP	- EndOfTrip method was executed, but there is no customer or terminal data to end. Check the Pss_Session table or the Pss_Shopping_List table for an entry.
"Scanner Abandoned by Customer (%s)" ST_TEXT_SCANNERABANDONDED	- A scanner that was detected as abandoned is being run through EndOfTrip. This will delete all information about the trip, saving nothing to history, and not sending any data to the POS.
"Done with special Customer (%s)" ST_TEXT_SPECIALCUSTOMER	- Logs that a store employee has completed a shopping trip and EndOfTrip has run, deleting all information about the trip and saving nothing to history and not sending any data to the POS.
"User Exit Returned Done %s" ST_TEXT_USEREXITDONE	- After user exit calls, to specify NOT using the standard algorithms that follow the user exit calls.
"For Customer, Error on Terminal Finding Item (%s)" ST_TEXT_NOITEM	- An error trying to find the item scanned in the Pss_Item table.
"For Customer, Error on Terminal Invalid Barcode %s" ST_TEXT_INVALIDBARCODE	- Logs that an invalid barcode was scanned or that the shopping trip was incorrectly started. Verify that data for the customer exists in the Pss_Session and Pss_Shopping_List tables
"No New Shopping List for this customer (%s)" ST_TEXT_NOCUSTOMERLIST	- No data in Pss_Shopping_Item table to list for this customer.
"Could not Get trip data to delete for terminal (%s)" ST_TEXT_DELETECUSTOMERSESSION	Error trying to clean up Pss_session and Pss_shopping_List tables after unsuccessful release of scanner.
"Customer (%s) Could not delete session" ST_TEXT_DELETETERMINALSESSION	- Error trying to clean up Pss_session and Pss_shopping_List tables after unsuccessful release of scanner.
"Customer (%s) has scanned an exception item" ST_TEXT_EXCEPTIONITEM	- Logs the scanning of an exception item by a customer.

8.3.8 Debug Messages:

The following are messages logged in the End Of Trip method to track progress through that method.

"EOT Done totals"	- Completed calculation of trip totals
"EOT Done Rescan"	- Completed Rescan calculation
"EOT Written to files"	- Completed writing Pss Transfer File
"EOT Done"	- End of trip completed successfully

8.3.9 Rescan Messages

These Information messages are specific to determine if a shopper gets rescanned or not.

"Into RescanLevel lOldLevel=%d, lPOSItemCount=%d, lPSAItemCount=%d, lBothItemCount=%d, lPOSItemValue=%d, IPSAItemValue=%d, 1BothItemValue=%d"	- Displays inputs into rescan calculation.
"Rescan by Currency"	- Employs the rescan calculation by absolute value difference in prices.
"IgnoreScanTooMuch"	Ignore overscans by customer in rescan determination.
"NOT IgnoreScanTooMuch"	- Do not ignore overscans by customer in rescan determination.
"Level diff RescanLevel Diff=%d, levelMod=%d"	- Displays the calculated difference and specified level change.
"Rescan by Percentage"	Employs the rescan calculation by percentage difference.
"new Level (%d) old(%d) modifier(%d)"	- Displays the shoppers new and previous rescan levels, and specified level change.
"Check new Level (%d) vs min(%d max(%d)"	- Displays new level and minimum and maximum allowed levels.
"Leave alone ret=%d"	- No change in shopper"s level.
"Customer (%s) rescan is FALSE - Gold Customer" ST_TEXT_RESCAN_GOLD	This customer will never be rescanned.
"Customer (%s) rescan is FALSE - Rescan Nobody, by date" ST_TEXT_RESCAN_NOBODYDATE	- During the current time period, nobody should be rescanned. This is selected from the rescan screen on the administrative service terminal.
"Customer (%s) rescan is FALSE - Rescan Nobody" ST_TEXT_RESCAN_NOBODY	- Nobody is to be rescanned. This is selected from the rescan screen on the administrative service terminal.
"Customer (%s) rescan is TRUE - Rescan Everybody, by date" ST_TEXT_RESCAN_EVERYBODYDATE	- During the current time period, everybody should be rescanned. This is selected from the rescan screen on the administrative service terminal.
"Customer (%s) rescan is TRUE - Rescan Everybody" ST_TEXT_RESCAN_EVERYBODY	- Everybody should be rescanned. This is e- Nobody is to be scanned. This is elected from the rescan screen on the administrative service terminal.
"Customer (%s) rescan is TRUE - Always First Trip" ST_TEXT_RESCAN_YESFIRSTTRIP	- This is the shopper"s first trip, rescan them.
"Customer (%s) rescan is FALSE - Never First Trip" ST_TEXT_RESCAN_NOFIRSTTRIP	- This is the shopper"s first trip, do NOT rescan them.

"Customer (%s) rescan is FALSE - Never Second" ST_TEXT_RESCAN_NOSECONDTRIP	- This is the shopper"s second trip, do NOT rescan them.
"Customer (%s) rescan is FALSE - Below Item Count" ST_TEXT_RESCAN_BELOWCOUNT	- This shopper"s trip is below the specified item count, do NOT rescan them.
"Customer (%s) rescan is FALSE - Below Value" ST_TEXT_RESCAN_BELOWVALUE	- This shopper"s trip is below the specified basket value, do NOT rescan them.
"Customer (%s) rescan is FALSE - Above Amount" ST_TEXT_RESCAN_ABOVEVALUE	- This shopper"s trip is above the specified basket value, do NOT rescan them.
"Customer (%s) rescan is FALSE - Above Item Count" ST_TEXT_RESCAN_ABOVECOUNT	- This shopper"s trip is above the specified item count, do NOT rescan them.
"Customer (%s) rescan is TRUE - Too long between trips" ST_TEXT_RESCAN_TOOLONG	- This shopper"s last PSS trip was too long ago, rescan them.
"Customer (%s) rescan is TRUE - Too long since last rescan" ST_TEXT_RESCAN_TOOLONGLASTRESCAN	- This shopper"s last rescanned trip was too long ago, rescan them.
"Customer (%s) rescan is FALSE - Too soon since last rescan" ST_TEXT_RESCAN_TOOSOONLASTRESCAN	- This shopper"s last rescanned trip was too short ago, do NOT rescan them.
"Customer (%s) rescan is FALSE - Never Consecutive" ST_TEXT_RESCAN_NO2INAROW	- Never rescan a shopper twice in a row, do NOT rescan.
"Time Window populations: QPCust = %d,%d,%d,%d; RSCust = %d,%d,%d,%d; QPItem = %d,%d,%d,%d; RSItem = %d,%d,%d,%d"	- Display data for load balance algorithm.
"Item Count Load Balance Average process time / PSS Lane = %.2f"	- Display data for load balance algorithm.
"HeadCount Load Balance #Rescan = %.1f, #QuickPay = %.1f"	- Display data for load balance algorithm.
"Customer (%s) rescan is FALSE - Load Balance headcount too big" ST_TEXT_RESCAN_QUEUEHEADCOUNTTOOBIG	- Do not rescan this customer, head count too large.
"Customer (%) rescan checking REDUCED – Load balance reduced checking" ST_TEXT_RESCAN_REDUCEDCHECKING	- Entering reduced rescan level due to load balancing.
"Customer (%s) rescan is TRUE - Load balance cashier IS available" ST_TEXT_RESCAN_CASHIERAVAILABLE	- Rescan this customer since load balancing cashier is available.
"Customer (%s) rescan is FALSE - Load balance cashier NOT available" ST_TEXT_RESCAN_CASHIERNOTAVAILABLE	- Do NOT Rescan this customer since load balancing cashier is not available
"TRUE - On max trip number since last rescan (%d)"	- Shopper is at maximum trip since last rescan for their rescan level, rescan them.
"Customer <%s> rescan is TRUE - Level=%d Pct=%d value=%d"	- Randomly calculated result yields a rescan for this shopper.
"Customer <%s> rescan is FALSE - Level=%d Pct=%d value=%d"	- Randomly calculated result yields a NO rescan for this shopper.
"Customer (%s) rescan had Error retrieving Rescan Data" ST_TEXT_RESCAN_DATA	- Error in reading this customer"s rescan data. Check the Pss_Customer table for valid data.

.

8.4 Quick Order Subsystem

While shopping, a user may choose to place a quick order through the Quick Order subsystem. The shopper identifies, through the quick order mechanism, the products and the quantities/weights of those products they wish to order. The quick order can then be sent to the appropriate department, where a web screen displays to store personnel the orders placed by shoppers. After they have finished processing the order, store personnel can send a message to the shopper that their order is ready. This is an optional feature.

PssQuickOrderBarcodeCom – Com object associated with quick order
PssQuickOrder – Class associated with quick order

Database tables associated with quick order are:

Pss_Order_List
Pss_Order_Status
Pss_Order_List_Item
Pss_Department

8.5 Queue Busting Subsystem

The Queue Busting application allows the PSS System to be used in a slightly different manner than normal, but also helps retailers and shoppers reduce the amount of check–out time. Any hand held terminal which has been released can be used for queue busting. This is normally done by store personnel to alleviate long check out lines which may have formed.

Transactions in this mode are initiated by scanning a special (configurable) barcode. More than one barcode can be configured for use in this manner. This barcode is expected to be on a plastic token. Once the special barcode has been scanned, the items in the customer basket are scanned. A PSS transaction is created in the same manner as a normal PSS transaction. After all items in the basket have been scanned, the special barcode, which began this mode, is scanned again.

This transaction is sent to the POS System. The plastic token containing the barcode is then handed to the customer with instructions to give it to the cashier at the check-out register. When the customer reaches the register, they present the token and their loyalty card (if used) to the cashier who then performs a standard non-audit self scan check out. After the customer checks out, the token can be reused for other customers.

There are no methods written specifically for queue busting. The hand held application differentiates between a normal shopping trip and a queue busting session.

PssSpecialBarcodeCom – Com object associated with queue busting
PssSpecialBarcodeLib – Library associated with queue busting

Associated database tables are:

Pss_Session
Pss_Special_Barcode

8.6 Electronic Marketing Subsystem

The PSS system also allows for a basic electronic marketing facility through the Electronic Marketing application. This application allows a retailer to send messages to a shopper's hand held terminal anytime that a given item is scanned. This is an optional feature.

PssEMarketingMessage - Class associated with electronic messages
ElectronicMarketing – Library associated with marketing

Database tables associated with marketing messages:

PSS_Marketing_Item
PSS_Marketing_Sent
PSS_Marketing_Mfg
PSS_Marketing_Department

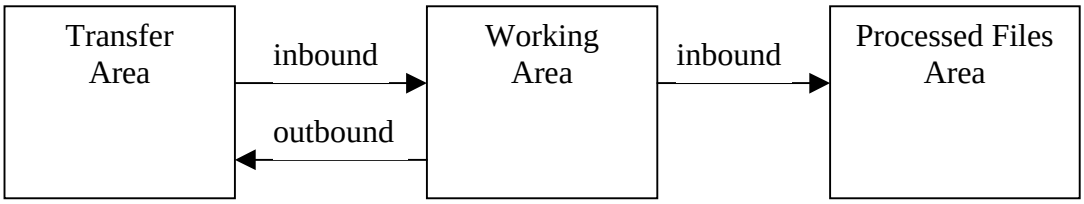
8.7 User Messaging Subsystem

Also, through the User Messaging application, the store has the option to send messages to shopper's hand held terminals. The messages can be any text, and can be made to be repeated, be displayed at fixed times during the day, and/or can be directed to any set of shoppers currently in the store. In addition, messages for a particular customer can be preloaded and are delivered to them the next time they shop with the WaveWorks PSS system.

PssSystemMessageCom - Com object associated with system messages
Pss_System_Message – Database table associated with system messages

8.8 POS Interface Subsystem

The WaveWorks PSS POS Interface is a file-based mechanism that provides communication between the PSS system and the store POS system by the exchange of files.



PSS software monitors and deposits files into a directory referenced as the *Transfer Area.* Inbound files (created by the POS) are detected in the Transfer Area and moved into the working area where a processing transaction processes the information in the file. When processing completes, the file is then moved into the Processed Files Area. Outbound files (created by PSS) are initially created in the working area and, when completed, are moved into the Transfer Area to be processed by the POS system.

Under normal operating conditions, files should only exist in the Transfer and Working areas temporarily. If files begin to accumulate in either of those areas, it is a sign of a system malfunction. Files moved into the Processed Files Area remain there until a File Housekeeping job deletes them. Files which are processed correctly are renamed with filetype "COMPLETED" and files which could not be processed are renamed with filetype "ERR".

The directory location of the Transfer Area and the Processed Files Area may be configured in the PSS System Settings, but they default to the "Transfer" and "Processed" subfolders of the folder where PSS was installed: (C:\Pss\Transfer and C:\Pss\Processed). This Transfer Area folder location is identified by System Setting value POSINTERFACE/Local_TransferArea.

If the POS system supports network file paths and WindowsNT can be configured to map a directory on the POS, then PSS can be configured to retrieve and send files to a directory area on the POS system. The Processed Files Area is identified by a System Setting value POSINTERFACE/ ProcessedFileArea. Since this folder can accumulate large numbers of files depending on how frequently File Housekeeping is configured to run, this directory may be configured to be on a different disk device than the disk on which PSS is installed. This would be done to avoid filling up that disk and causing a Disk Full problem.

8.8.1 POS Interface Files

The files defined are:

- PSS Transaction file—sent from PSS to POS to identify a shopper and the items they have scanned.
- POS Transaction file—sent from POS to PSS to identify that a shopper has paid for their trip and allows for sending the items scanned by the POS system.
- Item Record file—sent from POS to PSS and contains all of the items in the POS database such that the PSS database can be updated to reflect the items contained in the POS database.
- Price Changes file—subset of the Item Record file, processing is the same.
- Tax Table files—sent from the POS to PSS and contain tax information used by PSS to perform transaction tax calculations.
- Customer Update File—sent from POS to PSS to update the PSS database with information about valid PSS shoppers.

The following sections describe the file layout for each of the file types.

(File layouts unchanged—skipping to next section)

8.8.2 POS Interface Software Entities

The following software components are included to process the files listed above. A service exists which monitors a specific folder for the arrival of these files. Files created by PSS are, when detected, transferred to the POS system using the configured "transfer" transaction. Files created by the POS system are, when detected, "processed" on the PSS system using the configured "processing" transaction.

8.8.2.1 PSSFileMonitorSvc

PSSFileMonitorSvc is the service which monitors Transfer Area for the arrival of files from the POS system and for files generated by the PSS system. It moves inbound files into the Working Area and, when appropriate, initiates WaveWorks transactions for performing file-processing functions. It can also be configured to initiate transfer transactions when it detects outbound files. Release 1 of PSS provides no transfer transactions for sending files to a POS system.

8.8.2.2 PssPosService(IBM 4690 POS only)

This is an NT Service specifically developed for information exchange with IBM 4680/90 POS systems running the Supermarket Application(SA). This service handles the transfer function for all of the file types listed above.

8.8.2.3 IBM RCO for 4680/90 POS

PssPosService makes use of the IBM RCO (Retail Connectivity Option) product for the 4680/90 POS. Release 1 of PSS WaveWorks used IBM RCO release 2.4.1. This product supplies a client access service that communicates with a Server application on the 4680/90 POS. For this mechanism to work, the IBM POS must be running the adxsrvnl.286 application as a background application. The client application nbqmain.exe that runs as a WindowsNT service is automatically installed by the PSS installation procedure, but it must be configured by placing configuration information into file nbqmain.cfg in the Pss\bin directory. This file is delivered with a set of preconfigured values that are typical of a PSS/4690 POS installation, but may need to be changed to allow PSS to communicate with the POS. See the IBM RCO User's Guide for parameter values which can be placed in this file.

8.8.2.4 POSInterfaceCOM

This COM object contains the software methods used to perform the processing of transaction files, item files, and tax table files described earlier. In release 1 of PSS, it also contains the transactions which initialize the PssPosService. It contains the following functions to handle the associated transaction:

Function	Transaction	Activator
LoadPOSItemFile	P2P003	PssFileMonitorSvc
LoadTaxTable	P2P005	PssFileMonitorSvc
LoadPOSTransaction	P2P002	PssFileMonitorSvc
EnableItemUpdates	P2P304	PosInterfaceCOM/LoadPOSItemFile
DisableItemUpdates	P2P404	PosInterfaceCOM/POSSItemFileTransfer
POSSItemFileTransfer	P2P103	PssFileMonitorSvc
POSstartup	POS100	PssFileMonitorSvc
SetCommLinkStatus	POS301	PssPosService
POSLogin	POS300	PssPosService

8.8.2.5 PssCustomerMaintCOM

This COM object handles processing of the Customer Information File (CUSTMTxx.DAT) and updates the PSS database tables with information from that file.

It contains the following functions:

Function	Transaction	Activator
LoadFromFile	P2P006	PssFileMonitorSvc

8.8.2.6 PssFileMaintCOM

This COM object handles general file housekeeping functions. It contains the transaction used by the POSInterface to purge files from the Processed File Area, the transaction to move the PssTransactionFile from the Transfer area after the POS has processed it, and the transaction that emulates the POS (Converts PSSTransactionFile to POSTransactionFile)

It contains the following functions:

Function	Transaction	Activator
CleanupPssTransFile	P2P201	PssFileMonitorSvc
CleanupArchiveArea	P2P298	WindowsNT Scheduler/WaveWorks SENDTX
ConvertSItoSO	P2P299	PssFileMonitorSvc

8.8.2.7 Windows NT Scheduler

PSS uses the native scheduler inWindowNT to schedule a purge of files from the ProcessedFileArea area. During PSS installation, a scheduled task named *PurgeFiles* is placed in the Windows NT \Tasks folder which is scheduled to activate transaction P2P298 daily to purge the files. The FileHouseKeeping system settings are used to control the purge activity.

8.8.3 POS Interface Configuration

Configuration of the POS Interface for a specific store environment is achieved by modification of PSS System Settings and by implementation of User Exit routines supported by the POS Interface Applications. Changes to configuration items are made using the System Settings page on the System Administration Terminal. That page contains a detailed description of each system setting item with information that identifies valid values for the item. The page also contains a default value for each item.

The settings for the POS interface are divided into the following four areas

- settings that apply subsystem-wide to the POS interface
- settings that apply to the processing of specific file types
- settings that affect the operation of the NT Services used by the POS Interface
- settings that apply to a specific POS system (IBM 4680/90).

After modifying a system setting value, the link at the bottom of the system-setting page, Re-Initialize PSS System, must be activated to cause the modified setting to be recognized by the PSS software. The PssPosService must be stopped and restarted after changing any 4POSSINTERFACE settings.

The names of the items on the System Settings Page that are used to configure the POS interface portion of PSS are listed here for cross-reference to that page. Where the following pages mention a Configuration Group, that corresponds to the Name column on the System Setting page. Where they mention a Configuration Item, that corresponds to the Subname column on the System Setting page.

Settings that Apply to the POS Interface Subsystem as a Whole

Configuration Group (Name column on System Settings Page)		
POSInterface		
	Configuration Item (Subname column on System Settings Page)	
	LocalTransferArea	Directory to monitor for the arrival of files from the POS and where PSS should place files for the POS to retrieve
	RemoteTransferArea	(Future)Location on POS where files are to be deposited or retrieved
	ProcessedFileArea	Directory where PSS should place files when done processing them
	POSControllerType	Type of POS which PSS is interfacing to (4POSS is the 4690 interface)
	POSControllerName	Network Name of the POS controller
	LanAdapterNumber	LANA number of the network interface that the POS computer is connected to
	PSSItemBarcodeLength	Number of valid digits in the item barcode field of a file exchanged with the POS. This field is pre-sized to 24 digits in the files, but most barcodes have fewer digits.
	PSSTransactionType	TICKETONLY/LOYALTY
	UsePSSPrice	Set to Yes if the POS does not place price information in the POSTransactionFile. PSS rescan logic will not do price comparisons
	ExcludedDepartments	Identifies items **not** to be loaded into the PSS database if they are associated with these department numbers
	Trace_Level	Identifies the amount of status information that should be written to the PSS System Log by the software that is part of the POSInterface system. This value may be overridden by definition of a Trace_Level for a specific POSInterface component
PSS_Global		
	LoyaltyCardLength	Number of valid digits in a customer loyalty card
FileHouseKeeping		
	HousekeepingEnabled	Whether or not File's should be deleted from the ProcessedFileArea when the FileHousekeeping job runs
	FileKeepDays	Number of days a file should remain in the ProcessedFileArea before being deleted. This is actually the number of 24 hour periods to be kept. If the File Houskeeping job is set to run more frequently than once per day, some files dated on a specific day may be deleted and others remain.
ImportItemData		
	CheckDigitExists	Indicates whether or not a check digit is part of item file barcodes
	ValidateCheckDigit	Indicates whether check digit item file barcodes are valid

Settings for Configuring Processing Attributes of Each POS File Type

Configuration File Groups	
POSItemFile	Full Item file and Item/Price updates
POSTaxFile	Tax tables
POSTransFile	Created at completion of a POS transaction (Scan-Out)
PSSTransFile	Created at completion of a PSS transaction (Scan-In)
PSSTransFileX	Defined when the IBM 4690 POS(4POSS) is being used. 4POSS renames Scan-In files that have been successfully written to the POS and leaves them in the transfer area. This file type identifies those files and specifies a transaction that moves them into the ProcessedFileArea.
POSCustomerFile	Provides updates to customer information and identifies customers who may use PSS
Configuration Item	Applies to each of the above file types (not all apply to every file type).
FileName	Name used by POS system to identify the type of file to be processed
MonitorInterval	Delay(in ms) between scans by the PssFileMonitor service to look for the existence of the file type
TransferEnabled	Identifies to the PssFileMonitor service whether or not the filetype should be processed
TransferType	Identifies the direction of transfer(Inbound or Outbound) and distinguishes IMMEDIATE vs TOD (TimeOfDay) processing. The PssFileMonitor service activates a WaveWorks TransferTX when it detects an OUTBOUND file and it initiates a WaveWorks ProcessTX when it detects an INBOUND file. If the file is an INBOUND-TOD type file, then the PssFileMonitor initiates a TransferTX at a specified time to request a POS to send a filetype.
TransferTime	Only applies to TOD file types. This identifies the time of day at which the TransferTX will be activated to retrieve the file on the POS
TransferTX	A WaveWorks transaction ID. This transaction must be defined in the WaveWorks database with a string parameter for the file path of the file to retrieve
ProcessTX	A WaveWorks transaction ID. This transaction must be defined in the WaveWorks database with a string parameter for the file path of the file to process.
PSSExpansionLength	Identifies the number of additional bytes appended to each record in a file for transferring PSS specific information. See the specific filetype layout for more information
UserExpansionLength	Identifies the number of additional bytes appended to each record in a file for transferring client specific information. PSS standard processing ignores this data, but it may be accessed by PSS User Exits
FileFormatVersion (future)	Future. Will be used to distinguish file versions if the standard file format changes in the future
RemoveItemCheckDigit (not all files)	Identifies if the ITEM barcode in the record as written by the POS includes a checkdigit. PSS does not store the checkdigits in its database

	RemoveCustCheckDigit (not all files)	Identifies if the CUSTOMER CARD number in the record as written by the POS includes a checkdigit. PSS does not store checkdigits in its database
	AddItemCheckDigit (not all files)	Identifies if PSS should write a checkdigit value in the ITEM field in the records of files sent to the POS
	AddCustCheckDigit (not all files)	Identifies if PSS should write a checkdigit value in the CUSTOMER CARD number field in the records of files sent to the POS
	FullItemFileName (POSItemFile only)	The specific file name of a Full Item Update file. This is used to distinguish replacement database from a database update to the PSS item data
	Trace_Level	Identifies the amount of status information that should be written to the PSS System Log by the software that processes the file type

Settings for Configuration of NT Services that Perform POS Interface Functions

PssFileMonitor		
	Trace_Level	Trace level to be used by the PssFileMonitor service

POS Specific Settings for the IBM4680/90 POS

4POSSInterface (Specific settings for the interface to the IBM 4680/90 SA POS system)		
	POSTransFileVersion	Whether the 4690 ScanOut file user exit writes prices to the file and whether Quickpay transactions have items written to the transaction file
	CopyTaxTables	Whether or not to retrieve tax tables from the POS
	NumberPriceBytes	Number of price bytes expected in the ScanIn file by the POS User Exits
	DeletePOSIRC	Whether/When to delete the POS Item record changes file
	Trace_Level	Specific trace level for the PssPosService

.

8.8.4 PSSTransactionFile Processing

8.8.4.1 File-Based Interface

At the conclusion of a shopping trip (when a hand held terminal is returned to the dispenser or the End of Trip barcode is scanned), a PSS transaction file is created and placed in the transfer directory. Once the file is properly placed in the transfer directory, the processing for a PSS transaction file is complete in the file-based interface. Custom code must be developed to provide the transfer mechanism to the POS system.

8.8.4.2 4680/4690 Interface

The IBM SA Personality implementation of the POS API uses the IBM product RCO. The RCO API provides access to the POS files on the POS controller over a LAN. It is described in the IBM manuals:

- "Retail Connectivity Option—Version 2.4.1 including OS/2 appendix - Product Reference" dated September 15 1994.
- "Retail Connectivity Option—Version 2.3 - User's Guide" dated November 1996, which covers both OS/2 and NT

The POS Interface consists of a service (PssPosService) that includes a modified version of the POSAPI from Version 5.0 of the PSA. The POSAPI provides the interface between the PSS Controller and the native 4680/90 POS. It provides the PSS Controller with file access to data on the native POS controller.

When a hand held scanner is returned from a shopping trip, a PSSTransactionFile, according to the format listed above, is created in the transfer directory. The PssPosService scans the transfer directory at a regular interval. On detection of a file, the PssPosService handles transfer of the information in the transaction file to the 4690 controller by writing the transaction information into the Scan-In file on the POS. The frequency at which the PssPosService scans the directory is specified by the system setting PSStransactionFile/MonitorInterval.

8.8.4.3 POS Emulation (Demo System)

Under normal conditions, the PSS shopping trip creates PssTransaction files and places them in the transfer area for a POS to process. The PssFileMonitor process is configured to ignore these files and not attempt to move them or process them, since the store POS (PssPosService) will be processing them. For testing and demonstration purposes, when access to a POS system is unavailable, the System Setting values for the PSSTransactionFile can be modified to cause the PssFileMonitor to detect them and activate a POS emulation transaction. This transaction reads PSSTransaction files (Scan-In) and creates corresponding POSTransaction files (ScanOut) which are in turn detected and processed to complete the full shopping transaction cycle and unsuspend the customer record (customers are suspended until trips are completed).

In order to enable this feature, the following system settings must be configured as shown.

Name/Subname	Value
PSSTransFile/TransferEnabled	**Y** (Tells PssFileMonitor to process this filetype - default is N)
PSSTransFile/TransferTX	**P2P299** (the PosEmulation transaction -default is P2P199)
PSSTransFile/AddCustCheckDigit	**N** (should match value for POSTransFile/RemoveCustChkdigit)
PSSTransFile/AddCustItemDigit	**Y** (should match value for POSTransFile/RemoveItemChkdigit)

After changing these settings, stop both the PosService and PssFileMonitor services. Set the WaveWorks database ADM_DAEMON table so that PosService is NOT restarted automatically (just prefix the name so the service isn't found). This will prevent the PosService process from trying to process PssTransaction(ScanIn) files.

Start up the PssFileMonitor service. By setting the values above, PssFileMonitor will be enabled to monitor PSSTransaction(ScanIn) files and trigger event P2P299, which creates a corresponding POSTransaction(ScanOut) file. Normal POSTransaction file processing will then be triggered by the appearance of a file and cause the shopping trip to be completed.

The POSTransaction file that is created has the same RESCAN flag value and the same items as are in the PSSTransaction file. Future support for record level and file level user exits to be called by transaction P2P299 is planned. Thus the created POSTransaction file could be altered by integrators to add to, remove, or alter records. This support is **not** available in the initial release.

8.8.5 POS Transaction File Processing

8.8.5.1 File-Based Interface

The transfer directory is polled by the PssFileMonitor service. Once a POSTransactionFile arrives in the transfer directory, file processing is activated. The shopping trip information is moved into the Shopping History database tables. In addition, the shopper's rescan probability is recalculated. If the file contains an End of Day record, the End of Day Event is stored in the database (PSS_EOD table) and the End of Day user exit routine is activated.

8.8.5.2 4690 Interface

Upon completion of a transaction at the store check-out lane, a 4690 user exit running in the check out support application updates a ScanOut file on the POS with information about the completed transaction. The PssPosService polls the file on the POS, retrieves new information in the scan-out file, and creates a PosTransactionFile in the Transfer Area for each POS transaction completed since the last scan. Upon detection of a POS transaction file by the PssFileMonitor service, processing continues as described in the file-based interface. The frequency of the poll interval is specified by system setting POSTransactionFile/MonitorInterval.

8.8.6 Item Record File Processing

PSS maintains a database of items on the store shelves and their prices. This database is maintained by downloading the item information from the store's POS system. These downloads can be achieved as either full item database downloads or as item information updates.

The POSItemFile system settings are used to control the functioning of these operations. Either or both download types can be used to update the PSS item database. Since a download of a full item file requires a high level of computer system resources it is recommended that full item database downloads be performed when shopping activity is low. However, shopping trip activities may continue during Item File processing and full downloads or updates can occur at any time. Early benchmarks indicate approximately 1500-4000 records per minute can be loaded depending on other PSS system activities, load on the POS system, overall traffic load on the PSS-POS network segment, machine configuration (CPU speed and amount of memory) and on the database size.

There is one primary distinction between item update and full item file processing, item deletion. During full item file processing PSS tags all items in the database that were also in the item file and after finishing its update pass it deletes all untagged items. Since item update processing does not include an item delete function, this is the only way to purge PSS of items that have been deleted from the POS database.

8.8.6.1 File-Based Interface

The transfer directory is polled by the PSSFileMonitor service. When a Full Item File or Item/Price Update file arrives in the transfer directory, the Item File processing is initiated. File processing takes the data contained in each Item File record and loads that data into the PSS_Shopping_Item table in the PSS Database.

8.8.6.2 4690 Interface

The PssPosService creates an Item update or full Item file from the information maintained on the 4690 controller. On startup, the PSSFileMonitor service sends a WaveWorks transaction to the PssPosService requesting that it initiate Item/Price update processing. The PosService then begins polling the POS controller at the rate specified by the configuration item POSItemFile/MonitorInterval for changes. The PSSFileMonitor uses the configuration item POSItemFile/TransferTime to control when to request download of a full item file to the PSS system. When that time arrives, the PSSFileMonitor sends a pair of WaveWorks transactions to the PosService, one to disable item/price update polling and one to request creation of the full item file. On arrival of the item file in the transfer directory the PSSFileMonitor triggers the WaveWorks ItemFile load transaction. Upon completion of loading the file, the ItemFile load module sends a WaveWorks transaction to the PosService to tell it to resume polling the POS system for updates.

8.8.7 Customer Update File Processing

The information maintained for customers that are registered to use the PSS system may be updated by sending a Customer Update File to the PSS system. This file can be used to create and update customer records in the PSS Customer database. PSS maintains many data items not included in the CUSTMTxx.DAT file described earlier in this document, but the format has been maintained for compatibility with existing PSA systems. Sites, which choose to populate additional PSS customer fields, can append data to the customer record in the User Expansion Area of the record and can process that data by customizing the User Exits provided.

8.8.7.1 File-Based Interface

The transfer directory is polled by the PSSFileMonitor service. When a Customer Update file arrives in the transfer directory, the file processing transaction is activated. File processing takes the data contained in each Customer Update record and loads that data into the PSS_Customer table in the PSS Database.

8.8.7.2 4690 Interface

There is no special support provided with the base PSS system for creation and transfer of customer files.

8.9 PSS Services

Services are software programs that control individual components of the Portable Shopping System (there are also services concerned with the operating system and other components of your computer). The common services associated with the Portable Shopping System are defined below; your store may have additional custom services not listed here.

WaveWorks RF Server—handles all communications to RF devices (primarily the hand held scanners, and certain types of entrance units)

WaveWorks Message Server—routes messages between components of the system

WaveWorks Business Server—manages software applications (COM Objects)

WaveWorks STEP/Display Server—displays the screens on the hand held scanners

WaveWorks System Controller—the highest level service which encompasses all other services defined for your installation of the PSS

PSS POS Service—handles communications between the PSS and the IBM 4600 series of POS controllers

PSS File Monitoring Service—manages file transfers between PSS and POS (primarily for transfer of item, customer, scan in, and scan out files

PSS RF Entrance Service—controls operation of all entrance units

PSS Printer Service—controls operation of all printers

9. Customizing the PSS System Software

The WaveWorks PSS system offers a number of options for integrators that need to customize the base system to meet customer requirements. PSS allows integrators to customize the system by changing the DisplayServer scripts, writing custom services, or writing custom business objects. PSS also supports "user exits," which are function hooks in the base code that let the integrator provide custom processing at key points in the code.

9.1 Custom DisplayServer Scripts

The DisplayServer process runs scripts called "interfaces" that control the display on RF devices (terminals and RF entrance units). The code for the interfaces is stored in the WaveWorks database. The source files that were used to load the database at install time are found (by default) in the C:\PSS\StepDev folder. The integrator can make a customer-specific copy of these source files and modify them as needed. The modifications can range from small changes to the logic, such as running a custom transaction or changing the screen layout, to completely rewriting the interface. Note that most interfaces get their text from the PSS database, so changing the wording of prompt or message may be as simple as changing the text in the PSS_Text table.

9.2 Custom Services

The PSS system uses NT Services (standalone processes that are started at bootup) to communicate to hardware devices. Examples include the EhtService for communicating with the RF entrance unit, PssPrinterXX for talking to serial ticket printers, and the PssPosService for talking to the store's POS system. Writing an NT service requires a solid programming background and WaveWorks developer training, but is an option for providing an interface to special devices or computer systems. The services connect to the WaveWorks MessageServer and use WaveWorks transactions to access standard business services.

9.3 Custom Business Objects

PSS business services are contained in more than a dozen COM Objects. Each business service is a COM object method. The WaveWorks system uses transaction Ids to map business services to a particular method on a particular business object. Integrators don't have access to the base system source code (aside from user exits that are implemented in the COM Objects), but they can create their own COM Objects. These COM Objects can provide new business services (using new transaction Ids) or replace existing business services (by changing the mapping between transaction ID and COM object/method in the WaveWorks database). These new business services can be accessed from custom services and custom DisplayServer interfaces.

Writing custom COM Objects requires a solid programming background, preferably using Visual C++ and the ATL wizard.

9.4 User Exit DLL

A DLL, PssUserExits.dll, is provided which allows integration teams to customize the software for a given installation. **This mechanism is equivalent to the User Exits used in the PSA software.** The DLL contains functions that have been built into the code, but that, in most cases, contain stubs that only log a debug level message to identify that the function was called. The integrator is then allowed to add appropriate code to the functions and rebuild the dll.

The C++ language header file PssUserExits.h documents the intended purpose of each function, the detail of the function interfaces, and the return status codes expected by the base PSS software. It also documents how the various return code values will affect the operation of the base PSS software when the function returns.

Each User Exit function is implemented in a separate C++ source file named <UserExitName>.cpp. This C++ source file contains the stub code for the function.

9.4.1 Common Information

All methods should return one of three return status codes as described below. The return code is the last argument in the call list to the User Exit function.

The following description of standard arguments which are passed to all user exits applies to C++ modules only. User Exits called from "C" only code will not include the arguments which are C++ objects.

A C++ source module with the same name as the function name is supplied with a stub implementation routine for the User Exit. The stub routine "includes" the PssUserExits.h file to retrieve the function definition. All stubs return status UE_CONTINUE to the calling routine.

Many of the stub routines log a **debug** level message to the PSS System Log with the values of all the input arguments. This stub code can be reviewed to see how to use the CPssMsgLog object to write messages to the PSS System Log. See also the following section "Using the CPssMsgLog Class."

If a "Preprocess" User Exit needs to pass information to a "Postprocess" User Exit for the same PSS function (PreProcessRecord -> PostProcessRecord), it can be maintained in **static** storage in the dll. All other information exchange between user exits cannot safely rely on executing within the same instance of the dll. In those cases the User Exits must use other mechanisms for sharing information between user exit functions.

PSS base code is linked with Microsoft's Multithreaded DLL runtime library, so integrators **must** link a custom user exit DLL using that same library. Otherwise runtime errors will occur with the CpssMsgLog and CSymStatementPool classes that are used in the stub User Exits.

9.4.2 Return and Message Codes

The following return codes are used by the standard PSS software to determine how to proceed after the User Exit routine has returned to the calling routine. The name and numeric value must also be defined in database table PSS_TEXT if this name is to be used as the msgcode field for writing a message to the PSS System Log.

UE_CONTINUE	Indicates PSS should continue with normal processing
UE_DONE	Indicates the User Exit has performed all necessary processing and PSS should skip its normal processing. **Note:** A UE_DONE returned by a "PreProcess" function will cause PSS to skip the call for the "PostProcess")
UE_ERROR	Indicates the User Exit was unable to process sucessfully PSS processing will write a message to the log and continue processing in the same manner as if a UE_CONTINUE was returned unless documented otherwise for the individual User Exit.

9.4.3 Using the CPssMsgLog Class

All of the PSS software uses a C++ class named CPssMsgLog for writing operational status and error messages to the PSS System Log. Customized User Exits should make use of this class to keep system logging consistent for troubleshooting purposes. Each line in the PSS System Log displays the following:

- Date/time the message was written.
- Facility Name—a 64-character name which should identify the COM object, NT Service, or other high-level entity such as "ShoppingTripEndOfTrip" to identify a transaction type.
- Action Name—a 64-character name which represents the specific function which is logging the message. This should either be the UserExit name or the name of a specific subfunction used by the UserExit routine.
- Message text—a 256-character string that contains the information being reported. The text of these messages is maintained in the PSS_Text database table. The text in that table can contain tags that represent values which the software supplies to replace the tag. For instance, the POS file processing replaces a string tag with the filename of the file being processed and the Unit Management system replaces a string tag with the MAC address of a terminal that doesn't respond.

- Severity—represents the severity of the condition being logged. A System setting value, subname "Trace_Level", is maintained for the Facilities that log messages. If the message severity value is less than the severity value specified in the LogMessage call, the message is **not** written to the log. For instance, if the Trace_Level is set to 2 (Warning), then messages with a severity value 1 (Informational) are **not** written to the log, but messages logged with the severity 3 (Error) **are** written to the log. The value 2 is the default Trace_Level value set for most facilities by the PSS installation scripts. The Trace_Level values can be modified using the System Settings Page on the PSS Administration terminal.

The allowable values for severity which can be specified in a LogMessage call are:

- PML_DEBUG—level 0, used to log messages helpful in debugging code during development. Examples are logging the values of parameters used or calculated by software. The default Trace_Level settings suppress these messages.
- PML_INFO—level 1, informational messages that typically log the operational status of software. Examples are positive status about the completion of an operation (Function XYX completed successfully). These are messages that would not normally be enabled in the System Log because there would be many such messages cluttering the log and they would obscure more severe messages. If there were a perceived system problem, then troubleshooting procedures might suggest setting the Trace_Level to 1 so these messages would be written to the log and could be reviewed to see if software functions were being performed as expected.
- PML_WARNING—level 2, Error conditions which the PSS software is readily able to recover from. The default Trace_Level settings write these messages to the log. These messages may or may not indicate a system malfunction.
- PML_ERROR—level 3, Error conditions that indicate a system malfunction that is causing operational errors. The default Trace_Level settings write these messages to the log. Some corrective action will normally need to be taken to reinstate proper system operation.
- PML_FATAL—level 4, Critical Error conditions that stop the operation of PSS system components. These messages typically accompany other indications such as turning the PSS Administration terminal's Main Menu traffic light red and the disabling of the ability of PSS to perform shopping activities.
- PML_ALWAYS—level 10, Status messages that do **not** indicate error conditions, but which should always be written to the System Log. Examples are startup and shutdown messages logged by PSS services or activation messages logged by scheduled tasks.

See the PssMsgLog.h header file for more programming details.

9.5 Available User Exit Functions

Following is a complete list of the User Exit routines as called by the standard PSS Software, along with a brief description of the intended purpose of the User Exit routine and a description of how the base PSS software reacts to the return code received from the User Exit. For the most recent information on functionality and more details about specific arguments passed to the User Exit routines, review the header file PssUserExits.h.

9.5.1 UE_PreProcessItemFile

This function is called **before** processing of Item Record File is performed. It receives the file path of the item file as an argument.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the item record file as if no Pre-Processing was performed
UE_DONE:	PSS performs **no** Item record file processing
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

9.5.2 UE_PreProcessItemRecord

This function is called **after** an item record is read from the file, but **before** PSS performs any processing of the record. It is anticipated that this UE function will be used to either replace all PSS Item record processing or to alter the input buffer that PSS processes.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the item record as if no Pre-Processing was performed, validating the item record and loading it into the PSS database.
UE_DONE:	PSS performs **no** record processing and proceeds to read the next record
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

Note: If a UE_DONE is returned, the POS_Item_Flag field of the record in table PSS_Item **must** have been updated to "Y" or the record for that item will be deleted from the database by PSS end of file processing.

9.5.3 UE_PostProcessItemRecord

This function is called **after** the item record has been processed by PSS and loaded into the PSS database. It is anticipated that this User Exit function will be used to alter the PSS database item values set by default PSS processing or to process additional information in the User Expansion area of the item record.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS proceeds to read the next record
UE_DONE:	PSS proceeds to read the next record
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

9.5.4 UE_PostProcessItemFile

This function is called after PSS has processed the entire Item Record File. All database updates have been completed and the file is closed. It receives the file path of the item file as an argument. The file path for the processed Item record file will be different than the name the file had for UE_PreProcessItemFile and depends on the success of PSS processing.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS performs no additional file processing
UE_DONE:	PSS performs no additional file processing
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

9.5.5 UE_PreProcessTaxFile

This function is called **before** any processing of a POS Tax Table File is performed. It receives the path of the tax table file as an argument.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the file as if no Pre-Processing was performed
UE_DONE:	PSS performs NO file processing
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

9.5.6 UE_PostProcessTaxFile

This function is called after PSS has processed the entire Tax Table File, all updates to thePSS database have been completed, and the file has been closed. It receives the file path of the tax table file as an argument. The file path for the processed file will be different than the name the file had for UE_PreProcessTaxFile and depends on the success of PSS processing.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the file as if no Pre-Processing was performed
UE_DONE:	PSS performs **no** file processing
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

9.5.7 UE_PreProcessPOSTransFile

This function is called **before** any processing of a POS Transaction File is performed.

The POS Transaction file defines a completed transaction at the POS terminal. Arrival of the file indicates completion of the shopping trip. Base logic clears the trip from the list of current shopping trips and moves the trip into shopping trip history. The information in the file is used to calculate a shopper's **rescan** level

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the file as if no preprocessing was performed
UE_DONE:	PSS performs NO file processing
UE_ERROR:	PSS logs an error then proceeds as with UE_DONE

9.5.8 UE_PreProcessPOSTransRecord

This function is called **before** processing the current header or detail record read from a POS Transaction File.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the RECORD as if no Pre-Processing was performed
UE_DONE:	PSS performs NO RECORD processing and proceeds to the next record
UE_ERROR:	PSS logs the error an proceeds as with UE_CONTINUE.

9.5.9 UE_PostProcessPOSTransRecord

This function is called **after** processing the current header or detail record read from a POS Transaction File and after updates to the PSS database for the current record have been completed.

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	PSS logs the status and continues on to the next record

9.5.10 UE_ProcessEODRecord

This function is called **after** PSS processes an EOD or EOW record in a POS Transaction File

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	PSS logs the status and continues on to the next record

9.5.11 UE_PostProcessPOSTransFile

This function is called **after** all processing of a POS Transaction File and all updates to the PSS database for a POS transaction have been completed.

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	PSS logs the status

9.5.12 UE_PreProcessPSSTransFile

This function is called **before** processing of a PSS Transaction File is performed. The PSS Transaction file defines a completed by a shopper using a hand held terminal. This is the file used by the POS to retrieve the PSS shopping trip.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the FILE as if no Pre-Processing was performed
UE_DONE:	PSS logs the return status and performs NO FILE processing
UE_ERROR:	PSS logs the return status and performs NO FILE processing

9.5.13 UE_PreProcessPSSTransRecord

This function is called **before** writing the current header or detail record to a PSS Transaction File.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE:	PSS processes the RECORD as if no preprocessing was performed
UE_DONE:	PSS performs NO RECORD processing and proceeds to the next record
2UE_ERROR:	PSS logs the error an proceeds as with UE_ DONE

9.5.14 UE_PostProcessPSSTransFile

This function is called **after** all processing of a PSS Transaction File has been completed and the file has been closed. The file path to the processed file is passed as an argument.

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	PSS logs the status

9.5.15 UE_CalculateItemPrice_Method10to20

9.5.16 UE_RescanLevelCalculation

This function is called prior to the standard rescan level recalculation and is intended to replace it with the user-defined algorithm.

The base PSS software handles the following return codes from the User Exit as follows:

9.5.17 UE_Pre-RescanDetermination

This function is called prior to the standard rescan determination algorithm and could be

used to either replace or modify the inputs into the algorithm.

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	

9.5.18 UE_PostRescanDetermination

This function is called after the standard rescan determination algorithm and could be used to modify the result of the calculation.

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	

9.5.19 UE_PreProcessMarketingMessage

This function is called prior to the standard marketing message method to allow different messages to be sent based upon different criteria.

The base PSS software handles the following return codes from the User Exit as follows:

All return codes:	

9.5.20 UE_LoginAuthorization

This function is called after a user fills out the username and password fields on the Administration Terminal login screen. It receives the username and password as arguments and must return as status indicating if the user is authorized.

The base PSS software handles the following return codes from the User Exit as follows:

UE_CONTINUE	PSS performs its default authentication
UE_DONE	PSS proceeds considering the user is authorized
UE_ERROR	PSS proceeds considering the user is not authorized

10. System Administration Interface

The System Administration Interface provides browser-based access into the data contained in the system database, as well as various system control functions. The purpose is to allow store personnel to manage the system, obtain information, and provide updates to the information.

The available data concerns almost every facet of the system, including:

- Customer current and historical data
- Messaging
- Hardware configuration
- Item data and marketing
- Reports
- System Status
- System Configuration

10.1 Configuring the Browser Software

Install the desired browser software on the Service Terminal. The recommended browser software packages are:

- Netscape Navigator™ (Netscape Communications Corporation), v4.08 or greater
- Microsoft Internet Explorer™ (Microsoft Corporation), v5.0 or greater

Create a shortcut icon on the desktop for the browser, for ease of access by store personnel.

Set the PSS Administration Main Menu screen as the "HOME" page of your browser software. The home page will then be automatically displayed when the browser software is started. The path to the main menu is: http://*localhost*/pssinterface/administer.asp, where *localhost* is either the literal text string 'localhost' or the name or IP address of the Service Terminal computer itself.

10.2 Establish User Accounts on the Service Terminal

A browser screen is provided to create employees accounts and control employee access to the Service Terminal. To access the screen, from the PSS Main Menu, select System / Settings / Configuration / Administration Users. The Administration Users screen, which is ordered alphabetically by user names, appears. Related user data includes:

- Full name of the user.

- Valid starting and expiration dates and times for user's access to the PSS.
- Privilege level—security level to which the user is assigned. This controls the functions that are displayed (as described below). These levels, from highest to lowest, are:
  - Administrator—has the same functions as all others, plus additional functions.
  - Technician—has the same functions as Manager and Customer Service, plus some additional functions.
  - Manager—has the same functions as Customer Service, plus some additional functions.
  - Customer Service—has the fewest functions.
- The language spoken or preferred by this administrative user. If a particular text string does not exist for the language preferred by the user, the store language, or, as a last resort, US English will be used.
- All data concerning user accounts is stored in the PSS_Admin_User database table.

10.3 Understanding User Access to Administrative Pages and Actions

For each security level (Administrator, Technician, Manager, Customer Service) of user accounts, there is default access established to Administrative Pages (which are browser pages) and to Administrative Page Actions (which are distinct tasks contained on browser pages). This allows for a high degree of flexibility in controlling user access.

Two browser screens are provided to maintain these Administrative Pages and Page Actions.

To access the Administrative Pages screen, from the PSS Main Menu, select System / Settings / Configuration / Admin Pages. The Administration Pages screen, which is ordered alphabetically by page name, is displayed. Related data includes:

- Page description
- Minimum user privilege level
- Whether or not authenticated login is required
- Whether or not the page is disabled

All data concerning administrative pages is stored in the PSS_Admin_Page database table.

To access the Administrative Actions screen, from the PSS Main Menu, select System / Settings / Configuration / Admin Actions. The Administration Actions screen is displayed. Related data includes:

- Description of the action
- Minimum user privilege level
- Whether or not the action is disabled

All data concerning administrative actions is stored in the PSS_Admin_Action database table.

The default permission settings for pages and page actions should be sufficient for providing appropriate system access to each level of user. However, differing customer requirements may make modifications to these default settings necessary.

In general, raising the authentication levels to make features available to fewer users is not a problem, but care should be taken to make sure that each level of user has access to all of the features necessary for their use of the system. For example, a customer service user should always have access to customer-related functions—not only to the obvious lookupuser.asp (Find Customer) and registeruser.asp (Add Customer) pages--but also to the support pages for customer functions, such as issueterminal.asp (Issue Terminal) and unabletoissue.asp (Unable to Issue a Terminal). The same rule applies for privileges on actions.

Care be taken that the combinations make sense; for example, requiring higher privilege levels for viewing than for modifying a customer name would be largely ineffective.

Lowering the authentication levels to make features available to more users should be done with extreme caution. In some cases, lowering authentication levels can, in effect, remove all security from the system (i.e., lowering the privileges on the Administrative Pages or Page Actions pages). Lowering the authentication levels on particular actions could also cause problems. For example, lowering the Modify Customer Rescan Level or Modify Gold Customer action could allow any customer service user to set customers levels so that they would never be rescanned, regardless of previous rescan results. The Suspend / Unsuspend Customer actions could also be abused similarly.

10.4 Changing Screen Text on the Service Terminal

The text that appears on the Service Terminal is stored in the PSS_Text database table. The Service Terminal uses Text_IDs from 20000 to 30000 and is organized as follows:

Text_ID	**Contents**
20001 - 20099	Menu Titles—Text appearing in the menu buttons in the Service Terminal menu system
20101 - 20199	Menu Long Names—Text appearing in the pop-up “alternate text” descriptions for the menu buttons in the Service Terminal menu system
20201 - 20299	Page Titles—Text appearing as the page title in the browser title bar
20301 - 20399	Page Short Names—Text appearing as the page title at the top of the page, in the navigation bar, and in the page buttons in the Service Terminal menu system
20401 - 20499	Page Descriptions—Text appearing as pop-up “alternate text” descriptions for the page buttons in the Service Terminal menu system.
20501 - 20599	Action Descriptions—Text appearing as action Descriptions on the Admin Actions page
20601-20699	Page State Titles—Text appearing as the page title in the browser title bar for pages with multiple “states,” for example, the Add Customer page also performs Modify Customer activities.

20701 - 20799	Page State Short Names—Text appearing as the page title at the top of the page, in the navigation bar, and in the page buttons in the Service Terminal menu system for pages with multiple "states."
20801 - 20899	Page State Descriptions—Text appearing as pop-up "alternate text"descriptions for the page buttons in the Service Terminal menu system for pages with multiple "states."
20901 - 20999	Miscellaneous Menu Text—Text used for particular purposes in the Service Terminal menu system, for example, the text for logging in and out of the system is stored here.
21000 - 21099	Miscellaneous Text—Text used for particular purposes throughout the Service Terminal, for example, the strings for months of the year, or days of the week are stored here.
21101 - 29999	Page Text—Text for each page in the Service Terminal is stored in this region. The formula for determining the range for a particular page is (21000 + (Current Page ID x 100)). Page Ids are all stored in the PSS_Admin_Page table. For example, page 51 is the Admin Actions page, and its text region is 21000 + (51 x 100) = 26100 (to 26199).

10.4.1 Service Terminal System Settings

The behavior of the Service Terminal can also be modified by a collection of system settings, with Name "SVCTERMINAL." These settings control:

- The use of various features or fields that may not be used at every installation for display or in calculating the system status
- The dimensions of hand-held terminal messaging displays
- Service Terminal hardware settings
- The length of time before user login sessions time out
- Transactions for resetting system software components

The PSSSysInit settings provide the transactions for resetting system COM Objects through WaveWorks. On the System Settings administrative page, there is a link at the bottom of the page, Re-Initialize PSS System. When this link is selected, a CMS message "PSS998" is sent to all standalone executables, such as services, subscribed as "PSSSYS." In addition, all system settings with Name "SVCTERMINAL" and Subname "PSSSysInitX" (where X is a custom string appended to the Subname) are traversed. The value is sent as a *multicasted* transaction, that is, sent to all instances of a COM object that handles that particular transaction.

Additional transactions may be added to the system by adding system settings with Name "SVCTERMINAL," Subname "PSSSysInitXXX," where XXX is a custom string appended to the Subname. For custom transactions, it is recommended that letters be used for the custom string (e.g., A - Z) to avoid conflicting with future additions to the PSS system, which will use numbers.

Tx ID: PST105

Class: PssDbCleanupCOM

Method: PerformDBCleanup

Description: *PerformDBCleanup* performs a number of operations to reduce the number of rows in various database tables, thus improving system performance and eliminating obsolete or unnecessary data. Its operation is configurable based on entries in the system settings with name "PSSDBCleanup." The configuration is read when the COM object is initialized, and since this routine typically runs only once per night, it should happen each time that PerformDBCleanup is run.

The first step is the cleaning of the PSS_Message_Log table. All records in the table that are older than the configured number of hours *and* are not required for system reporting are deleted from the table. Currently, the only message log entries used for system reporting are those with facility 'Unit Management' and action 'TerminalCount', 'OperationalCount', 'TerminalIssued', or 'TerminalReturned'. Actions required for reporting are kept for 13 months, which is the current limit for the system reports.

Next, the PSS_POS_Status table is processed. All records with a Start_Time or Detection_Time older than the configured number of hours for each type is deleted from the table, except for the records with Item_Name 'POSControllerName', for which there is always only one record for each POS controller in use.

Finally, the shopping trip data is processed. This includes both the current shopping list data from PSS_Shopping_List, and the historical data from both PSS_Shopping_History_List and PSS_Shopping_Incomplete_List.

For current shopping lists, any shopping list data with a Time_Started older than the configured number of hours is deleted. Shopping list data includes all associated items (PSS_Shopping_Item), activities (PSS_Shopping_Activity), tax data (PSS_Shopping_Tax), marketing data (PSS_Marketing_Sent), exception items (PSS_Unknown_Item), and quick order list (PSS_Order_List) and item (PSS_Order_List_Item) data. Exception item data for which barcode was scanned, terminal used, and time of scan is kept in the PSS_Unknown_Item table, but the link to the customer shopping list is removed.

Parameters:

Input:

none

Output:

rc	long integer	indicates whether the call succeeded

ReturnCode Values:

PSS_SUCCESS
DB_CONNECT_ERROR
DB_OPEN_ERROR
DB_EXECUTE_ERROR
DB_FETCH_ERROR
DB_NO_DATA

11. Specific Features

11.1 Multiple Language Support

Support for multiple languages is provided for all screens displayed on the hand held terminals, all screens displayed on the entrance units, and all PSS log messages. All language sensitive text is held in the PSS_Text table in the database. The keys for each text string are a Text ID and a Language ID. Based on the key, the string in the proper language is returned.

The Active Server Pages used for the System Administration screens will need to be rewritten in the store's preferred language, if the store's preferred language is other than English.

11.2 Pricing Methods

See Appendix C.

11.3 Currency Conversions

The forumla for converting currencies is as follows:

Currency Y amount = (Exchange_Rate Y / Exchange_Rate X) * Currency X amount

Where each Exchange_Rate value is read from the PSS_Currency table for the particular currencies in use. Exchange_Rates for all currencies should be in the same base amount, that is, a single currency should be chosen as having an exchange rate of 1, and all other currencies should be entered as values relative to that currency. For example, if US Dollars is chosen as having an Exchange_Rate of 1, Canadian Dollars may have an Exchange_Rate of 1.47. To convert 2 Canadian Dollars to US Dollars, the formula is therefore:

US Dollar amount = (1 / 1.47) * 2 = 1.36 US Dollars

11.4 Control Ticket Printing

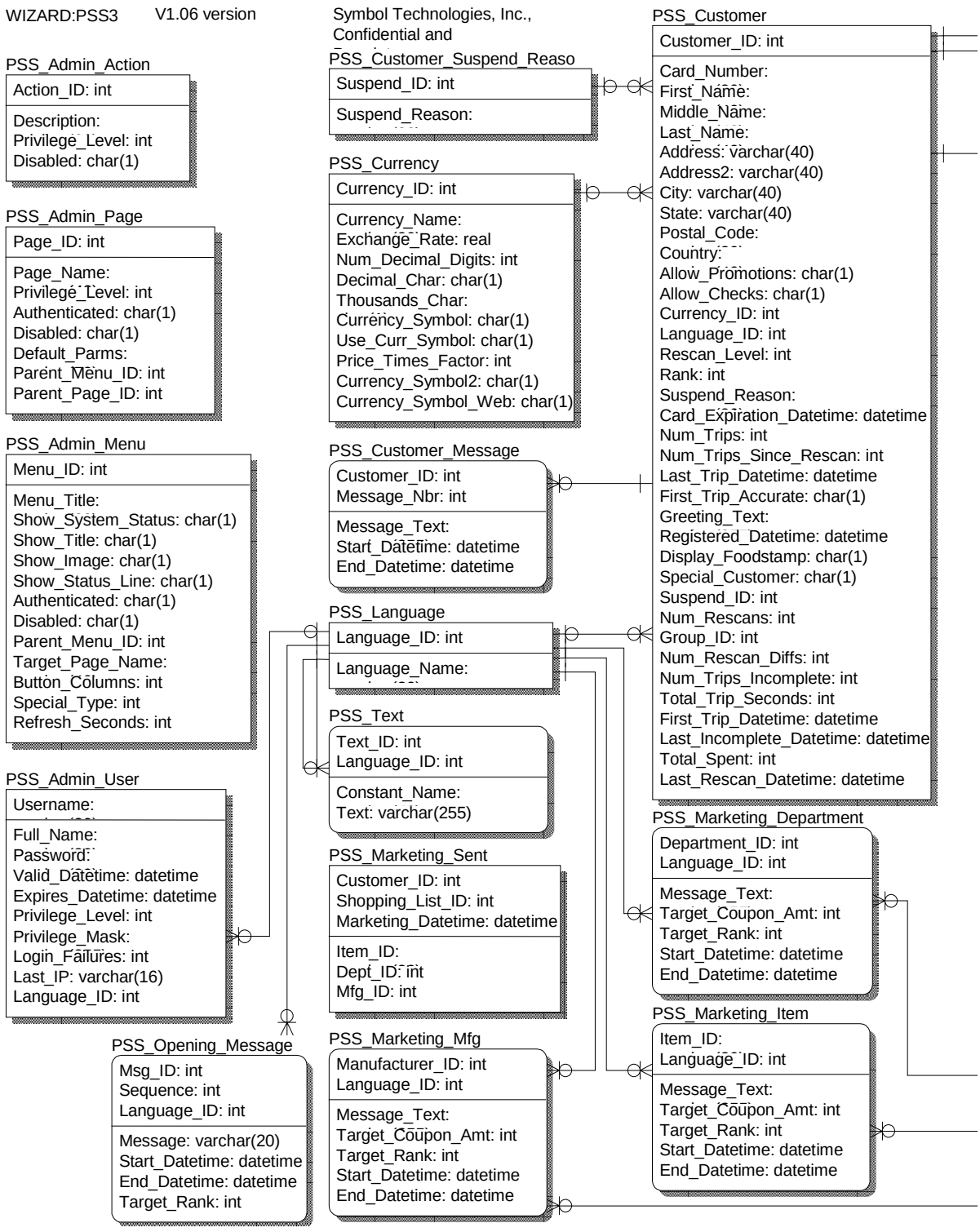
The text that appears on the PSS shopping trip transaction ticket is stored in the PSS_Text table in the PSS database. The Service Terminal uses Text_IDs starting at 7000.

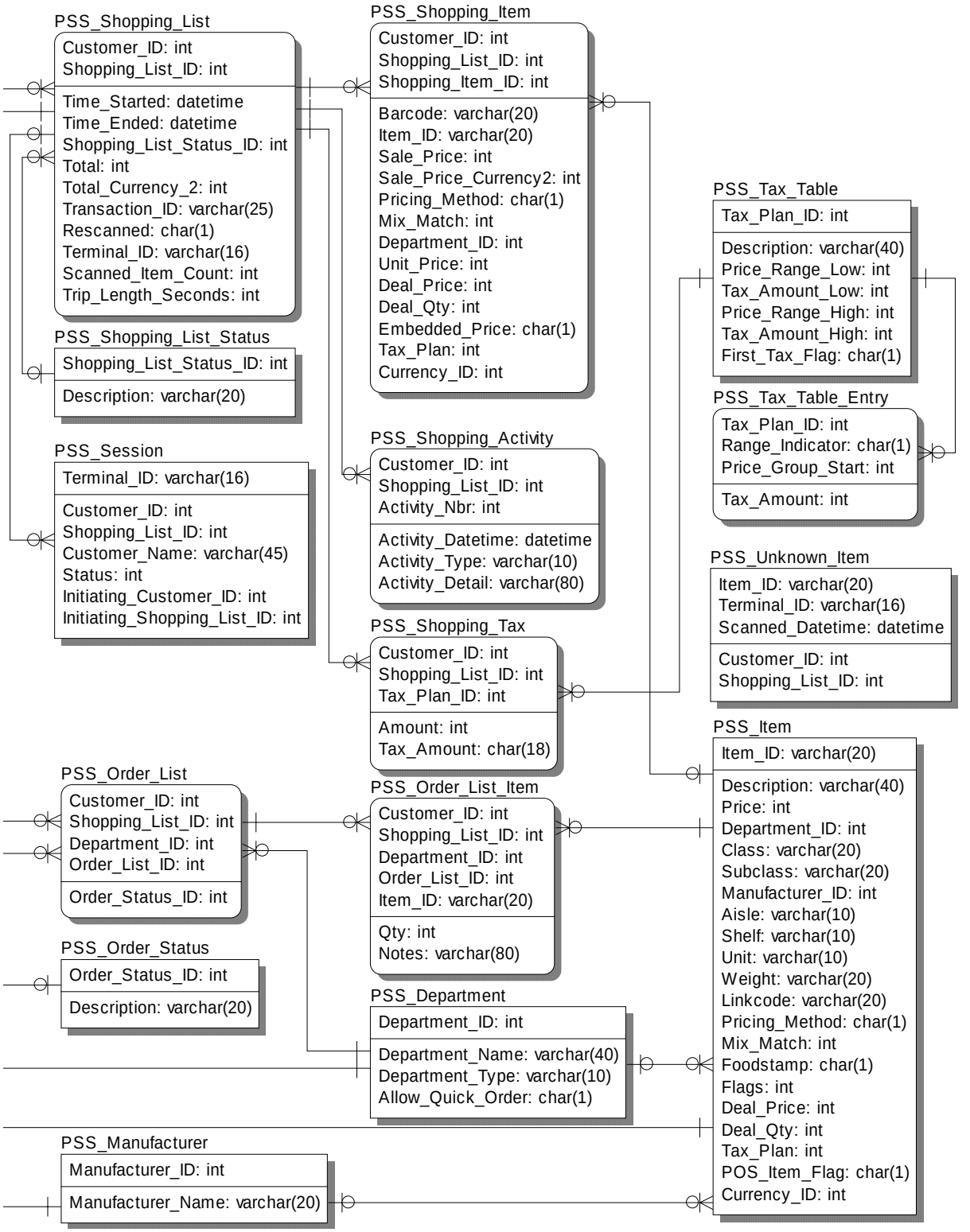
The contents of the transaction ticket are configurable based on system settings in PSS_System_Setting with a name of "TRANSTICKET."

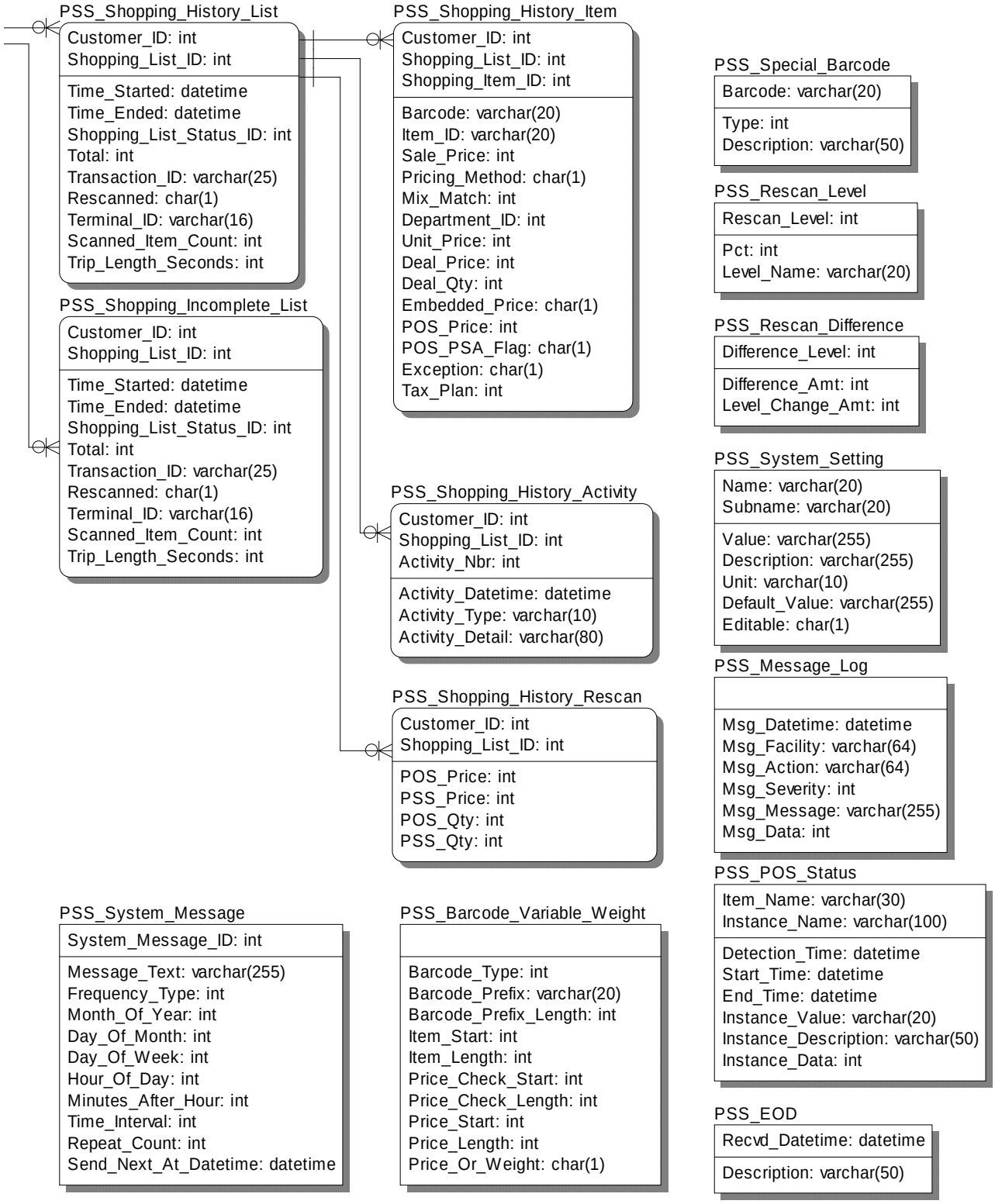
There is also a printing user exit, UE_PrintReceipt, which is called after the Transaction Ticket COM object creates a template print file. The template print file is sent to the Printer Service to be interpreted to literal print commands for a particular printer.

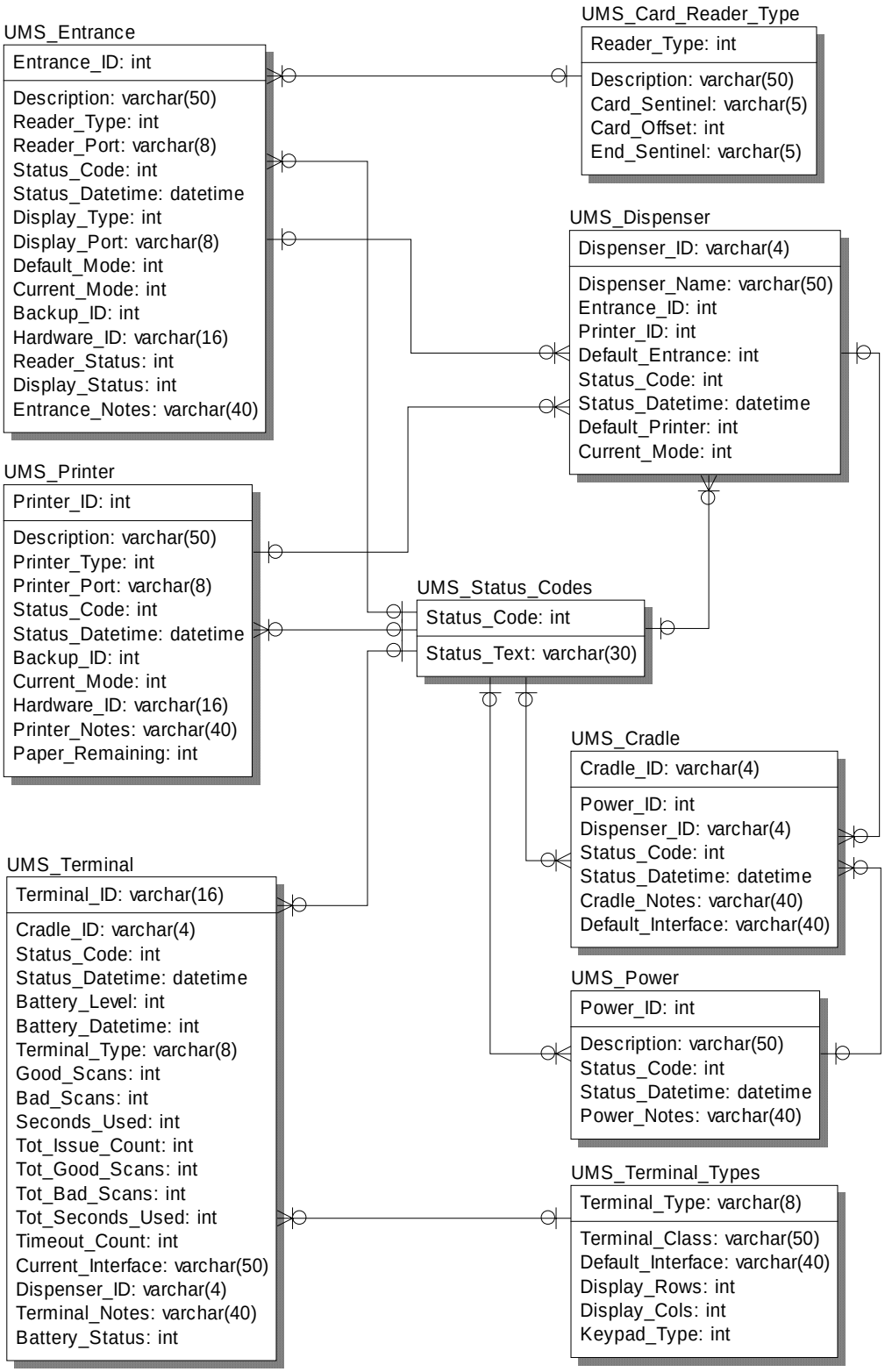
Appendix A Database Layout Diagram

The following diagram represents the database layout for the WaveWorks PSS system.









Appendix B Database Tables / Physical Properties

B.1 Table Name: PSS_Admin_Action

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Action_ID	int	NOT NULL	Yes	No
Description	varchar(60)	NULL	No	No
Privilege_Level	int	NULL	No	No
Disabled	char(1)	NULL	No	No

PSS Admin Action Table holds privilege information for various actions available through the service terminal

Action_ID unique ID associated with this Administrative Action
Description description of this Administrative Action
Privilege_Level privilege level associated with this Administrative Action
Disabled indicates whether this Administrative Action is disabled or not

B.2 Table Name: PSS_Admin_Menu

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Menu_ID	int	NOT NULL	Yes	No
Menu_Title	varchar(30)	NULL	No	No
Show_System_Status	char(1)	NULL	No	No
Show_Title	char(1)	NULL	No	No
Show_Image	char(1)	NULL	No	No
Show_Status_Line	char(1)	NULL	No	No
Authenticated	char(1)	NULL	No	No
Disabled	char(1)	NULL	No	No
Parent_Menu_ID	int	NULL	No	No
Target_Page_Name	varchar(40)	NULL	No	No
Button_Columns	int	NULL	No	No
Special_Type	int	NULL	No	No
Refresh_Seconds	int	NULL	No	No

PSS Admin Menu Table holds configuration information for the service terminal menu system

Menu_ID unique ID associated with this Administrative Menu entry
Menu_Title title of this Administrative Menu entry
Show_System_Status indicates whether the system status displays on this Administrative Menu page or not
Show_Title indicates whether the menu title displays on this Administrative Menu page or not
Show_Image indicates whether the PSS menu image displays on this Administrative Menu page or not
Show_Status_Line indicates whether the status line (showing copyright and current time) displays on this Administrative Menu page or not

Authenticated indicates whether this Administrative Menu entry requires authentication or not
Disabled indicates whether this Administrative Menu entry is disabled or not
Parent_Menu_ID the Menu ID of the parent page to this Administrative Menu entry
Target_Page_Name name of the ASP script that this Administrative Menu entry points to
Button_Columns number of buttons across this Administrative Menu page
Special_Type indicates whether this page is of a special type (e.g. a login/out button)
Refresh_Seconds number of seconds before this Administrative Menu page should automatically refresh

B.3 Table Name: PSS_Admin_Page

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Page_ID	int	NOT NULL	Yes	No
Page_Name	varchar(40)	NULL	No	No
Privilege_Level	int	NULL	No	No
Authenticated	char(1)	NULL	No	No
Disabled	char(1)	NULL	No	No
Default_Parms	varchar(50)	NULL	No	No
Parent_Menu_ID	int	NULL	No	No
Parent_Page_ID	int	NULL	No	No

PSS Admin Page Table holds privilege and configuration information for the service terminal Administrative Pages

Page_ID unique ID associated with this Administrative Page
Page_Name name of this Administrative Page
Privilege_Level privilege level required to access this Administrative Page
Authenticated indicates whether this Administrative Page is authenticated or not
Disabled indicates whether this Administrative Page is disabled or not
Default_Parms the default parameters sent in the URL when running this Administrative Page
Parent_Menu_ID the Administrative Menu Page ID on which a link to this Administrative Page appears
Parent_Page_ID the Administrative Page ID that acts as a parent to this Administrative Page

B.4 Table Name: PSS_Admin_User

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Username	varchar(20)	NOT NULL	Yes	No
Full_Name	varchar(80)	NULL	No	No
Password	varchar(20)	NULL	No	No
Valid_Datetime	datetime	NULL	No	No
Expires_Datetime	datetime	NULL	No	No
Privilege_Level	int	NULL	No	No

Privilege_Mask	varchar(255)	NULL	No	No
Login_Failures	int	NULL	No	No
Last_IP	varchar(16)	NULL	No	No
Language_ID	int	NULL	No	Yes

PSS Admin User Table holds information about Service Terminal Users

Username the username this Administrative User will log in as
Full_Name the full name of this Administrative User
Password the password this Administrative User will use to log in
Valid_Datetime the date/time this Administrative User account is active
Expires_Datetime the date/time this Administrative User account is no longer active
Privilege_Level the privilege level of this Administrative User
Privilege_Mask the privilege mask for this Administrative User (currently unused)
Login_Failures number of failures since the last successful login for this Administrative User
Last_IP the last IP address from which this Administrative User last accessed their account
Language_ID the language (from PSS_Language) in which this Administrative User prefers to view the service terminal

B.5 Table Name: PSS_Barcode_Variable_Weight

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Barcode_Type	int	NOT NULL	No	No
Barcode_Prefix	varchar(20)	NULL	No	No
Barcode_Prefix_Length	int	NULL	No	No
Item_Start	int	NULL	No	No
Item_Length	int	NULL	No	No
Price_Check_Start	int	NULL	No	No
Price_Check_Length	int	NULL	No	No
Price_Start	int	NULL	No	No
Price_Length	int	NULL	No	No
Price_Or_Weight	char(1)	NULL	No	No

PSS Barcode Variable Weight Table holds configuration information for price or weight-embedded barcodes

Barcode_Type Type of Barcode (unknown, UPC, EAN, etc.)
Barcode_Prefix Sequence of digits starting the embedded barcode
Barcode_Prefix_Length Length of prefix
Item_Start Position of start of item code
Item_Length Length of item code
Price_Check_Start Position of price check digit
Price_Check_Length Length of price check digit(s)
Price_Start Position of price
Price_Length Length of price
Price_Or_Weight Indicates whether barcode is Price or Weight embedded

B.6 Table Name: PSS_Currency

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Currency_ID	int	NOT NULL	Yes	No
Currency_Name	varchar(20)	NULL	No	No
Exchange_Rate	real	NULL	No	No
Num_Decimal_Digits	int	NULL	No	No
Decimal_Char	char(1)	NULL	No	No
Thousands_Char	char(1)	NULL	No	No
Currency_Symbol	char(1)	NULL	No	No
Use_Curr_Symbol	char(1)	NULL	No	No
Price_Times_Factor	int	NULL	No	No
Currency_Symbol2	char(1)	NULL	No	No
Currency_Symbol_Web	char(1)	NULL	No	No

PSS Currency Table holds the supported currency types and conversion rates

Currency ID	Unique ID associated with this currency
Currency Name	text name of currency
Exchange Rate	exchange rate with base currency
Num_Decimal_Digits	number of digits in the stored integer value that are decimal digits
Decimal_Char	character used to delineate the decimal portion
Thousands_Char	character used to delineate groups of three decimal digits
Currency_Symbol	currency symbol used for display on hand-held terminals
Use_Curr_Symbol	indicates whether the currency symbol should be used in display
Price_Times_Factor	amount that a stored value should be multiplied by to arrive at a displayable value
Currency_Symbol2	currency symbol used for display on printed transaction tickets
Currency_Symbol_Web	currency symbol used for display on service terminal pages

B.7 Table Name: PSS_Customer

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	No
Card_Number	varchar(30)	NOT NULL	No	No
First_Name	varchar(40)	NULL	No	No
Middle_Name	varchar(40)	NULL	No	No
Last_Name	varchar(40)	NULL	No	No
Address	varchar(40)	NULL	No	No
Address2	varchar(40)	NULL	No	No
City	varchar(40)	NULL	No	No
State	varchar(40)	NULL	No	No
Postal_Code	varchar(20)	NULL	No	No
Country	varchar(40)	NULL	No	No
Allow_Promotions	char(1)	NULL	No	No
Allow_Checks	char(1)	NULL	No	No

Currency_ID	int	NULL	No	Yes
Language_ID	int	NULL	No	Yes
Rescan_Level	int	NULL	No	No
Rank	int	NULL	No	No
Suspend_Reason	varchar(80)	NULL	No	No
Card_Expiration_Datetime	datetime	NULL	No	No
Num_Trips	int	NULL	No	No
Num_Trips_Since_Rescan	int	NULL	No	No
Last_Trip_Datetime	datetime	NULL	No	No
First_Trip_Accurate	char(1)	NULL	No	No
Greeting_Text	varchar(255)	NULL	No	No
Registered_Datetime	datetime	NULL	No	No
Display_Foodstamp	char(1)	NULL	No	No
Special_Customer	char(1)	NULL	No	No
Suspend_ID	int	NULL	No	Yes
Num_Rescans	int	NULL	No	No
Group_ID	int	NULL	No	No
Num_Rescan_Diffs	int	NULL	No	No
Num_Trips_Incomplete	int	NULL	No	No
Total_Trip_Seconds	int	NULL	No	No
First_Trip_Datetime	datetime	NULL	No	No
Last_Incomplete_Datetime	datetime	NULL	No	No
Total_Spent	int	NULL	No	No
Last_Rescan_Datetime	datetime	NULL	No	No

PSS Customer Table holds all persistent data needed for a given PSS user.

Customer_ID a unique identifier for a given customer
Card_Number number on PSS shopper loyalty card which matches this customer
First_Name customer first name
Middle_Name customer middle name
Last_Name customer last name
Address customer address
Address2 customer address, line 2
City customer city
State customer state/province
Postal_Code customer postal code
Country customer country
Allow_Promotions indicates whether this customer allows marketing message display on hand-held terminal
Allow_Checks indicates whether this customer may be rescanned
Currency_ID customer preferred currency (from PSS_Currency)
Language_ID customer preferred language (from PSS_Language)
Rescan_Level customer rescan level (from PSS_Rescan_Level)
Rank customer rank (0 to 100), typically based on amount spent through PSS in last 30 days
Suspend_Reason reason text for customer suspension from PSS
Card_Expiration_Datetime expiration date/time of the customer's loyalty card
Num_Trips number of PSS shopping trips this customer has been on
Num_Trips_Since_Rescan number of PSS shopping trips since this customer was last rescanned
Last_Trip_Datetime date/time of this customer's last PSS shopping trip
First_Trip_Accurate indicates whether the customer's first shopping trip resulted in an accurate rescan

Greeting_Text greeting text to be displayed upon dispensing a terminal to this customer
Registered_Datetime date/time that this customer was added to PSS
Display_Foodstamp indicates whether this customer is shown foodstamp totals on their and-held terminal
Special_Customer indicates whether this is a special type of customer (e.g. Queue-Buster, Price Checker, or Express Shopping-type customer)
Suspend_ID indicates whether this customer is suspended from PSS, and with which reason code (from PSS_Customer_Suspend_Reason)
Num_Rescans number of rescanned PSS shopping trips for this customer
Group_ID number used for grouping customers (e.g. family cards, etc.)
Num_Rescan_Diffs number of rescanned PSS shopping trips with item count/amount differences for this customer
Num_Trips_Incomplete number of incomplete PSS shopping trips (started, but never checked out) for this customer
Total_Trip_Seconds total number of seconds this customer has spent shopping using a PSS hand-held terminal
First_Trip_Datetime date/time of the first PSS shopping trip for this customer
Last_Incomplete_Datetime date/time of the last incomplete PSS shopping trip for this customer
Total_Spent total amount this customer has spent using PSS
Last_Rescan_Datetime date/time of the last rescanned shopping trip for this customer

B.8 Table Name: PSS_Customer_Message

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Message_Nbr	int	NOT NULL	Yes	No
Message_Text	varchar(255)	NULL	No	No
Start_Datetime	datetime	NULL	No	No
End_Datetime	datetime	NULL	No	No

PSS Customer Message Table holds messages to be displayed at the start of a shopping trip for a particular PSS customer.

Customer_ID unique identifier for this customer
Message_Nbr unique identifier for this message for this customer
Message_Text message text to be displayed
Start_Datetime date/time after which this message is to be displayed
End_Datetime date/time after which this message will no longer be displayed

B.9 Table Name: PSS_Customer_Suspend_Reason

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Suspend_ID	Int	NOT NULL	Yes	No
Suspend_Reason	Varchar(80)	NULL	No	No

PSS Customer Suspend Reason Table holds the Suspend ID reason codes for which a customer may be suspended from using PSS.

Suspend_ID unique identifier for this suspend reason code
Suspend_Reason text description of this suspend reason code

B.10 Table Name: PSS_Department

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Department_ID	Int	NOT NULL	Yes	No
Department_Name	Varchar(40)	NULL	No	No
Department_Type	Varchar(10)	NULL	No	No
Allow_Quick_Order	char(1)	NULL	No	No

PSS Department Table contains the data describing a store's departments

Department ID Unique identifier for a department
Department Name text name of department
Department Type text type of department (meant to provide particular functionality for departments that require universal functionality, i.e. functionality particular to all deli's)
Allow Quick Order when set, this department shows up in the quick order department list

B.11 Table Name: PSS_EOD

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Recvd_Datetime	Datetime	NOT NULL	Yes	No
Description	varchar(50)	NULL	No	No

PSS EOD Table contains the End-Of-Day indicators received from the POS.

Recvd_Datetime date/time when this End-Of-Day indicator was received
Description description of this End-Of-Day indicator

B.12 Table Name: PSS_Item

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Item_ID	varchar(20)	NOT NULL	Yes	No
Description	varchar(40)	NULL	No	No
Price	Int	NULL	No	No
Department_ID	Int	NULL	No	Yes
Class	varchar(20)	NULL	No	No
Subclass	varchar(20)	NULL	No	No
Manufacturer_ID	Int	NULL	No	Yes
Aisle	varchar(10)	NULL	No	No
Shelf	varchar(10)	NULL	No	No
Unit	varchar(10)	NULL	No	No
Weight	varchar(20)	NULL	No	No
Linkcode	varchar(20)	NULL	No	No
Pricing_Method	char(1)	NULL	No	No
Mix_Match	Int	NULL	No	No
Foodstamp	char(1)	NULL	No	No
Flags	Int	NULL	No	No
Deal_Price	Int	NULL	No	No
Deal_Qty	Int	NULL	No	No
Tax_Plan	int	NULL	No	No
POS_Item_Flag	char(1)	NULL	No	No
Currency_ID	int	NULL	No	No

PSS Item Table holds item information; source of information can be the POS item file

Item ID unique item identifier
Description text description of the item
Price unit price of the item represented in the base store currency
Department ID unique identifier for the department assigned to this item (from PSS_Department)
Class the classification group of this item
Subclass a more specific classification of an item within a class
Manufacturer ID unique identifier for this item manufacturer (from PSS_Manufacturer)
Aisle usual aisle in store for this item
Shelf shelf location in store
Unit the unit used to measure quantities of an item (i.e. fluid ounces, pounds, etc.)
Weight the amount of the unit
Linkcode a barcode for an item linked to this item
Pricing Method identifies pricing method to be used for this item
Mix Match when set, this item is part of an identically priced group (i.e. yogurts, canned soups)
Foodstamp when set, indicates that this item is food stamp eligible
Flags holds information specific to the item (i.e. discontinued, applicable tax levels, etc.)
Deal Price The price of the deal, if any, assigned to this item
Deal Quantity quantity of this item which must be purchased to enable the deal
Tax_Plan unique identifier for the text plan under which this item falls (from PSS_Tax_Table)

POS_Item_Flag indicates whether this item has been processed through the POS during the current download process
Currency_ID currency identifier for the currency in which this items price is described (from PSS_Currency)

B.13 Table Name: PSS_Language

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Language_ID	Int	NOT NULL	Yes	No
Language_Name	Varchar(20)	NULL	No	No

PSS Language Table holds the languages supported by the system

Language ID unique identifier for a language
Language Name text name of language

B.14 Table Name: PSS_Manufacturer

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Manufacturer_ID	int	NOT NULL	Yes	No
Manufacturer_Name	varchar(20)	NULL	No	No

PSS Manufacturer Table holds the manufacturers for PSS items

Manufacturer ID unique identifier for a manufacturer
Manufacturer Name text name of manufacturer

B.15 Table Name: PSS_Marketing_Department

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Department_ID	int	NOT NULL	Yes	Yes
Language_ID	int	NOT NULL	Yes	Yes
Message_Text	varchar(255)	NULL	No	No
Target_Coupon_Amt	int	NULL	No	No
Target_Rank	int	NULL	No	No
Start_Datetime	datetime	NULL	No	No
End_Datetime	datetime	NULL	No	No

PSS Marketing Department Table contains the marketing messages for particular departments

Department ID unique department identifier (from PSS_Department)

Language_ID language that this message is in (from PSS_Language)
Message Text marketing message text
Target Coupon Amt if the message contains a coupon offer, this is the amount of the coupon
Target Rank the customer rank level targeted by this message
Start_Datetime date/time after which this message is to be displayed
End_Datetime date/time after which this message will no longer be displayed

B.16 Table Name: PSS_Marketing_Item

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Item_ID	varchar(20)	NOT NULL	Yes	Yes
Language_ID	int	NOT NULL	Yes	Yes
Message_Text	varchar(255)	NULL	No	No
Target_Coupon_Amt	int	NULL	No	No
Target_Rank	int	NULL	No	No
Start_Datetime	datetime	NULL	No	No
End_Datetime	datetime	NULL	No	No

PSS Marketing Item Table contains the marketing messages for particular items

Item ID unique item identifier (from PSS_Item)
Language_ID language that this message is in (from PSS_Language)
Message Text marketing message text
Target Coupon Amt if the message contains a coupon offer, this is the amount of the coupon
Target Rank the customer rank level targeted by this message
Start_Datetime date/time after which this message is to be displayed
End_Datetime date/time after which this message will no longer be displayed

B.17 Table Name: PSS_Marketing_Mfg

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Manufacturer_ID	Int	NOT NULL	Yes	Yes
Language_ID	Int	NOT NULL	Yes	Yes
Message_Text	Varchar(255)	NULL	No	No
Target_Coupon_Amt	int	NULL	No	No
Target_Rank	int	NULL	No	No
Start_Datetime	datetime	NULL	No	No
End_Datetime	datetime	NULL	No	No

PSS Marketing Mfg Table contains the marketing messages for particular manufacturers

Manufacturer ID unique identifier for this manufacturer (from PSS_Manufacturer)
Language_ID language that this message is in (from PSS_Language)
Message Text marketing message text

Target Coupon Amt if the message contains a coupon offer, this is the amount of the coupon
Target Rank the customer rank level targeted by this message
Start_Datetime date/time after which this message is to be displayed
End_Datetime date/time after which this message will no longer be displayed

B.18 Table Name: PSS_Marketing_Sent

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	Int	NOT NULL	Yes	No
Shopping_List_ID	Int	NOT NULL	Yes	No
Marketing_Datetime	Datetime	NOT NULL	Yes	No
Item_ID	Varchar(20)	NULL	No	No
Dept_ID	Int	NULL	No	No
Mfg_ID	Int	NULL	No	No

PSS Marketing Sent Table contains information on which marketing messages have been sent to which customers on current shopping trips

Customer ID unique identifier for this customer (from PSS_Customer)
Shopping List ID identifies shopping list (from PSS_Shopping_List)
Marketing_Datetime date/time when this message was displayed to the customer
Item ID unique item identifier for item messages (from PSS_Item)
Department ID unique department identifier for department messages (from PSS_Department)
Manufacturer ID unique identifier for this manufacturer for manufacturer messages (from PSS_Manufacturer)

B.19 Table Name: PSS_Message_Log

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Msg_Datetime	datetime	NOT NULL	No	No
Msg_Facility	varchar(64)	NOT NULL	No	No
Msg_Action	varchar(64)	NOT NULL	No	No
Msg_Severity	int	NOT NULL	No	No
Msg_Message	varchar(255)	NULL	No	No
Msg_Data	int	NULL	No	No

PSS Message Log Table contains logging information from PSS processes

Msg_Datetime date/time this message was logged
Msg_Facility the facility logging this action (e.g. POSInterfaceCOM)
Msg_Action the action this facility was performing
Msg_Severity the severity level of this message (e.g. Informational, Fatal, etc.)
Msg_Message the message text of this message
Msg_Data optional integer data associated with this message

B.20 Table Name: PSS_Opening_Message

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Msg_ID	Int	NOT NULL	Yes	No
Sequence	Int	NOT NULL	Yes	No
Language_ID	Int	NOT NULL	Yes	Yes
Message	Varchar(20)	NULL	No	No
Start_Datetime	Datetime	NULL	No	No
End_Datetime	Datetime	NULL	No	No
Target_Rank	Int	NULL	No	No

PSS Opening Message Table holds messages that the customer may scroll through at the start of a shopping trip

Msg_ID unique identifier for this message
Sequence sequence number for this line of this message
Language_ID the language this message is in (from PSS_Language)
Message the message text for this line
Start_Datetime date/time after which this message is to be displayed
End_Datetime date/time after which this message will no longer be displayed
Target_Rank the customer rank level targeted by this message

B.21 Table Name: PSS_Order_List

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	Int	NOT NULL	Yes	Yes
Shopping_List_ID	Int	NOT NULL	Yes	Yes
Department_ID	Int	NOT NULL	Yes	Yes
Order_List_ID	Int	NOT NULL	Yes	No
Order_Status_ID	Int	NULL	No	Yes

PSS Order List Table contains the quick-order list information

Customer ID identifies customer associated with this order (from PSS_Customer)
Shopping List ID identifies the customer shopping list associated with this order (from PSS_Shopping_List)
Department ID the department which this order is for (from PSS_Department)
Order List ID unique identifier for this quick order list
Order Status ID status of this order (from PSS_Order_Status, e.g. ready, picked up, etc.)

B.22 Table Name: PSS_Order_List_Item

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	Yes
Department_ID	int	NOT NULL	Yes	Yes
Order_List_ID	int	NOT NULL	Yes	Yes
Item_ID	varchar(20)	NOT NULL	Yes	Yes
Qty	int	NULL	No	No
Notes	varchar(80)	NULL	No	No

PSS Order List Item Table contains the item information in a customer order

Customer ID identifies customer associated to this order item
Shopping List ID identifies shopping list associated to this order item
Department ID the department which this order is for
Order List ID unique identifier for this quick order list (from PSS_Order_List)
Item ID identifies item in the list
Qty quantity ordered of this item
Notes text field entered via browser screen

B.23 Table Name: PSS_Order_Status

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Order_Status_ID	int	NOT NULL	Yes	No
Description	varchar(20)	NULL	No	No

PSS Order Status Table contains the allowable order states

Order Status ID status identifier
Description text message corresponding to this status

B.24 Table Name: PSS_POS_Status

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Item_Name	varchar(30)	NOT NULL	Yes	No
Instance_Name	varchar(100)	NOT NULL	Yes	No
Detection_Time	datetime	NULL	No	No
Start_Time	datetime	NULL	No	No
End_Time	datetime	NULL	No	No
Instance_Value	varchar(20)	NULL	No	No
Instance_Description	varchar(50)	NULL	No	No
Instance_Data	int	NULL	No	No

PSS POS Status Table contains the quick-order list information

Item_Name the type of entry this is (e.g. POSControllerName, PSSItemFile)

Instance_Name the particular instance of this type of entry (typically the controller or file name)
Detection_Time date/time when the activity was initiated
Start_Time date/time when the processing started
End_Time date/time when the processing ended
Instance_Value text value associated with this entry
Instance_Description text description associated with this entry
Instance_Data integer data associated with this entry

B.25 Table Name: PSS_Rescan_Difference

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Difference_Level	int	NOT NULL	Yes	No
Difference_Amt	int	NULL	No	No
Level_Change_Amt	int	NULL	No	No

PSS Rescan Difference Table contains the difference amounts (in currency or percent) between a PSS shopping list, and the re-scanned shopping list, necessary to reduce a customer's rescan possibility by the given number of levels.

Difference Level difference level identifier
Difference Amount the minimum currency amount or percentage necessary for a rescan to qualify he customer for this difference level
Level Change Amt the number of rescan levels that a customer qualifying for this difference level will be reduced by, a greater number of rescan levels indicating a greater possibility of being checked on the next trip.

B.26 Table Name: PSS_Rescan_Level

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Rescan_Level	int	NOT NULL	Yes	No
Pct	int	NULL	No	No
Level_Name	varchar(20)	NULL	No	No

PSS Rescan Level Table contains the rescan levels and the associated probability of being checked when qualifying for each level. Level 1 will hold the highest probability level (i.e. 1:1, or a 100% chance of rescanning on the next PSS shopping trip), with subsequent levels holding increasingly lower probabilities.

Rescan_Level rescan level identifier
Pct the probability 1 in N that this customer will be rescanned on the next shopping trip.
Level_Name name for this rescan level

B.27 Table Name: PSS_Session

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Terminal_ID	varchar(16)	NOT NULL	Yes	No
Customer_ID	int	NOT NULL	No	Yes
Shopping_List_ID	int	NOT NULL	No	Yes
Customer_Name	varchar(45)	NULL	No	No
Status	int	NULL	No	No
Initiating_Customer_ID	int	NULL	No	No
Initiating_Shopping_List_ID	int	NULL	No	No

PSS Session Table contains the high level PSS terminal issue information

Terminal ID unique hand held terminal identifier (from UMS_Terminal)
Customer ID identifies customer (from PSS_Customer)
Shopping List ID identifies shopping list (from PSS_Shopping_List)
Customer Name abbreviated version of the customer name for hand-held terminal display
Status status of this session (used to mark queue-busting session in progress, etc.)
Initiating_Customer_ID identifies the customer who started this session (used to maintain the main queue-busting customer)
Initiating_Shopping_List_ID identifies the shopping list created by the customer who started this session (used to maintain the main queue-busting customer's shopping list)

B.28 Table Name: PSS_Shopping_Activity

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	Int	NOT NULL	Yes	Yes
Shopping_List_ID	Int	NOT NULL	Yes	Yes
Activity_Nbr	Int	NOT NULL	Yes	No
Activity_Datetime	Datetime	NULL	No	No
Activity_Type	Varchar(10)	NULL	No	No
Activity_Detail	Varchar(80)	NULL	No	No

PSS Shopping Activity Table contains activities that occurred on the hand-held terminal during a shopping trip, including addition/removal of items, marketing messages displayed on the terminal, etc.

Customer ID identifies the customer (from PSS_Customer)
Shopping List ID identifies shopping list (from PSS_Shopping_List)
Activity Nbr a sequential number showing the order in which the events occurred
Activity Datetime the date and time of the activity
Activity Type the type of activity which occurred (i.e. ADDITEM, DELITEM, etc.)
Activity Detail a description of the activity which occurred

B.29 Table Name: PSS_Shopping_History_Activity

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	Int	NOT NULL	Yes	Yes
Shopping_List_ID	Int	NOT NULL	Yes	Yes
Activity_Nbr	Int	NOT NULL	Yes	No
Activity_Datetime	Datetime	NULL	No	No
Activity_Type	Varchar(10)	NULL	No	No
Activity_Detail	Varchar(80)	NULL	No	No

PSS Shopping History Activity Table contains activities that occurred on the hand-held terminal during a historical shopping trip, including addition/removal of items, marketing messages displayed on the terminal, etc.

Customer ID identifies the customer (from PSS_Customer)
Shopping List ID identifies shopping list (from PSS_Shopping_List)
Activity Nbr a sequential number showing the order in which the events occurred
Activity Datetime the date and time of the activity
Activity Type the type of activity which occurred (i.e. ADDITEM, DELITEM, etc.)
Activity Detail a description of the activity which occurred

B.30 Table Name: PSS_Shopping_History_Item

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	Yes
Shopping_Item_ID	int	NOT NULL	Yes	No
Barcode	varchar(20)	NULL	No	No
Item_ID	varchar(20)	NULL	No	No
Sale_Price	int	NULL	No	No
Pricing_Method	char(1)	NULL	No	No
Mix_Match	int	NULL	No	No
Department_ID	int	NULL	No	No
Unit_Price	int	NULL	No	No
Deal_Price	int	NULL	No	No
Deal_Qty	int	NULL	No	No
Embedded_Price	char(1)	NULL	No	No
POS_Price	int	NULL	No	No
POS_PSA_Flag	char(1)	NULL	No	No
Exception	char(1)	NULL	No	No
Tax_Plan	int	NULL	No	No

PSS Shopping History Item Table contains the items for a given historical shopping trip

Customer ID customer whom which this shopping trip is associated (from PSS_Customer)
Shopping List ID shopping list with which this item is associated (from PSS_Shopping_List)
Shopping Item ID sequential unique item identifier
Barcode the actual barcode scanned before any translation occurs (i.e. for embedded price barcodes)
Item ID item ID as matched against item information in PSS_Item
Sale Price the actual price paid for this item
Pricing_Method the pricing method used to price this item
Mix_Match the POS mix/match category this item was included in
Department_ID the department ID (from PSS_Department) that this item belongs to
Unit_Price the unit price for this item (from PSS_Item)
Deal_Price the deal price for this item (from PSS_Item)
Deal_Qty the deal quantity for this item (from PSS_Item)
Embedded_Price indicates whether this item had an embedded price barcode
POS_Price the price for this item sent from the POS
POS_PSA_Flag indicates whether this item was in the PSS shopping list (S) only, the POS shopping list (P) only, or both (B).
Exception indicates whether this was flagged as an exception item at checkout
Tax_Plan indicates the tax plan that applied to this item (from PSS_Tax_Table)

B.31 Table Name: PSS_Shopping_History_List

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	No
Time_Started	datetime	NULL	No	No
Time_Ended	datetime	NULL	No	No
Shopping_List_Status_ID	int	NULL	No	No
Total	int	NULL	No	No
Transaction_ID	varchar(25)	NULL	No	No
Rescanned	char(1)	NULL	No	No
Terminal_ID	varchar(16)	NULL	No	No
Scanned_Item_Count	int	NULL	No	No
Trip_Length_Seconds	int	NULL	No	No

PSS Shopping History List Table contains the high-level historical shopping trip information. Once shopping trips are completed, the data is moved into the historical tables.

Customer ID customer with whom this shopping trip is associated (from PSS_Customer)
Shopping List ID unique shopping list identifier
Time Started the date and time that a shopping list was started
Time Ended the date and time that a shopping list was completed
Shopping List Status ID status of this shopping trip (from PSS_Shopping_List_Status)
Total total amount for this trip
Transaction_ID transaction identifier sent to/from the POS to help match shopping lists
Rescanned indicates whether this shopping trip was rescanned
Terminal_ID the terminal ID (from UMS_Terminal) of the terminal last used on this shopping trip
Scanned_Item_Count number of items in this shopping list
Trip_Length_Seconds number of seconds the terminal was in use for this shopping trip

B.32 Table Name: PSS_Shopping_History_Rescan

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	Yes
POS_Price	int	NULL	No	No
PSS_Price	int	NULL	No	No
POS_Qty	int	NULL	No	No
PSS_Qty	int	NULL	No	No

PSS Shopper History Rescan Table contains the re-scan data for a customer's previous shopping trip.

Customer ID customer with whom this shopping trip is associated (from PSS_Customer)
Shopping List ID unique shopping list identifier
POS Price the re-scanned shopping list amount from the POS
PSS Price the original shopping list amount from PSS
POS Qty the number of items in the re-scanned shopping list from the POS
PSS Price the number of items in the original shopping list amount from PSS

B.33 Table Name: PSS_Shopping_Incomplete_List

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	No
Time_Started	datetime	NULL	No	No
Time_Ended	datetime	NULL	No	No
Shopping_List_Status_ID	int	NULL	No	No
Total	int	NULL	No	No
Transaction_ID	varchar(25)	NULL	No	No
Rescanned	char(1)	NULL	No	No
Terminal_ID	varchar(16)	NULL	No	No
Scanned_Item_Count	int	NULL	No	No
Trip_Length_Seconds	int	NULL	No	No

PSS Shopping Incomplete List Table contains the high-level shopping trip information for incomplete shopping trips. The DB Cleanup task moves all current shopping trips that haven't been checked out after a configurable length of time (8 hours is the default) to this table.

Customer ID customer with whom this shopping trip is associated (from PSS_Customer)
Shopping List ID unique shopping list identifier
Time Started the date and time that a shopping list was started
Time Ended the date and time that a shopping list was completed
Shopping List Status ID status of this shopping trip (from PSS_Shopping_List_Status)

Total	total amount for this trip
Transaction_ID	transaction identifier sent to/from the POS to help match shopping lists
Rescanned	indicates whether this shopping trip was rescanned
Terminal_ID	the terminal ID (from UMS_Terminal) of the terminal last used on this shopping trip
Scanned_Item_Count	number of items in this shopping list
Trip_Length_Seconds	number of seconds the terminal was in use for this shopping trip

B.34 Table Name: PSS_Shopping_Item

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	Yes
Shopping_Item_ID	int	NOT NULL	Yes	No
Barcode	varchar(20)	NULL	No	No
Item_ID	varchar(20)	NULL	No	Yes
Sale_Price	int	NULL	No	No
Sale_Price_Currency2	int	NULL	No	No
Pricing_Method	char(1)	NULL	No	No
Mix_Match	int	NULL	No	No
Department_ID	int	NULL	No	No
Unit_Price	int	NULL	No	No
Deal_Price	int	NULL	No	No
Deal_Qty	int	NULL	No	No
Embedded_Price	char(1)	NULL	No	No
Tax_Plan	int	NULL	No	No
Currency_ID	int	NULL	No	No

PSS Shopping Item Table contains the items for a given current shopping trip

Customer ID customer with whom this shopping trip is associated (from PSS_Customer)
Shopping List ID shopping list with which this item is associated (from PSS_Shopping_List)
Shopping Item ID sequential unique item identifier
Barcode the actual barcode scanned before any translation occurs (i.e. for embedded price barcodes)
Item ID item ID as matched against item information in PSS_Item
Sale Price the actual price paid for this item
Sale_Price_Currency2 the actual price paid for this item in the secondary currency
Pricing_Method the pricing method used to price this item
Mix_Match the POS mix/match category this item was included in
Department_ID the department ID (from PSS_Department) that this item belongs to
Unit_Price the unit price for this item (from PSS_Item)
Deal_Price the deal price for this item (from PSS_Item)
Deal_Qty the deal quantity for this item (from PSS_Item)
Embedded_Price indicates whether this item had an embedded price barcode
Tax_Plan indicates the tax plan that applied to this item (from PSS_Tax_Table)
Currency_ID currency identifier (from PSS_Currency) for the currency in which Sale_Price is described

B.35 Table Name: PSS_Shopping_List

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	No
Time_Started	datetime	NULL	No	No
Time_Ended	datetime	NULL	No	No
Shopping_List_Status_ID	int	NULL	No	Yes
Total	int	NULL	No	No
Total_Currency_2	int	NULL	No	No
Transaction_ID	varchar(25)	NULL	No	No
Rescanned	char(1)	NULL	No	No
Terminal_ID	varchar(16)	NULL	No	No
Scanned_Item_Count	int	NULL	No	No
Trip_Length_Seconds	int	NULL	No	No

PSS Shopping List Table contains the high level shopping trip information

Customer ID customer with whom this shopping trip is associated (from PSS_Customer)
Shopping List ID unique shopping list identifier
Time Started the date and time that this shopping list was started
Time Ended the date and time that this shopping list was completed
Shopping List Status ID status of this shopping trip (from PSS_Shopping_List_Status)
Total total amount for this trip
Transaction_ID transaction identifier sent to/from the POS to help match shopping lists
Rescanned indicates whether this shopping trip was rescanned
Terminal_ID terminal ID (from UMS_Terminal) of the terminal last used on this shopping trip
Scanned_Item_Count number of items in this shopping list
Trip_Length_Seconds number of seconds the terminal was in use for this shopping trip

B.36 Table Name: PSS_Shopping_List_Status

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Shopping_List_Status_ID	int	NOT NULL	Yes	No
Description	varchar(20)	NULL	No	No

PSS Shopping List Status Table identifies the allowable shopping list states

Shopping List Status ID unique status identifier
Description text describing this state

B.37 Table Name: PSS_Shopping_Tax

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Customer_ID	int	NOT NULL	Yes	Yes
Shopping_List_ID	int	NOT NULL	Yes	Yes
Tax_Plan_ID	int	NOT NULL	Yes	Yes
Amount	int	NULL	No	No
Tax_Amount	char(18)	NULL	No	No

PSS Shopping Tax Table contains the tax amounts by plan for items from a particular shopping list

Customer ID customer with whom this shopping list is associated (from PSS_Customer)
Shopping List ID shopping list with which this item is associated (from PSS_Shopping_List)
Tax_Plan_ID tax table ID as referenced in item records (from PSS_Tax_Table)
Amount taxable amount in base currency
Tax_Amount amount of tax in base currency (currently unused)

B.38 Table Name: PSS_Special_Barcode

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Barcode	Varchar(20)	NOT NULL	Yes	No
Type	Int	NULL	No	No
Description	Varchar(50)	NULL	No	No

PSS_Special_Barcode Table contains configuration information for barcodes with special meanings, such as an End-Of-Trip barcode, or Queue Buster barcodes.

Barcode the special barcode as returned from the scanner
Type the type of barcode (quick order weight = 10, end-of-trip = 11, swap scanner = 12, queue buster = 13)
Description a text description of this barcode

B.39 Table Name: PSS_System_Message

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
System_Message_ID	int	NOT NULL	Yes	No
Message_Text	varchar(255)	NULL	No	No
Frequency_Type	int	NULL	No	No
Month_Of_Year	int	NULL	No	No
Day_Of_Month	int	NULL	No	No
Day_Of_Week	int	NULL	No	No
Hour_Of_Day	int	NULL	No	No
Minutes_After_Hour	int	NULL	No	No
Time_Interval	int	NULL	No	No
Repeat_Count	int	NULL	No	No
Send_Next_At_Datetime	datetime	NULL	No	No

PSS_System_Message Table contains configuration information for barcodes with special meanings, such as an End-Of-Trip barcode, or Queue Buster barcodes.

System_Message_ID unique identifier for this system message
Message_Text the text of this message
Frequency_Type type of send frequency (annual = 1, monthly = 2, weekly = 3, daily = 4, hourly = 5, interval = 6)
Month_Of_Year month of the year when message should be sent (1-12)
Day_Of_Month day of the month when message should be sent (1-31)
Day_Of_Week day of the week when message should be sent (1-7)
Hour_Of_Day hour of the day when message should be sent (0-23)
Minutes_After_Hour month of the year when message should be sent (1-12)
Time_Interval time interval (in minutes) between which the message should be sent
Repeat_Count number of times to repeat the message send over a time interval
Send_Next_At_Datetime date/time when this message will be sent next (set by PssSystemMessageCOM)

B.40 Table Name: PSS_System_Setting

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Name	Varchar(20)	NOT NULL	Yes	No
Subname	Varchar(20)	NOT NULL	Yes	No
Value	Varchar(255)	NULL	No	No
Description	varchar(255)	NULL	No	No
Unit	varchar(10)	NULL	No	No
Default_Value	varchar(255)	NULL	No	No
Editable	char(1)	NULL	No	No

PSS System Setting Table contains system-wide configuration settings

Name the configuration option name
Subname the component name within the given configuration option name
Value the data associated with this configuration setting
Description the description of this system setting
Unit the unit of measure for this system setting
Default_Value the default Value for this system setting
Editable indicates whether this system setting is editable from the System Setting web page

B.41 Table Name: PSS_Tax_Table

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Tax_Plan_ID	int	NOT NULL	Yes	No
Description	varchar(40)	NULL	No	No
Price_Range_Low	int	NULL	No	No
Tax_Amount_Low	int	NULL	No	No
Price_Range_High	int	NULL	No	No
Tax_Amount_High	int	NULL	No	No
First_Tax_Flag	char(1)	NULL	No	No

PSS Tax Table Table contains the definitions of the possible tax plans for this location

TaxPlan Taxtable ID as referenced in item records
Description Identifies type of tax (State, Federal , Excise, …)
PriceRangeLow Transaction price below which the Low Tax table is used
TaxAmountLow Tax cost for a transaction with price equal to PriceRangeLow
PriceRangeHigh Price range that corresponds to TaxAmountHigh. Transaction price is divided by this amount to determine the number of TaxAmountHigh increments to charge. Any remainder is computed by lookup in the High Tax Table for the Tax Plan.
TaxAmountHigh Tax cost for each PriceRangeHigh increment.
First_Tax_Flag indicates whether tax is collected on the first tax range.

B.42 Table Name: PSS_Tax_Table_Entry

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Tax_Plan_ID	Int	NOT NULL	Yes	Yes
Range_Indicator	char(1)	NOT NULL	Yes	No
Price_Group_Start	int	NOT NULL	Yes	No
Tax_Amount	int	NULL	No	No

PSS Tax Table Entries Table contains the definitions of the tax ranges for each tax plan

Tax Plan Tax table ID as referenced in item records
Range Indicator "H" or "L" indicates whether this is an entry in the Low tax table or is an entry in the High tax table
Price Group Start Threshold price for identifying a transaction with this tax group
Tax Amount Tax cost for a transaction in this tax group

B.43 Table Name: PSS_Text

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Text_ID	int	NOT NULL	Yes	No
Language_ID	int	NOT NULL	Yes	Yes
Constant_Name	varchar(40)	NULL	No	No
Text	varchar(255)	NULL	No	No

PSS_Text Table contains text strings displayed throughout the system in various languages

Text_ID unique identifier for this text entry
Language_ID language ID (from PSS_Language) that this version of this text entry is in
Constant_Name a C++-style identifier for this text entry
Text the text for this text entry in this language

B.44 Table Name: PSS_Unknown_Item

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Item_ID	varchar(20)	NOT NULL	Yes	No
Terminal_ID	varchar(16)	NOT NULL	Yes	No
Scanned_Datetime	datetime	NOT NULL	Yes	No
Customer_ID	int	NULL	No	No
Shopping_List_ID	int	NULL	No	No

PSS_Unknown_Item Table contains items scanned during shopping trips for which no matching item could be found in PSS_Item

Item_ID	the barcode as returned by the scanner
Terminal_ID	the Terminal ID (from UMS_Terminal) that scanned this item
Scanned_Datetime	date/time that this item was scanned
Customer_ID	the customer ID (from PSS_Customer) that scanned this item
Shopping_List_ID	the shopping list ID (from PSS_Shopping_List) of the list that this item was to be added to

B.45 Table Name: UMS_Card_Reader_Type

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Reader_Type	Int	NOT NULL	Yes	No
Description	Varchar(50)	NULL	No	No
Card_Sentinel	varchar(5)	NULL	No	No
Card_Offset	int	NULL	No	No
End_Sentinel	varchar(5)	NULL	No	No

UMS_Card_Reader_Type Table contains definitions of the various types of card reading hardware used by the system

Reader_Type	a unique identifier for the type of card reader hardware for this entry
Description	the description of this card reader
Card_Sentinel	the sentinel character(s) that mark the start of the card number
Card_Offset	the number of characters after the first character of the start sentinel before the card number begins
End_Sentinel	the sentinel character(s) that mark the end of the card number

B.46 Table Name: UMS_Cradle

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Cradle_ID	varchar(4)	NOT NULL	Yes	No
Power_ID	int	NULL	No	Yes
Dispenser_ID	varchar(4)	NULL	No	Yes
Status_Code	int	NULL	No	Yes
Status_Datetime	datetime	NULL	No	No
Cradle_Notes	varchar(40)	NULL	No	No
Default_Interface	varchar(40)	NULL	No	No

UMS_Cradle Table contains definitions of the configured cradles, or charging slots, for the terminals used by the system

Cradle_ID the unique identifier scanned by a terminal placed into this cradle
Power_ID the power supply ID (from UMS_Power) from which this cradle gets its power
Dispenser_ID the dispenser ID (from UMS_Dispenser) that this cradle is associated with
Status_Code the status code (from UMS_Status_Codes) for the current state of this cradle
Status_Datetime the date/time of the last status update for this cradle
Cradle_Notes text associated with this cradle (typically for reason marked out of service)
Default_Interface the default interface for the terminals dispensed from this cradle

B.47 Table Name: UMS_Dispenser

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Dispenser_ID	varchar(4)	NOT NULL	Yes	No
Dispenser_Name	varchar(50)	NULL	No	No
Entrance_ID	int	NULL	No	Yes
Printer_ID	int	NULL	No	Yes
Default_Entrance	int	NULL	No	No
Status_Code	int	NULL	No	Yes
Status_Datetime	datetime	NULL	No	No
Default_Printer	int	NULL	No	No
Current_Mode	int	NULL	No	No

UMS_Dispenser Table contains definitions of the dispensers, or logical grouping of cradles, used by the system

Dispenser_ID a unique identifier for this dispenser
Dispenser_Name a text description of this dispenser
Entrance_ID the entrance unit ID (from UMS_Entrance) associated with this dispenser
Printer_ID the printer ID (from UMS_Printer) associated with this dispenser

Default_Entrance the default entrance unit ID (from UMS_Entrance) for this dispenser, in case the entrance unit in use must be changed
Status_Code the status code (from UMS_Status_Codes) for the current state of this dispenser
Status_Datetime the date/time of the last status update for this dispenser
Default_Printer the default printer ID (from UMS_Printer) for this dispenser, in case the printer in use must be changed
Current_Mode indicates whether this dispenser is in Key-Controlled Unlock mode or not

B.48 Table Name: UMS_Entrance

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Entrance_ID	int	NOT NULL	Yes	No
Description	varchar(50)	NULL	No	No
Reader_Type	int	NULL	No	Yes
Reader_Port	varchar(8)	NULL	No	No
Status_Code	int	NULL	No	Yes
Status_Datetime	datetime	NULL	No	No
Display_Type	int	NULL	No	No
Display_Port	varchar(8)	NULL	No	No
Default_Mode	int	NULL	No	No
Current_Mode	int	NULL	No	No
Backup_ID	int	NULL	No	No
Hardware_ID	varchar(16)	NULL	No	No
Reader_Status	int	NULL	No	No
Display_Status	int	NULL	No	No
Entrance_Notes	varchar(40)	NULL	No	No

UMS_Entrance Table contains definitions of the configured entrance units used by the system

Entrance_ID unique identifier for this entrance unit
Description description of this entrance unit
Reader_Type the type of card reader (from UMS_Card_Reader_Type) used by this entrance unit
Reader_Port the communications port (typically COMx port) used to communicate with the non-RF card reader
Status_Code the status code (from UMS_Status_Codes) for the current state of this entrance unit
Status_Datetime the date/time of the last status update for this entrance unit
Display_Type the type of display used by this entrance unit
Display_Port the communications port (typically COMx port) used to communicate with the non-RF display
Default_Mode the default status message displayed by this entrance unit (swipe card, no terminals available, etc.)
Current_Mode the current status message displayed by this entrance unit (swipe card, no terminals available, etc.)
Backup_ID the entrance unit ID (from UMS_Entrance) to act as a backup if this entrance unit is out of order

Hardware_ID the hardware ID for this entrance unit (either the name of the PssEntranceXX service which controls this non-RF entrance unit, or the MAC address for RF entrance units)
Reader_Status the status code (from UMS_Status_Codes) for the card reader associated with this entrance unit
Display_Status the status code (from UMS_Status_Codes) for the display associated with this entrance unit
Entrance_Notes text associated with this entrance unit (typically for reason out of service)

B.49 Table Name: UMS_Power

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Power_ID	int	NOT NULL	Yes	No
Description	varchar(50)	NULL	No	No
Status_Code	int	NULL	No	Yes
Status_Datetime	datetime	NULL	No	No
Power_Notes	varchar(40)	NULL	No	No

UMS_Power Table contains definitions of the configured power supplies used by the system

Power_ID unique identifier for this power supply
Description description of this power supply
Status_Code the status code (from UMS_Status_Codes) for the current state of this power supply
Status_Datetime the date/time of the last status update for this power supply
Power_Notes text associated with this power supply (typically for reason out of service)

B.50 Table Name: UMS_Printer

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Printer_ID	int	NOT NULL	Yes	No
Description	varchar(50)	NULL	No	No
Printer_Type	int	NULL	No	No
Printer_Port	varchar(8)	NULL	No	No
Status_Code	int	NULL	No	Yes
Status_Datetime	datetime	NULL	No	No
Backup_ID	int	NULL	No	No
Current_Mode	int	NULL	No	No
Hardware_ID	varchar(16)	NULL	No	No
Printer_Notes	varchar(40)	NULL	No	No
Paper_Remaining	int	NULL	No	No

UMS_Printer Table contains definitions of the configured printers used by the system

Printer_ID unique identifier for this printer
Description description of this printer
Printer_Type the type of printer
Printer_Port the communications port (typically COMx port) used to communicate with the printer
Status_Code the status code (from UMS_Status_Codes) for the current state of this printer
Status_Datetime the date/time of the last status update for this printer
Backup_ID the printer ID (from UMS_Printer) for the printer to be used as a backup for this printer
Current_Mode the current mode of this printer (currently unused)
Hardware_ID the name of the printer service (PssPrinterXX) which controls this printer
Printer_Notes text associated with this printer (typically for reason out of service)
Paper_Remaining a number associated with the remaining paper in this printer (currently unused)

B.51 Table Name: UMS_Status_Codes

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Status_Code	int	NOT NULL	Yes	No
Status_Text	varchar(30)	NULL	No	No

UMS_Status_Codes Table contains definitions of the hardware status codes used throughout the system

Status_Code unique identifier for this status code
Status_Text text description of this status code

B.52 Table Name: UMS_Terminal

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Terminal_ID	varchar(16)	NOT NULL	Yes	No
Cradle_ID	varchar(4)	NULL	No	No
Status_Code	int	NULL	No	Yes
Status_Datetime	datetime	NULL	No	No
Battery_Level	int	NULL	No	No
Battery_Datetime	int	NULL	No	No
Terminal_Type	varchar(8)	NULL	No	Yes
Good_Scans	int	NULL	No	No
Bad_Scans	int	NULL	No	No
Seconds_Used	int	NULL	No	No
Tot_Issue_Count	int	NULL	No	No
Tot_Good_Scans	int	NULL	No	No
Tot_Bad_Scans	int	NULL	No	No
Tot_Seconds_Used	int	NULL	No	No
Timeout_Count	int	NULL	No	No
Current_Interface	varchar(50)	NULL	No	No
Dispenser_ID	varchar(4)	NULL	No	No
Terminal_Notes	varchar(40)	NULL	No	No
Battery_Status	int	NULL	No	No

UMS_Terminal Table contains definitions of the configured hand-held terminals used by the system

Terminal_ID unique identifier for this terminal (the MAC address of its network card)
Cradle_ID the cradle ID (from UMS_Cradle) for the current location of this terminal
Status_Code the status code (from UMS_Status_Codes) for the current state of this terminal
Status_Datetime the date/time of the last status update for this terminal
Battery_Level a number corresponding to the amount of battery life remaining for this terminal
Battery_Datetime the date/time of the last battery level update for this terminal
Terminal_Type the type of terminal (from UMS_Terminal_Types)
Good_Scans the number of good scans for this terminal from the last issue

Bad_Scans the number of bad scans for this terminal from the last issue
Seconds_Used the number of seconds this terminal was used from the last issue
Tot_Issue_Count the total number of times this terminal was issued (since last statistics reset)
Tot_Good_Scans the total number of good scans for this terminal (since last statistics reset)
Tot_Bad_Scans the total number of bad scans for this terminal (since last statistics reset)
Tot_Seconds_Used the total number of seconds this terminal was used (since last statistics reset)
Timeout_Count the total number of issue timeouts for this terminal (since last issue or last statistics reset)
Dispenser_ID the dispenser ID (from UMS_Dispenser) for the dispenser this terminal is currently in
Terminal_Notes text associated with this terminal (typically for reason out of service)
Battery_Status indicates whether the terminal is currently on battery or AC power
Current_Interface the current interface this terminal is running

B.53 Table Name: UMS_Terminal_Types

Table Column Name	Table Column Datatype	Table Column Null Option	Table Column Is PK	Table Column Is FK
Terminal_Type	varchar(8)	NOT NULL	Yes	No
Terminal_Class	varchar(50)	NULL	No	No
Default_Interface	varchar(40)	NULL	No	No
Display_Rows	int	NULL	No	No
Display_Cols	int	NULL	No	No
Keypad_Type	int	NULL	No	No

UMS_Terminal_Types Table contains definitions of the types of hand-held terminals used by the system

Terminal_Type unique identifier for this type of terminal
Terminal_Class a text description for this type of terminal
Default_Interface the default interface run by this type of terminal
Display_Rows the number of rows available on the display of this type of terminal
Display_Cols the number of columns available on the display of this type of terminal
Keypad_Type the keypad type, or number of keys, available on this type of terminal

Appendix C Price Calculation Algorithms

This appendix describes how price calculations are performed.

A price is calculated for every item scanned during a shopping trip. Each item has an associated pricing method. The following item characteristics are used to determine how to calculate its price.

- Pricing method—The pricing method helps to determine the equation used to calculate the price of an item.
- Mix and match number—The mix and match number is used to indicate whether or not an item can be part of a pricing deal that includes other items.
- Department—The department helps to determine which mix and match pricing deal to apply to an item.

The following item characteristics are used as inputs to price calculation equations.

- Deal quantity / Deal weight—A deal quantity must never be negative. A deal quantity of zero is treated as a one.
- Deal price—The meaning of the deal price associated with an item depends on the pricing method associated with that item. A deal price is always equal to or greater than zero.
- Unit price—The unit price helps to determine which mix and match pricing deal to apply to an item. The unit price is always equal to or greater than zero.

C.1 Pricing Methods

The following pricing methods are used by the PSS system:

- Split Package Pricing
- Unit Pricing
- Base Plus One Pricing
- Group Threshold Pricing
- Group Adjusted Pricing
- Unit Adjusted Pricing

Below is a description of each of the pricing methods used on the PSS system. Included in each description is an example calculation. All pricing methods described as a group pricing method require a price rounding method. All calculated prices in the examples are always rounded up.

C.2 Split Package Pricing

The *Split Package Pricing Method* is a group pricing method based on a deal price and a deal quantity or a deal weight.

The price of an item is calculated using one of the following equations:

$$\text{Item Price} = \frac{(\text{Purchase Quantity} * \text{Deal Price})}{\text{Deal Quantity}}$$

OR

$$\text{Item Price} = \frac{(\text{Purchase Weight} * \text{Deal Price})}{\text{Deal Weight}}$$

Split Package Pricing Example

An item costs $1.00 for a quantity of 5. The customer purchases a quantity of 3 of the item.

$$\text{Item Price} = (3 * 1.00)/5 = \$0.60$$

A weight-based item costs $5.00 for a weight of 2 lbs. The customer purchases 3 lbs. of the item.

$$\text{Item Price} = (3 * 5.00)/2 = \$7.50$$

C.3 Unit Pricing

The *Unit Pricing Method* is a simplified version of the *Split Package Pricing Method.* The price used in the equation is the unit price instead of the deal price and the deal quantity / deal weight is equal to 1 or 0.

The price of an item is calculated using one of the following equations:

$$\text{Item Price} = \text{Purchase Quantity} * \text{Unit Price}$$

OR

```
Item Price = Purchase Weight  *  Unit Price
```

Unit Pricing Example

An item costs $0.49 for a quantity of 1. The customer purchases a quantity of 3 of the item.

```
Item Price =  3 * $0.49 = $1.47
```

A weighted item costs $1.29 for a weight of 1 lb. The customer purchases 3 lbs. of the item.

```
Item Price =  3 * $1.29 = $3.87
```

C.4 Base Plus One Pricing

The *Base Plus One Pricing Method* is a group pricing method based on a deal price and a deal quantity. This pricing method accumulates previously sold items of the same type or group. Every time an item needs to be included in the group the total price is recalculated for the new purchase quantity. The price of the added item is adjusted to reflect the new total for the group.

The price of the added item is calculated using the following equations.

$$\text{New Total Price} = \frac{(\text{Purchase Quantity} * \text{Deal Price})}{\text{Deal Quantity}}$$

$$\text{Item Price} = \text{New Total Price} - \text{Previous Total Price}$$

Base Plus One Pricing Example

Items of the same type or group are sold at five for $0.47. The items are added at different times during the shopping trip.

Item Number 1 added to shopping list.

```
New Total Price = ( 1 * $0.47 )/5 = $0.10
Item Price = $0.10 – $0.00 = $0.10
```

Item Number 2 added to shopping list.

```
New Total Price = ( 2 * $0.47 )/5 = $0.19
Item Price = $0.19 – $0.10 = $0.09
```

Item Number 3 added to shopping list.

```
New Total Price = ( 3 * $0.47 )/5 = $0.29
Item Price = $0.29 – $0.19 = $0.10
```

Item Number 4 added to shopping list.

```
New Total Price = ( 4 * $0.47 )/5 = $0.38
Item Price = $0.38 – $0.29 = $0.09
```

Item Number 5 added to shopping list.

```
New Total Price = ( 5 * $0.47 )/5 = $0.47
Item Price = $0.47 – $0.38 = $0.09
```

Item Number 6 added to shopping list.

```
New Total Price = ( 6 * $0.47 )/5 = $0.57
Item Price = $0.57 – $0.47 = $0.10
```

C.5 Group Threshold Pricing

The *Group Threshold Pricing Method* is a group pricing method based on a unit price, a deal price, and a deal quantity. This pricing method accumulates previously sold items of the same type or group. The unit price is applied to items added to a group until its deal quantity is reached. When the deal quantity is reached the total price is calculated using the deal price and the deal quantity. The price of the threshold item is adjusted to reflect the total price for the group. Adding additional items to the group begins a new deal.

The prices of the items sold before reaching the deal quantity are calculated using the following equations:

```
Item Price = Unit Price

                Purchase Quantity
 Total Price = ------------------ * Deal Price +
               Deal Quantity

             ((Purchase Quantity % Deal Quantity) * Unit Price)
```

The price of the threshold item is calculated using the following equations:

```
                (Purchase Quantity * Deal Price)
Total Price  = --------------------------------
```

Deal Quantity

Item Price = Total Price – Previous Total Price

Group Threshold Pricing Example

Items of the same type or group are sold at five for $0.47. The unit price of the items in the group is $0.10. The items are added at different times during the shopping trip.

Item Number 1 added to shopping list.

```
Item Price  = $0.10
Total Price = ((1/5) * 0.47) + ((1 % 5) * $0.10) = $0.10
```

Item Number 2 added to shopping list.

```
Item Price  = $0.10
Total Price = ((2/5) * 0.47) + ((2 % 5) * $0.10) = $0.20
```

Item Number 3 added to shopping list.

```
Item Price  = $0.10
Total Price = ((3/5) * 0.47) + ((3 % 5) * $0.10) = $0.30
```

Item Number 4 added to shopping list.

```
Item Price  = $0.10
Total Price = ((4/5) * 0.47) + ((4 % 5) * $0.10) = $0.40
```

Item Number 5 added to shopping list.

```
Total Price = ( 5 * $0.47 )/5 = $0.47
Item Price  = $0.47 – $0.40 = $0.07
```

Item Number 6 added to shopping list.

```
Item Price  = $0.10
Total Price = ((6/5) * 0.47) + ((6 % 5) * $0.10) = $0.57
```

Item Number 7 added to shopping list.

```
Item Price  = $0.10
Total Price = ((7/5) * 0.47) + ((7 % 5) * $0.10) = $0.67
```

Item Number 8 added to shopping list.

```
Item Price  = $0.10 ( Unit Price )
Total Price = ((8/5) * 0.47) + ((8 % 5) * $0.10) = $0.77
```

Item Number 9 added to shopping list.

```
Item Price  = $0.10 ( Unit Price )
Total Price = ((9/5) * 0.47) + ((9 % 5) * $0.10) = $0.87
```

Item Number 10 added to shopping list.

```
Total Price = ( 10 * $0.47 )/5 = $0.94
Item Price  = $0.94 – $0.87 = $0.07
```

C.6 Group Adjusted Pricing

The *Group Adjusted Pricing Method* is a group pricing method based on a unit price, a deal price and a deal quantity. This pricing method accumulates previously sold items of the same type or group. The deal price for this pricing method is often referred to as the reduced price. The deal price is treated as an item price rather than a package price.

The unit price is applied to items in a group until the deal quantity is reached. The price of the threshold item is calculated to bring the total group cost equal to the number of items sold at the deal price. All items added to the group after the threshold item are sold at the deal price.

The prices of the items sold before the threshold item are calculated using the following equations:

```
Item Price  = Unit Price
Total Price = Purchase Quantity * Unit Price
```

The price of the threshold item is calculated using the following equations:

```
Total Price = Purchase Quantity * Deal Price
Item Price  = Total Price – Previous Deal Price
```

The prices of all items added after the threshold item are calculated using the following equations:

```
Item Price  = Deal Price
Total Price = Purchase Quantity * Deal Price
```

Group Adjusted Threshold Pricing Example

Items of the same type or group are sold at a unit price of $0.10. A special price of $0.08 is activated when 3 items are purchased.

Item Number 1 added to shopping list.

```
Item Price  = $0.10
Total Price = 1 * $0.10 = $0.10
```

Item Number 2 added to shopping list.

```
Item Price  = $0.10
Total Price = 2 * $0.10 = $0.20
```

Item Number 3 added to shopping list.

```
Total Price = 3 * $0.08 = $0.24
Item Price  = $0.24 - $0.20 = $0.04
```

Item Number 4 added to shopping list.

```
Item Price  = $0.08
Total Price = 4 * $0.08 = $0.32
```

Item Number 5 added to shopping list.

```
Item Price  = $0.08
Total Price = 5 * $0.08 = $0.40
```

C.7 Unit Adjusted Pricing

The *Unit Adjusted Pricing Method* is a group pricing method based on a unit price, a deal price and a deal quantity. This pricing method accumulates previously sold items of the same type or group. The deal price for this pricing method is often referred to as the reduced price. The deal price is treated as an item price rather than a package price. This pricing method can be used to control the distribution of free merchandise.

The deal price is applied to items in a group until the deal quantity is exceeded. All items added to the group after the threshold item are sold at the unit price.

The prices of the items sold before the deal quantity is exceeded are calculated using the following equations:

```
Item Price  = Deal Price
```

```
Total Price = Purchase Quantity * Deal Price
```

The prices of all items added after the threshold item are calculated using the following equations:

```
Item Price  = Unit Price

Total Price = ( Deal Quantity * Deal Price ) +

              ( ( Purchase Quantity – Deal Quantity ) * Unit
Price )
```

Unit Adjusted Threshold Pricing Example 1

Items of the same type or group are sold at a unit price of $0.25. The items are on special for $0.20 with a limit of 3 per customer.

Item Number 1 added to shopping list.

```
Item Price  = $0.20
Total Price = 1 * $0.20 = $0.20
```

Item Number 2 added to shopping list.

```
Item Price  = $0.20
Total Price = 2 * $0.20 = $0.40
```

Item Number 3 added to shopping list.

```
Item Price  = $0.20
Total Price = 3 * $0.20 = $0.60
```

Item Number 4 added to shopping list.

```
Item Price  = $0.25
Total Price = ( 3 * $0.20 ) + ( ( 4 – 3 ) * $0.25 ) = $0.85
```

Item Number 5 added to shopping list.

```
Item Price  = $0.25
```

Unit Adjusted Threshold Pricing Example 2

Items of the same type or group are sold at a unit price of $0.20. One item of the group per shopping trip is free.

Item Number 1 added to shopping list.

```
Item Price  = $0.00
Total Price = 1 * $0.00 = $0.00
```

Item Number 2 added to shopping list.

```
Item Price  = $0.20
Total Price = ( 1 * $0.00 ) + ( ( 2 – 1 ) * $0.20 ) = $0.20
```

Item Number 3 added to shopping list.

```
Item Price  = $0.20
Total Price = ( 1 * $0.00 ) + ( ( 3 – 1 ) * $0.20 ) = $0.40
```

C.8 Mix and Match Groupings

Mix and match is the name for the technique that lets a customer purchase items within the same group or pricing deal while mixing items but matching the unit price. Each item eligible for a mix and match pricing deal will have the same group number in the item record of each item in the group. Mixed items in a pricing deal must have the same unit price and the same department.

C.9 Pricing Method Application Rules

The following application rules apply to the pricing methods.

- All weight items use either the *Unit Pricing Method* or the *Split Package Pricing Method*.
- All items with price embedded barcodes use the *Unit Pricing Method*. The embedded price becomes the unit price and the purchase quantity is treated as a one.
- If an item is assigned to use the *Base Plus One Pricing Method* but its deal price is evenly divisible by its deal quantity, its price is calculated using the *Split Package Pricing Method.*
- If an item is assigned to use the *Group Threshold Pricing Method* and its deal quantity is 1, its price is calculated using the *Unit Pricing Method.*
- Only the *Group Adjusted Pricing Method* and the *Unit Adjusted Pricing Method* allow an item to be given away using a deal price of zero and a deal quantity of one.
- Mix and match groupings can be used in the *Unit Pricing Method*, the *Base Plus One Pricing Method*, the *Group Threshold Pricing Method*, the *Group Adjusted Pricing Method*, and the *Unit Adjusted Pricing Method.*

C.10 Rounding Methods

The following rounding methods are used on the PSS system:

Rounding Method 1

Prices are rounded up. Any decimal past the units position increases the units position by 1.

```
 ( 1 * $1.00 )
-------------- = $0.33333    Rounds up to $0.34
     3
```

Rounding Method 2

Prices are rounded down. Any decimal past the units position is discarded.

```
( 1 * $1.00 )
-------------- = $0.33333    Rounds down to $0.33
     3
```

Rounding Method 3

Prices are rounded up and down. Any decimal past the units position is rounded down for decimals one through four and rounded up for decimals five through nine.

```
( 1 * $1.00 )
-------------- = $0.33333    Rounds down to $0.33
     3

( 2 * $1.00 )
-------------- = $0.66666    Rounds up to $0.67
     3
```

Appendix D POS Interface File Descriptions

Information is exchanged between the PSS system and the POS system using ASCII flat files copied between the systems. The files are exchanged in a directory configured in the PSS System Settings. It defaults to the C:\PSS\TRANSFER subdirectory.

This section contains detailed information regarding the contents of these files. Each of these files contains multiple record types. The record type identifier is a two byte ASCII numeric value which is the first two bytes of the record.
The file formats match those of the PSA version 5 release with some minor modifications.

Record Type range	Associated POS Interface file
"01", "02"	PssTransaction File
"03" - "06"	POSTransaction File
"08"	Item Record File
"09"	Customer Information File
NA	TaxTable File

D.1 PSS Transaction File

Filename	File directory path specified in PSS System Settings. The file name defaults to SCxxyyyy.IN, where 'xx' is the day of the month of this transaction as specified in the transaction barcode on the dispenser ticket and 'yyyy' is the four character transaction number stored in the transaction barcode on the dispenser ticket (see transaction ticket barcode description in this integration guide for details of format).
Source:	PSS system
Description	Contains the list of items scanned by the PSS shopper during a selfscan shopping trip
Type	ASCII Sequential
Record Length	Fixed for each record type (length depends on expansion area configuration)
Number of Records	Variable

This file consists of a transaction information and item information. The Standard Transaction Information has details of the customer number, the selfscan transaction 'barcode', the number of items in the selfscan transaction and whether the POS operator is to check (re-scan) the items. After the Transaction record are the item records. Each item record represents one article from the customer selfscan transaction and includes the item code, price and a number of implementation specific 'flags'. The file is in ASCII format with each record being terminated by a carriage return/line feed delimiting character pair.

Transaction Information (Header record)

Field Name	Start Offset	Length	Comments
Record Type	0	2	Standard SCAN IN transaction information = 01
Customer Number	2	20	Customer Loyalty Card Number (right justified, zero filled on left)
Transaction 'Barcode'	22	24	This field is used to link the self scan transaction receipt barcode to the self scan transaction. (right justified, zero filled on left)
Date & Time	46	14	Date and time of self scan transaction in format DDMMYYYYHHMMSS.
Number Of Items	60	4	The number of articles in the self scan transaction (right justified, zero filled on left).
Check Indicator	64	1	Flag to indicate if the POS operator is to check (re-scan) the selfscan transaction items. "N" = Don't check (Quick Pay) "Y" = Check (RESCAN) "S" = Pre-scan (Queue Buster)
PSS Expansion Area			This area is reserved for future expansion by the PSS system. Its size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. Its size is configurable and defaults to zero length.
Record Terminator	66	2	Carriage Return/Linefeed (0x0D, 0x0A)

Item Information

Field Name	Start Offset	Length	Comments
Record Type	0	2	'Standard' SCAN IN item record type = 02
Item Number	2	24	Item barcode number right justified, zero filled on left (only last 13 digits are currently used).
Item Price	26	10	Item price in units (max 9,999,999,999) (right justified, zero filled on left)
Flags	36	5	Implementation specific flags. Defaults as follows: 'X_1 X_2 X_3 X_4 X_5' X_1 unused X_2 0=Price in Store Currency, 1=Price in Euros X_3 0=Normal item, 1=reduced to clear (RTC) item X_4 unused X_5 unused
PSS Expansion Area			This area is reserved for future expansion by the PSS system. It's size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. It's size is configurable and defaults to zero length.
Record Terminator	41	2	Carriage Return/Linefeed (0x0D, 0x0A)

D.2 POS Transaction File

Filename	File directory path specified in PSS System Settings. The file name defaults to SCxxyyyy.OUT, where 'xx' is the day of the month of this transaction as specified in the transaction barcode on the dispenser ticket and 'yyyy' is the four character POS transaction number.
Source:	POS system
Description	Contains the list of items processed at the POS terminal by the cashier during the checkout process.
Type	ASCII Sequential
Record Length	Fixed for each record type (length depends on expansion area configuration
Number of Records	Variable, one header record, plus one record for each item purchased.

This file consists of a transaction information and item information. The Header Record has details of the customer number, the selfscan transaction 'barcode', the number of items in the selfscan transaction and whether the POS operator re-scanned the items. Following the header record are the item records. Each item record represents one article processed at the POS and includes the item code, price and a number of implementation specific 'flags'. The file is in ASCII format with each record being terminated by a carriage return/line feed delimiting character pair.

Transaction Information (Header record)

Field Name	Start Offset	Length	Comments
Record Type	0	2	'Standard' SCAN OUT transaction information = 03
Customer Number	2	20	Customer Loyalty Card Number (right justified, zero filled on left)
Transaction 'Barcode'	22	24	This field is used to link the self scan transaction receipt barcode to the self scan transaction. Note that this field may used for other purposes in specific system implementation (right justified, zero filled on left)
Date & Time	46	14	Date and time of POS rescan transaction in format DDMMYYYYHHMMSS
Number Of Items	60	4	The number of articles in the checked transaction (right justified, zero filled on left)
Check Indicator	64	1	Flag to indicate if the POS operator did check (re-scan) the selfscan transaction items. "N" = Didn't check (Quick Pay) "Y" = Did Check (RESCAN)
PSS Expansion Area			This area is reserved for future expansion by the PSS system. It's size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. It's size is configurable and defaults to zero length.
Record Terminator	65	2	Carriage Return/Linefeed (0x0D, 0x0A)

Standard Item Information

Field Name	Start Offset	Length	Comments
Record Type	0	2	'Standard' item information = 04
Item Number	2	24	Item barcode number right justified, zero filled on left (only the last 13 bytes are currently used).
Item Price	26	10	Item price in units (max 9,999,999,999) (right justified, zero filled on left)
Flags	36	5	Implementation specific flags. Defaults as follows: 'X_1 X_2 X_3 X_4 X_5' X_1 unused X_2 unused X_3 0=Normal item, 1=exception item X_4 unused X_5 unused
PSS Expansion Area			This area is reserved for future expansion by the PSS system. It's size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. It's size is configurable and defaults to zero length.
Record Terminator	41	2	Carriage Return/Linefeed (0x0D, 0x0A)

End Of Day Information

Field Name	Start Offset	Length	Comments
Record Type	0	2	End of Day information = 05
Identifier	2	3	"EOD"
Reserved	5	2	Zero filled
PSS Expansion Area			This area is reserved for future expansion by the PSS system. It's size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. It's size is configurable and defaults to zero length.
Record Terminator	7	2	Carriage Return/Linefeed (0x0D, 0x0A)

End Of Week Information

Field Name	Start Offset	Length	Comments
Record Type	0	2	End of Week information = 06
Identifier	2	3	"EOW"
Week Number	5	2	"00" – "52"
PSS Expansion Area			This area is reserved for future expansion by the PSS system. It's size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. It's size is configurable and defaults to zero length.
Record Terminator	7	2	Carriage Return/Linefeed (0x0D, 0x0A)

D.3 Item Record File

Filename	Specified in PSS System Settings, defaults to PLUMT??.DAT for updates and to PLUNW.DAT for a full Item file replacement.
Source:	POS Controller
Description	Contains item attributes needed by PSS for retrieval of item descriptions & computation of item prices.
Type	ASCII Sequential
Record Length	Fixed, 117* (Actual length depends on configured expansion area sizes)
Number of Records	Variable

This file is created by the POS. PSS transfers the file from the POS host to the PSS system and then processes it to load the data contained in it into the PSS. It exists primarily to update item description and pricing information.

Field Name	Start Offset	Length	Comments
Record Type	0	2	Item file information, value = "02"
Action	2	1	'A' = Add\Update 'D' = Delete
Barcode	3	24	Item Barcode, (right justified, zero filled on left - currently only the last 13 bytes used)
Description	27	30	Description (left justified, space filled on right)
Price	57	10	Unit Price 9,999,999,999 (right justified, zero filled on left)
Item Flags	67	6	'X_1 X_2 X_3 X_4 X_5 X_6' X_1 = 'Y', Tax plan A applicable = 'N' Tax plan A NOT applicable X_2 = 'Y', Tax plan B applicable X_3 = 'Y', Tax plan C applicable X_4 = 'Y', Tax plan D applicable X_5 = 'Y', Foodstamps applicable X_6 = 'Y', Reserved
LinkCode	73	24	Linked Item Code Right justified, zero filled on left (currently only the last 13 bytes are used)
Depart	97	3	Department Code (right justified, zero filled on left)
MixMatch	100	2	Mix & Match Number (right justified, zero filled on left)
Method	102	1	Pricing method: 4 = Unit Adjusted Pricing 3 = Group Adjusted Pricing 2 = Group Threshold Pricing 1 = Base + 1 Pricing 0 = Unit Pricing
Deal Price	103	10	Deal Price 9,999,999,999 (right justified, zero filled on left)
Deal Quantity	113	3	Deal Quantity (right justified, zero filled on left)
PSS Expansion Area			This area is reserved for future expansion by the PSS system. It's size is configurable and defaults to zero length.
User Expansion Area			This area is used for customized implementations of PSS. It's size is configurable and defaults to zero length.
Record Terminator	116	2	Carriage Return/Linefeed (0x0D, 0x0A)

D.4 Tax Table File

Filename	File directory path specified in PSS System Settings. The file name defaults to NXGTAXTxx.DAT, where 'xx' is a two digit number identifying which taxt table on the POS it represents
Source:	POS system
Description	
Type	ASCII Sequential
Record Length	Variable
Number of Records	Variable.

This file is not in standard fixed length record format. It is a variable record length file with comma-delimited fields. Up to ten separate tax tables can be defined using the file names NXGTAXT01-10.

Each Tax Table File contains the deltas for the tax brackets and amounts that are required to compute tax. The file is sequential with three different record formats.

Tax Table Record 1

Field Name	Type	Length	Description
	ASC	1	" (0x22)
DESCRIPTOR	ASC	18	Tax Table descriptor.
	ASC	1	" (0x22)
	ASC	1	, (0x2C)
TAXRATE	ASC	1-4	Start of repeat range for table (0-99.99).
	ASC	1	, (0x2C)
FIRSTTAX	ASC	1	True implies tax is collected on first range. False implies tax is not collected on first range. True = any non zero value, False = 0.
	ASC	2	CRLF (0x0D0A)

Tax Table Record 2

Field Name	Type	Length	Description
NUMRANGES	ASC	1-2	Number of ranges in the tax table.
	ASC	2	CRLF (0x0D0A)

Tax Table Record 3

Field Name	Type	Length	Description
RANGEDELTA	ASC	1-2	Delta amount for high end of bracket for this range.
	ASC	1	, (0x2C)
AMTDELTA	ASC	1-2	Delta amount for tax amount for this range.
	ASC	2	CRLF (0x0D0A)

D.5 Customer Information File

Filename	File directory path specified in PSS System Settings. The file name defaults to CUSTMTxx.DAT, where 'xx' is a two digit ID number.
Source:	POS system
Description	Contains the list of customers which may use the PSS system.
Type	ASCII Sequential
Record Length	Fixed, 291* (Actual length depends on configured expansion area sizes)
Number of Records	Variable, one record for each customer.

Each record represents one customer and information needed by PSS to identify the customer as a valid user of the system. The file is in ASCII format with each record being terminated by a carriage return/line feed delimiting character pair.

Field Name	Start Offset	Length	Comments
Record Type	0	2	Customer update record type = 09
Action	2	1	'A' = Add/Update 'D' = Delete
Customer card number	3	20	Customer loyalty card number, (right justified, zero filled on left)
Customer Name	23	40	Customer's name (left justified, space filled on right)
Customer address line 1	63	40	First line of customer address (left justified, space filled on right)
Customer address line 2	103	40	Second line of customer address (left justified, space filled on right)
Customer address line 3	143	40	Third line of customer address (left justified, space filled on right – populates PSS "City" field by default)
Customer address line 4	183	40	Fourth line of customer address (left justified, space filled on right– populates PSS "State" field)
Customer address line 5	223	40	Fifth line of customer address (left justified, space filled on right– populates PSS Postal_Code and Country fields)
Language Code	263	1	Language Code 0 = Use store language, 1 = First language, 2 = Second language, 3 = Third language If multiple languages are not used, set to zero (0)
Customer category	264	2	This field exists for compatibility with versions of PSA. It is availiable for use by User Exit routines
Enable/ Disable flag	266	1	Contains an optional flag to specify whether this customer is created enabled or disabled for shopping. 'D' indicates the customer is created disabled and must be manually enabled using the Service Terminal. 'E' indicates the customer is created enabled and may use PSS to shop. It the field is left blank, the customer is created disabled
Customer Greeting	267	20	This field updates the PSS Greeting_Text field in the customer record. It is displayed on the terminal at the start of the shopping trip in place of the customer name if supplied.
Checkchance Level	287	3	A numeric value that is used in determining whether a customer is sent to the re-scan or quick-pay checkout lane. (right justified, zero filled on left)
PSS Expansion			This area is reserved for future expansion by PSS. Default size is zero
User Expansion			This area is set aside for use by systems customized with user exit routines. Default size is zero
Record Terminator	290	2	Carriage Return/Linefeed (0x0D, 0x0A)