

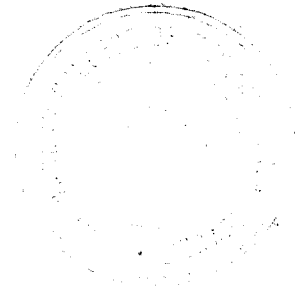
# ***DISTRIBUTED PROCESSING SYSTEM***

P- 298

***DISSERTATION SUBMITTED IN  
PARTIAL FULFILMENT OF THE REQUIREMENTS FOR  
THE DEGREE OF MASTER OF COMPUTER APPLICATIONS  
OF BHARATHIAR UNIVERSITY***

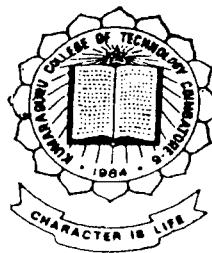
***BY***

***JAYASHREE .S  
REG. No. 9438MO196***



***UNDER THE GUIDANCE OF***

***Prof. P. SHANMUGAM M.Sc., M.S.,(Hawaii)***



***DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
KUMARAGURU COLLEGE OF TECHNOLOGY  
COIMBATORE - 641 006***

***JUNE 1997***

## DECLARATION

*I hereby declare that the Project Entitled*

### **DISTRIBUTED PROCESSING SYSTEM**

*Submitted to Kumaraguru College of Technology, is a record of the original work done by me under the supervision and Guidance of Mr. P. Shanmugam Professor and Head, Department of computer Science and Engineering, Kumaraguru College of Technology, Coimbatore and that this project work has not formed the basis for the award of any Degree Diploma / Associateship / fellowship or similar title to any candidate of any University.*

Place : Coimbatore

Date :

*Jayashree.s*  
(JAYASHREE .S)

Countersigned by  
Staff in Charge / Guide

Prof. P. Shanmugam, M.Sc., (Engg) M.S. (Hawaii)  
Professor and Head,  
Department of Computer Science and Engineering,  
Kumaraguru College of Technology,  
Coimbatore - 641 006.



Mascot Systems Pvt. Ltd.  
MASTECH (USA) GROUP COMPANY

1, MAIN ROAD,  
KASANDRA,  
SARJAPUR ROAD,  
RAMANGALA EXTENSION,  
BANGALORE - 560 034, INDIA.  
PHONE + 91 80 5521 701/702/703/705/706  
FAX + 91 80 5521 704  
EMAIL: mascot@india.mastech.com

### To Whomsoever It May Concern

This is to certify that **Miss. Jayshree S.** student of Master of Computer Application, **Kumaraguru College of Technology**, Coimbatore, has undertaken her project work with us from January 1997 onwards for a period of 22 weeks. She was assigned the project '**Distributed Processing System - Query On Inventory**'. This project was developed using Oregon Pascal. Her performance during this period was satisfactory.

Since this project was for an external client, we have insisted on maintaining confidentiality on any detail regarding the client including name of the client. Also, we will not be giving him the source code as per our company policy.

Yours Sincerely,  
For Mascot Systems Pvt. Ltd.

Ms. Kavita Khanna  
Manager Training & Selection

Mr. Anil Kumar  
Project Manager

# CERTIFICATE

*This is to certify that this project work entitled*

## **“DISTRIBUTED PROCESSING SYSTEM”**

*submitted to Kumaraguru College of Technology, Coimbatore (affiliated to Bharathiar University) in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications is record of original work done by Miss. JAYASHREE, Reg No. 9438MO196 during his period of study in the Department of Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore under my supervision and guidance and this project work has not formed the basis for the award of any Degree / Diploma / Associateship / Fellowship or similar title to any candidate of any University.*

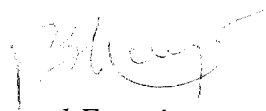


*Professor and Head*

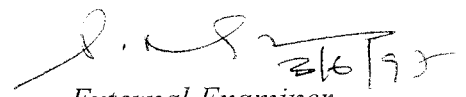


*Staff-in-charge*

*submitted for University Examination held on ~~03/06~~ 03/06/1997*



*Internal Examiner*



*External Examiner*

## ACKNOWLEDGEMENT

*I express my profound sense of gratitude and indebtedness to my guide Mr.P.Shanmugam M.sc(engg),M.S for his valuable guidance and constant help to do this project.*

*I am highly thankful to Mr.Ravi Joseph C.E.O of Mascot Systems Pvt.Ltd, for having given me the oppurtunity to carry out the project for their organization.*

*I owe a great debt of gratitude to Mr.Anil, Project Manager for his valuable guidance & encouragement at every stage of my project work.*

*Finally, I express my heartfelt thanks to everyone, who directly or indirectly helped me in the successful completion of this project.*

## **SYNOPSIS**

The project titled 'QUERY ON INVENTORY' is an application system package that runs on CC130 hardware and VERSADOS operating system. CC130 hardware is proprietary system of the client.

The system satisfies the request from the field for flexible query on inventory, which include the query based on partial keys, class and category.

Access to the query will be in 2 different ways. One by function key from the existing inquiry and other by a menu selection from the inventory maintenance menu.

The front end used is OREGON PASCAL.

The backend used is AS/400.

Operating system is VERSADOS.

Contents

1. Company's Profile.	1
2. Introduction	3
2.1 Overview of the DPS.	
2.2 Introduction to the QOI	
3. System Analysis	33
3.1 Description of the existing system.	
3.2 Problems of the existing system.	
3.3 Requirements of the new system.	
3.4 Processing Environment.	
3.4.1 System Architecture.	
3.4.2 Language Chosen.	
3.4.3 Database Structure.	
3.4.4 Operating System.	
3.4.5 Screen Utility.	
4. System Design And Development	50
4.1 Input Design.	
4.2 Process Design.	
4.3 Output Design.	
4.4 Database Design.	
4.5 System Development.	
5. Implementation.	82
6. Conclusion.	83
7. Appendix.	84
8. Glossary.	96
9. Bibliography.	103



*Company Profile*

---



### 1. COMPANY PROFILE

**Mascot Systems Pvt. Ltd.**, a two year old company came into existence as an off shore center for Mastech USA , the parent company. Mastech USA is \$100 million company having 16 branches all over the world. It is one of the fastest growing companies in US. Mastech has a very good client base. It is one of the **Fortune-500 Companies** and is expected to be one of the **Fortune-100 Companies** by the end of the year. Mastech has serviced and is serving 400 customers around the world.

Out of the 400 customers of Mastech, Mascot at present is servicing 30 customers. The Indian operations are 100% **EOU**. This business is made possible by the global presence of Mastech. Mascot is a **GLOBAL SOFTWARE SERVICE PROVIDER** and there lies the secret of its success in the software industry.

Mascot operates mainly on a **3x3** line of business principle. The three areas are

- Development
  - Maintenance and
  - Conversion / Migrations
- of commercial applications.

The three main working platforms are

- Mainframes
- Mid range, i.e. AS/400 and
- Client/Server Systems.

Apart from the above platforms now there are projects running on VC++ and Lotus notes. Mascot regards its clients as partners in a process of understanding. At Mascot, ongoing communication throughout the project cycle is maintained in order to be responsive to our client's needs. Constant communication is the basic foundation of Mascot's client

relationship. The vast geographical distance between the client and the Bangalore office is bridged by a full range of communication technologies that include phone, fax, electronic mail, voice mail and data exchange over a dedicated 128 kbps satellite link. Further the Lotus Notes based management and control system ensures that the client is always in touch.

## *Introduction*

---

## 2. INTRODUCTION

### 2.1 OVERVIEW OF THE DPS

The Distributed Processing System (DPS) is designed to provide a comprehensive data processing system run on distributed computers (i.e., computers in every store and warehouse). Communication between processors take place only when it becomes necessary to access or move data between stores and/or the central location, although communications links are up virtually at all times.

DPS is the application system package that runs on CC130 hardware and VERSADOS operating system. It has different subsystem that enables Sales Processing ( Order Entry), Inventory Management, Cash Management, Reporting, Personnel Data Management etc.

The benefits of a distributed system include :

- **Off-Line Operation**

The DPS architecture places the processing intelligence for a location at that location. The location's access to its own data and ability to perform sale transactions is not impacted by a failure of communications or of the central site processor. Only a few of the DPS functions, such as inquiry and sale of inventory at an alternate site, and credit card approvals, are impacted during operation "off-line".

- **Local Processing Response Times and Capacity**

With a local DPS node, the store is guaranteed dedicated processing capacity far exceeding its current share of the central machine. The store's productivity is enhanced by faster response time. Client expansion can continue with greatly reduced sensitivity to the real-time processing capacity limitations at the central site. Since most of the interaction occurs between the terminals and the local DPS node, telephone line capacity no longer constitutes a severely limiting factor.

- **Disaster Protection**

Currently, the centralized computer facility at Client headquarters constitutes some degree of exposure. A major site disaster would undoubtedly interrupt computer support for the stores for an uncomfortable length of time. DPS's capability to operate in an off-line mode would greatly reduce the impact of such a disaster, should it occur.

### **Order entry features**

Order entry is the most used function of DPS. The DPS design focuses on increased flexibility. Increased flexibility will eliminate the need for complex refund - reply of order entry operations.

### **Line Items Dispositions Of Orders**

DPS removes the restriction that all items on an order must be of the same sale type (Lay away , Special order or Sale )

- Lay away :

When a customer wants to make a down payment on an item in the store's warehouse , press the <LAYAWAY> Key. The customer agrees to make payments and receive the merchandise when it's paid in full.

- Special Order :

When a customer wants to pay a either a percentage or the full amount for an item not in a store's warehouse, but available from the Distribution centre, press the < SPC ORD> key. The customer will be notified when the product arrive in the store.

- Sale :

When a customer wants to pay full for an item that's in the store's warehouse press the < SALE > key.

Most DPS algorithms are designed to handle transactions at the line-item level rather than address orders as a whole.

**User Friendly Interface :**

- DPS uses block mode techniques for the user interaction in most functions, particularly Order Entry.
- An on-line help facility is also provided in DPS. At any point of time the user can request information by pressing a HELP key. The system displays pertinent data describing the current transaction and what is expected of the user.

**Releases Of Paid Items At The Line Level :**

- DPS allows items to be released on an order, provided that the amount paid is sufficient to cover the amount due implied by the release.
- By contrast the current system requires that the order be fully paid before any releases are accepted
- This reduces the need for refund and reapply transactions

**Exchanges :**

- DPS will support exchange scenarios by allowing returns and add-ons to the original orders.

**ESP Edits :**

DPS supports an ESP file that provides information to validate the type of merchandise to which the ESP is applied. DPS will not permit an ESP to be sold which is inappropriate to the merchandise sold.

**Rain Checks :**

DPS supports a line item type of Rain-check. Rain checks are permitted on out-of-stock, sale-priced merchandise. Rain-check items can be converted into sales with an appropriate discount at a latter date. Rain-check quantities are available to automatic replenishment.

**Sales Pricing Support :**

DPS supports the distribution of promotion (Mark down) files describing sale pricing at the store, market or region level. Prices are automatically adjusted at the beginning and end of the sale.

**Coupon Support :**

DPS supports a coupon payment type. A coded identification is expected to appear on each coupon when it is printed. When this code is entered, DPS derives the amount or percentage, ledger account number and restrictions on the use of the coupon.

**Special Order Edits :**

DPS provides special orders only for items not in stock at the specified inventory location. In addition, special order items must appear on the division item master as approved for special order.

**Deposit Sale :**

DPS supports a sale line-item type and inventory disposition called Deposit Sale. Deposit sale treatment is appropriate for items which are not available for release at the time of sale. Filled special orders are also treated as deposit sales, until fully paid.

## **Inventory management**

### **Inventory Databases**

Two systems exist for managing inventory data, one residing on the S/38 and the other within DPS. The S/38 manages the corporate financial aspects and distribution of inventory. The Dps design is oriented towards managing store inventory with a focus on sales.

An accurate inventory is present on the S/38 as a centralized database. If two databases diverge for whatever reason, the s/38 version is considered the correct one.

DPS maintains a location Item Master on each DPS node. This file contains information that pertains only to that location. A division Item Master is also maintained for a dummy location. This file is used for information about an item not in the location Item Master, Principally for special orders.

### **Inventory Dispositions**

In the DPS database, each item is counted in one of several dispositions meaningful to the retail operation. i.e., Layaway, Open box, etc. The disposition of an item designates whether it is saleable, non-saleable, or reserved for an open order.

### **Reserved and Tagged Items**

Reserved, Open box, and defective items are individually tracked in DPS, and are identified by a system assigned number. A unique Tag number is generated by Dps to track defective items and open box merchandise.



### **Open Order and Tag Files**

Each reserved or stock tagged item is represented in Dps by a record accessible by order or tag number respectively. These record support the various disposition counts in the location Item Master. The Order and Tag files contain this information on each Dps node. A Central, consolidated Open Order file is maintained on the S/38 to support automatic Replenishment and the Financial systems.

### **Variance**

DPS maintains a Variance (float) quantity for each store. Variance is a quantity used to offset locally entered adjustments to inventory. Counts in variance represent items lost, stolen, destroyed, or found (negative variance). The variance quantity allows the store to adjust inventory counts in real time, without diverging from the on-hand quantity in the Perpetual Inventory.

### **Inventory Information Data Flow**

Transactions affecting inventory quantities are captured on both the S/38 and DPS. All such updates must be communicated to the other system in order to maintain the two databases in parallel. Data flowing from the Dps nodes, through the HP-3000, and to the S/38 is said to be up-loaded. Conversely, data traveling in the opposite direction is said to be down-loaded.

- **Overlay Downloads**

Most movement of inventory information from the S/38 to DPS is accomplished via overlays. The S/38 system supplies DPS with an "official" on-hand quantity for items. When that occurs, DPS recalculates net saleable by backing out its non-saleable quantities. Overlay down-loads may be initiated on the S/38 system at any time that there is reason to believe that the quantities on the DPS database are incorrect.

- **Transmittals**

The S/38 Distribution system creates transactions affecting DPS inventory. These transactions, called transmittals, represent routine shipments of merchandise, chiefly to replenish store inventory that has been sold.

- **Adjustments**

Adjustments originates on the S/38. They are usually intended to correct errors that occurred in the sales or distribution processes.

- **Point - of - Sale Transactions**

The majority of transactions affecting inventory are captured through Order Entry. These transactions are up-loaded to the Consolidated Transaction Log described above. All sales are processed on the S/38, and the indicated updates to inventory are made on the S/38 inventory databases.

- **Transfers**

Certain inventory movements Client locations occur independent of the S/38 based Distributed System. These transactions are called Transfers, and are captured in DPS. Transfers are communicated through the up-load of the Consolidated Transaction Log.

- **Write - Offs**

DPS provides a high security transaction, described above, to write-off lost of damaged items. The write-offs transaction is only performed against a positive variance or items tagged defective. The write-off transaction is communicated to the S/38 via the Consolidated Transaction Log.

### **Timing Considerations**

Most transactions affecting the inventory of a store or service location (sale, return, etc. ) are captured at the DPS node supporting that location. These transactions affect the DPS inventory in real time. They only affect the S/38 inventory when the Consolidated Transaction Log is processed nightly.

Most transactions affecting distribution center inventory originate on the S/38 and the associated network. These include receiving of vendor shipments, adjustments, and shipping of transmittals. Overlays are batch transferred nightly from the S/38 to the HP-3000 system and distributed to update the distribution center inventory.. Down-load of adjustments to the Distribution Center is unnecessary since quantities are reset nightly by overlay.

### **Receiving Variances**

Receiving transactions ( other than vendor receiving ) are captured at the receiving location through DPS. A receiving variance is a reported discrepancy between the items and quantities shown on the transmittal or transfer transaction and those physically in the shipment. Receiving variances on transmittals are posted to DPS as variance, and do not affect the S/38 inventory.

### **Database Controls**

To ensure that the DPS store inventory and the S/38 databases do not diverge, the two databases can be synchronized through overlay, as is done with the distribution center. While it is not reasonable to overlay on-hand nightly at every location, this can be done on a rotating basis or initiated if some DPS data becomes suspect.

### **Physical Inventory**

Client takes a manual inventory twice annually to verify and correct the perpetual inventory. Several days prior to physical inventory, transfer shipping is disabled throughout the DPS network to ensure that all in transit items have been received when the counts are taken. DPS supports transfer shipping disable and re-enable through a down-loaded transaction.

Once the Physical inventory is processed on the S/38 , quantity overlay downloads are generated for each store. The physical inventory quantity download does not include records for zero quantity items. Each file of quantity downloads contains a record which triggers a process to zero the on-hand and variance quantities in all location Item Master records.

### **Summary**

In summary, the DPS inventory is designed to agree with the S/38 database. The store management has the option of submitting transactions to change the dispositions of merchandised charged to that store. This capability allows the store control of the Net Saleable, without affecting the integrity of the perpetual inventory.

### **Cash register controls**

The following describes the transactions and controls associated with the management of the cash registers.

### **Multiple Cash Drawers**

By assigning a separate cash drawer to each cashier, DPS supports accountability of each cashier for the money in their drawer. The CC-130 can support up to 64 cashiers on duty concurrently.

The DPS initialization file contains the relationship between a register CRT and its associated cash drawers. The File is maintained at the store level by the service technicians. Since the file is used to assign drawers to cashiers and to locate the drawer to be 'popped out' during a cashier transactions, the technicians must be careful to assign to a CRT only cash drawers which are physically located below the CRT.

The cluster of hardware that includes the register CRT, cash drawers and printers as the REGISTER CONFIGURATION

Cash drawer security is mainly dependent upon cashier access-key and physical access to the register. Cashiers must ensure that their access-key remains secret.

### **Store Open and Order Entry On**

At the beginning of the day, when the DPS batch processing is complete, the DPS menu is displayed but all options, except STORE OPEN are disabled. Selecting STORE OPEN unlocks all other menu options except those requiring the order entry option to be on.

Selecting store requires STORE OPEN function requires STORE OPEN license, which is usually assigned to the store manger. In addition, no sale or register functions can be performed until the ORDER ENTRY ON option is selected, also a manager function.

When the STORE OPEN is selected, the contents of the STORE-MASTER file are read into a global memory area and a STORE OPEN record is posted to the tag file.

## **Cashier Open**

Cashier open performs the set-up required to enable the user to perform Cashier functions. Selecting the CASHIER OPEN function requires CASHIER licence, and can only be performed on a CRT configured as a register

When a user selects CASHIER OPEN on a processing day, DPS allows the user to select an available cash drawer, and prompts for BEGINNING-BANK amount. The transactions is not allowed, of course, if all cash drawers associated with the terminal are already use. DPS then sets up a cashier table entry to track the tender typed and amounts for which the cashier is responsible.

## **Tender Transactions**

An Order Entry sale may be created at either a floor CRT or a register CRT; however, if payments of ticket printing is involved, the sale can be tendered only at a register. When any Order Entry transactions is tendered, all payment totals are accumulated in the Cashier Table entry associated with the cashier's employee number. DPS thus tracks these amounts by cashier and payment type,

Tendering a non-trivial transaction requires TENDER licence. In addition, DPS only permits the cashier to tender transactions at his/her assigned register CRT.

## **Remove Cash Drawer**

The REMOVE CASH DRAWER function allows the cashiers to remove their cash till and release the Cash Drawer at the register. This function is used to switch drawer locations or to store the till while retaining tender totals.

Remove Cash Drawer removes the Cash Drawer number from the Cashier Table entry. The REMOVE CASH DRAWER function should be performed by the cashier assigned to the drawer.

### **Drawer Lock**

The DRAWER LOCK function allows someone with a CASHIER or CASHIER CLOSE licence to lock the CASHIER'S Table Entry. The result will be that no further activity can occur for this cashier without first closing and reopening. The cash office manager uses this transactions to assure there is no activity by a cashier while the CASHIER CLOSE is in progress.

When the drawer lock is selected the screen displays all open cashiers and allows a single cashier to be locked, or all cashiers to be locked.

### **Cashier Close**

The CASHIER CLOSE function is performed by the cash office manager at the end of the cashier's shift. The purpose of this function is to verify the contents of the till. Selecting

the CASHIER CLOSE function requires CASHIER CLOSE licence, which is usually assigned to the cash office manager. Completing the Cashier Close-Out posts a CASHIER CLOSE-OUT record to the log file and generates a printed cash-out report.

Totals from the log record are added to the cashier close-out record.

DPS requires a Cashier Close transaction for each cashier with activity during the day. The Store Close transaction is not allowed unless all Cashiers have closed.

### **Store Close**

The STORE CLOSE function is performed by the manager at the end of the business day. It signals that all Dps transactions are finished and the files are ready to be processed by the nightly batch system.

Access to the STORE CLOSE function requires a STORE CLOSE licence, which is usually assigned to the cash office manager or the store manager.

### **Incomplete Processing Days**

Several conditions may occur that prevent an entire day's transaction log from reaching the S/38 accounting and inventory systems by the start of central batch processing. A node or communications failure can interrupt the flow of log records at any point during the day, and may not be corrected before cut-off. Similarly, as described above, a store may not complete its Store Close before a predefined time at which batch processing must start.

In the latter case all unclosed cashiers are closed automatically, after which the store is automatically closed. The log records are marked as being closed by time-out. The next morning before the store be closed, each cashier closed by time-out, must be closed by the cash office manager. The store must also be closed properly.



When the problem is resolved, be it hardware, store management, or communication, the balance of the incomplete day's activity is logged. In the latter case, the log records reflect the day of the actual activity. Otherwise, they are logged with the current (i.e. following) day's date, when they were actually performed at the node.

The S/38 treats all late arriving transaction log records from the incomplete day as if they occurred the following day. Essentially, records that were held up due to communication are re-dated. The records of transactions that were actually entered late are not easily distinguished from the current day's activity. The cash management reports then reflect prior day activity combined with the current day. The net of the two days, however, should balance properly.

### **Store reports**

This section describes reports available at the DPS location. ALL reports produced on the local node are included. Some centrally produced ( and downloaded for printing ) reports, which are integral to DPS processing, are also discussed.

#### **Cash management report**

These reports are produced nightly, or on demand, using the Cashier Close-out file.

- **Cashier Cash-Out Report**

The Cashier Cash-Out Report summarizes tender amounts by payments type and cashier. Information on the report includes total number of transaction, amount and

variance of each payment type tendered. The report is automatically generated by completing the CASHIER CLOSE-OUT function.

- **Store Cash-Out and Deposit Report**

The Store Cash-Out and Deposit Report summarizes the store's tender amount totals by payment totals by payment type. The report is essentially a summary total of the individual Cashier Cash-out Reports and is similar in format. It is generated by the STORE CLOSE function.

## **Finance contracts**

The following reports are now produced by the existing Finance Contract System

- **Suggested Transmittal**

All financed sales involve paperwork which is required by the finance company and must be sent in before the finance company issues payment. The Suggested Transmittal report, sorted by finance company, lists financed transactions that need to be processed by the store's credit office and sent to the finance company. Information on the report includes customer name, ticket number, release date, customer finance account# and ticket totals.

- **Transmittal and Assignment of sales Invoices**

All suggested Transmittals are loaded inn the HP's finance transmittal file each morning from information down-loaded from the S/38 A/R tápe. The Credit office is

responsible for completing finance paperwork and selecting transmittals which are to be sent to the finance company from a screen provided for this purpose. The report lists customer information similar to the Suggested Transmittal Report and includes final totals for amounts financed, adjustments and total due store. This report is then forwarded to the finance company for reimbursement.

- **Transmittal Trial Run**

The Transmittal Trial Run report is used by the credit office to proof read the selected-transmittal work before “finalizing” a group of transmittals. Information on the report is the same as the final version but omits the final totals since they are not valid.

- **Open for Payment**

After a finance transmittal has been selected and sent to the finance company, it has an Open for payment status in the Open Order record until it is paid off. The Open for Payments lists all financed contracts awaiting payment from secondary sources. Information on the report includes customer name, ticket number, release date, transmit date, finance account number, amount financed and ticket age in days. The report is generated each morning from information down-loaded to the Hp from the S/38 A/R tape. The report is then passed to the CC130 to be printed at the store level.

## **Management control reports**

- **Manager’s Approval Reports**

The manager's Approval Report is a series of reports designed to call attention to transaction line types or line changes which require manager review due to potential for fraud or accidental misuse. The reports are generated daily during nightly batch processing and are sorted by cashier number and ticket number.

A report is created for each of the following categories :

- No-history
- Petty Cash
- Price/Spiff Changes
- Refunds
- Price Adjustments
- Returns
- Delivary charges
- Wards Changes
- Manually Entered Credit Cards
- Inventory Maintanance
- Tax Exempt

Information on each individual report varies slightly from reprot to report, since some data ellements may not be present on all orders. Data elements reported are :

- Cashier#
- Ticket number and type
- Customer name
- Ticket total
- Tender time
- Detail info of each CBM - Class,Brand and Model
- Authorizing employee number

## **Sales reporting**

- **Daily Processed Sales Report**

The Daily Processed Sales Report details processed sales for the day by employee, along with the employee's month-to-date totals. The information consists of customer name and phone#, ticket#, transaction type, QBM, price, gross margin %, and spiff amount. A summary following the item details shows the daily and month-to-date totals for the number of tickets, merchandise dollars, gross margin % , ESP amount, Esp % of sales, and spiff amount.

- **Daily Processed Sales Summary**

The Daily Processed Sales Summary is a three-part report that lists salesman totals for the day, store totals broken down by category, and service charge totals

- **Daily Written Sales Summary ( Flash Sales )**

The Flash Sales Report is a two-part daily written sales summary of salesman and store productivity. The report is computed and printed upon request from the store. The data is obtained from the Flash totals maintained on-line in the Employee Table.

- **Promotional Sale Pricing**

The promotional Sale Pricing Report lists current and pending promotional pricing by CBM. The report is generated at the store daily or upon request. It contains information such as SALE-BEGIN and SALE-END dates, normal price, sale price,

cost, spiff, and whether the sale price was generated by corporate, market or store personnel.

## **Open order reports**

These reports are produced by batch processing of the Open Order file.

- **Open For Delivery Report**

The purpose of the Open for Delivery Report is to control reserved merchandise by listing all delivery items which are fully paid but not yet released. The report is generated by store/warehouse batch processing of the order file.

- **Open For Pick-Up Report**

The purpose of the Open for Pick-Up Reports is to control reserved merchandise by listing all pick-up items not yet released. The report is generated by store/warehouse batch processing.

- **Balance Due Report**

The Balance Due Report provides the Store Manager with an overview of his store's Open Orders which are not fully paid and may require attention. The report contains information on all Open Orders. Orders are separated by age into groups 30, 60, and 90+ days.

## **Rain - check reports**

- **Rain check Customer Notifications**

Raincheck customer notifications are generated in the nightly batch processing. Any raincheck order for which the merchandise has become available on that day, will have a customer postcard printed which states that the merchandise is now available.

- **The Raincheck Report**

The raincheck report lists raincheck orders grouped into three categories. Rainchecks as confirmed, available, or unconfirmed. The report provides a page for each salesperson.

- **Release Detail Report**

The Release Detail report lists all items that were released that day. It is intended to provide a log of all released merchandise for later reference.

Each released item is printed on the report at both the sale and release location. Included on the report are customer name, ticket number, release date, quantity, brand, model, and salesperson(s).

This report also includes failed releases items and the associated failure reason code.

## **Release handling**

Merchandise releasing is the process of conveying physical merchandise to the customer. The release transaction is of inventory and financial accounting significance. The transaction representing the release reduces inventory and converts liabilities to income. The releasing also applies to returns of merchandise.

The term "Written Sale" is used to refer to a sale before merchandise release. At the time of release, it becomes a processed sale. Profit from the sale is not realised until it is processed .

### **Release Types**

DPS Supports six release types for merchandise listed as follows.

- 1) Carry
- 2) Service
- 3) Today Pick-up
- 4) Pick-up
- 5) Delivery
- 6) Fetch (Retrieval)

- Carry releases takes place at the point of sale. For line items of type sale, carries are assumed to be released when the transaction designating the item as paid-in-full is tendered. No release transaction is required.

When a layaway, special order or deposit sale line item has a release type of carry, is paid in full, and is available for release, DPS assumes the release of the item at that time. If the customer does not want the items at that time, the item(s) should be assigned a release type of pick-up. These items are then tagged with a reservation tag which prints at the pick-up printer.

- A pick-up or delivery release type is designated for merchandise which cannot be given to the customer in over-the-counter sales. Physical size of the merchandise, location of the merchandise (within the store or at the alternate location), insufficient payment or merchandise availability are all reasons for selecting a pick up or delivery release type.
- There are two variations of the pick-up release type. For some day merchandise pick-up, the release type is entered as today pick-up. For today pick-ups no release line is generated and no reservation tags print, but a release receipt prints and a release transaction is required.
- Future day pick-up are simply referred to as pick-up. They Require a release line, reservation line, reservation tag, release receipt and release transaction.
- The Fetch (Retrieval) release type is used to indicate that return is planned and that the merchandise must be picked up from the customer by the store's delivery service.



Service release type is used to indicate that a return of merchandise is being done and that the customer's merchandise item is at a service center awaiting repair.

## **Order entry processing**

DPS allows mixtures of release types and release dates on an order. Each order contains information which constitutes a schedule for the releasing of fully paid and available items on that order.

When each item is entered, a release type is specified or defaulted. Subsequent to the creation of the order, release date changes, addition of items to an existing scheduled delivery and other changes are allowed, providing the following the following rules are observed.

- The transaction cannot be tendered if the minimum payment due exceeds the total payments.
- Addition of releases with current or previous dates is not allowed.
- Changes to current date schedules can only be done at the release location

## **Delivery Scheduling**

The use of customer home delivery resources is controlled through delivery schedule allocations. DPS provides a mechanism whereby the selling location is forced to confirm to delivery "slot" allocations specified by the delivery location.

Changes in delivery dates on an order are handled through the normal mechanism by which inventory is released and reserved. A batch clean-up process purges the delivery location Item master of past delivery items.

## **Tracking and processing of scheduled releases**

### **Release Transactions**

DPS is informed of merchandise release through the merchandise release transaction. The release transaction provides three basic functions.

- To acknowledge the release of merchandise.

- To change or fix a prior release error.
- To indicate the failure reason for an expected release

When a delivery fails to complete normally on the scheduled date, an indication of the reason is entered in the release transaction and is stored in the order record. Reasons for the failure is entered as a three character code . Possible failure examples are as follows.

- CNH - Customer not at home.
- CRM - Customer refused merchandise.
- N I S - Merchandise not in stock.
- CNS - Customer no-show(pick up ).

Completed releases are posted to open order record. A release detail report lists all releases daily.

### **Merchandise Retrieval**

Merchandise fetch (retrieval) is treated as a special case of delivery. These items appear on the release ticket but are, of course, not included on the pick documents.

All completed fetches for orders with a credit balance appear on the balance due report, so that any refunds can be issued if appropriate.

### **Menu and access security**

Access to interactive DPS functions is controlled by the DPS menu. DPS initially a default menu on each interactive terminal. In terms of the network, the terminal is assigned a default connection ( virtual circuit) to the Menu task. This connection each time the terminal user exists an interactive task.

- **Function Codes**

To access a DPS function through the menu, the user enters a function code (1 to 4 characters), or presses a function key. The code of key appears on the menu screen beside

the desired task. the menu tack then requests a virtual circuit ( logical connection ) between the user's terminal and the task which performs the desired function. Any task access is, of course, subject to security check, a function that is also implemented in the Menu task.

- **Menu Selection Forms**

The part of the menu screen listing the functions is called the Menu Selection Form. Any one of several Menu Selection Forms may be presented to a user. Each form contains a sub-set of the functions available. The default form for a terminal is chosen to present the functions most likely to be used at that particular terminal. It is specified code regardless of which Menu Selection Form is displayed, but function keys are only when they appear on the current form

The user may select a function that causes a different Menu selection Form to be displayed. The EXIT function key is used to return to the default form for the terminal. A program returning control to Menu can also specify which Menu Selection From appears.

- **Sign-On and Access Key**

All users are required to sign on to Dps before accessing any functions beyond the menu. Initial user sign-on consists of entering an employee# and personal access-key on the Menu screen. Since the access key is used to identify the user, each time the user selects a menu option his access-key must be entered.

Access security in Dps is implemented through a set of licenses, which are assigned to DPS users. Licenses no longer are identified by job title, but rather by function. For example, instead of a " manager" access level there is "store open" licence, "inventory maintenance" licence, "price change" licence etc. This division of access by function allows greater flexibility in assignment of licenses.

the desired task. the menu task then requests a virtual circuit ( logical connection ) between the user's terminal and the task which performs the desired function. Any task access is, of course, subject to security check, a function that is also implemented in the Menu task.

- **Menu Selection Forms**

The part of the menu screen listing the functions is called the Menu Selection Form. Any one of several Menu Selection Forms may be presented to a user. Each form contains a sub-set of the functions available. The default form for a terminal is chosen to present the functions most likely to be used at that particular terminal. It is specified code regardless of which Menu Selection Form is displayed, but function keys are only when they appear on the current form

The user may select a function that causes a different Menu selection Form to be displayed. The EXIT function key is used to return to the default form for the terminal. A program returning control to Menu can also specify which Menu Selection Form appears.

- **Sign-On and Access Key**

All users are required to sign on to Dps before accessing any functions beyond the menu. Initial user sign-on consists of entering an employee# and personal access-key on the Menu screen. Since the access key is used to identify the user, each time the user selects a menu option his access-key must be entered.

Access security in Dps is implemented through a set of licenses, which are assigned to DPS users. Licenses no longer are identified by job title, but rather by function. For example, instead of a " manager" access level there is "store open" licence, "inventory maintenance" licence, "price change" licence etc. This division of access by function allows greater flexibility in assignment of licenses.

- **Global Network Access**

Most users access the system through devices wired directly to a node at their location. Some users, however, are granted a special access privilege which allows them to access locations on nodes other than the one to which they physically connect. Certain nodes are also configured with auto-answer modems for dial-up access.

Since these users may enter the network through any of a number of nodes, and can potentially contact a number of locations, the locally maintained security that supports the majority of users is not adequate. Instead, DPS provides a function which allows the user to logically connect to the menu task on another "foreign" node. Once established, that connection allows the execution of any function to which the user has licence on that node.

- **Passwords**

There are four key data elements in the DPS access control mechanism. These are :

1. Employee number.
2. Access - key.
3. Licenses.
4. Password.

A password is used to secure access to functions where exposure to fraudulent misuse exists, or erroneous or unauthorized use could seriously impact store operations. Employees with licences to access such functions must have a password assigned, in order to make use of those licenses.

The password is longer than access key up to 8 characters are allowed

- **System State Access Restrictions**

In addition to licence based security, Menu restricts access to functions according to system state. This means that the ability for a user to select a function depends upon the status of a related function or system operation. For example Order Entry On must be selected before Order Entry or Register Open. Additionally, system states such as Backup, Batch, Recovery, and Power-down restrict access to functions.

- **Personnel Files**

Personnel files are stored at the node level and consist of all pertinent employee information, such as employee#, access-key, security licenses, employee name, month-to-date sales totals, password etc. When an employee first signs on for the day, he enters his employee# and access-key. DPS reads the employee's personnel record to verify the access key and retrieve the security licenses.

- **License Maintenance**

As mentioned above, an employee's initial set of licenses are assigned by store personnel, except for the Store Manager. DPS supports license maintenance through an option of personnel maintenance. Access to this features of Personnel Maintenance is limited to users with "Delegate Licenses" license. This license is intended for managers. Users of this function may only delegate licenses that they possess.

The DPS design has focused attention on three areas : *Inventory Integrity, Ease of Use, and Security.*

### Inventory Integrity

1. The Stock Tag facility supports designation of merchandise items as defective or open box (saleable). A uniquely numbered stock tag is produced for such items. This information remains associated with the item for its life in inventory. When a tagged item is sold , the price and Spiff are retrieved from the tag record, thus no management intervention is required.
2. DPS provides Store Location Management with a tool for adjusting on-line inventory quantities by assigning them to or from a variance disposition. Variance implies that the quantity specified is either lost, stolen, destroyed, or otherwise unavailable for sale, it has been found and is in excess of the quantity that DPS currently has recorded. DPS maintains a detailed audit-trail of transactions affecting variance.
3. DPS provides reporting to support inventory counts to insure inventory integrity and security against merchandise losses.
4. DPS updates open order records when items on the order are affected by Transfers or Transmittals. Deliveries that require transfer of items to the delivery location can be better controlled. This change also allows replenishment for special orders that are not yet released.
5. DPS supports a relationship between Brand/Models which makes them interchangeable for sales purposes, but retains the distinction in inventory. Substitutions of equivalent items are made automatically during Order Entry, based on item availability in inventory and on a selling priority.

## System Security Enhancements

DPS provides for a much enhanced security facility to ensure that transactions with fraud exposure are executed only by authorized individuals. Better reporting and Cashier accountability are also supported for improved management control. The features listed here address security directly. Features described in other sections, such as Stock Tags and Order Entry flexibility also have positive implications.

- **Access Security Enhancements**

Each DPS user is assigned a set of licenses, each of which allow him to perform a set of operations. The exercise of certain licenses is protected by a longer password than the current two character id. Once signed on, the user's two character access code is sufficient for routine transactions, but the longer password is required for secured operations.

- **Refund In-Kind**

DPS imposes restrictions on the type of refund chosen when non-cash payments were made originally. In general, the refund would be of the same type as the payment. This feature prevents simple fraud scenarios where an invalid credit payment is refunded in cash.

- **Multiple Cash Drawer Support**

DPS associates a cash drawer with a Cashier rather than with a register hardware device as does the current POS. Thus a single CRT may be assigned multiple cash drawers. This change supports Cashier accountability.

## Ease Of Use



1. DPS supports Block Mode techniques for user interaction in most functions, especially Order Entry. Descriptive error messages and prompts are provided, so users will need to remember fewer codes and procedures.
  
2. An on-line help facility displays pertinent data describing the current transaction and what is expected of the user. At any point in the transaction, a user may request information by pressing a HELP key.
  
3. DPS increases Order Entry flexibility .
  - a. DPS sale transactions are handled at the line-item level rather than addressing orders as a whole. Different transaction types, i.e. Sale, Layaway, and Special Order, may exist on the same order.
  - b. Up to five different paytypes may be used to tender an order in DPS, which includes the use of multiple accounts of the same credit card.
  - c. DPS allows returns and new sales on original orders.
  - d. Merchandise to which an ESP is applied is validated, thus not permitting an inappropriate ESP to be sold.
  - e. DPS supports the distribution of promotion files containing sale pricing and descriptions to the store, market, or region level. Prices are automatically adjusted at the beginning and end of the sale.
  - f. Special orders are permitted for items not in stock at the specified inventory location or for special items not normally stocked at the location.
  
4. The DPS design also includes flexibility during the release of merchandise.
  - a. An order can have multiple release transactions allowing the store to accurately relieve its inventory when a customer receives their merchandise.
  - b. Each delivery item is associated to a delivery line-item containing delivery date and delivery charge. All delivery charges are accumulated on a single delivery line-item.
  - c. DPS limits the number of deliveries specified for a given location on a given day.

- d. DPS provides the facility to schedule a pickup of merchandise from the customer by the delivery team.

## **2.2 INTRODUCTION TO THE QUERY ON INVENTORY APPLICATION**

The design is concerned on query on inventory. Currently the user can query the system for details of a particular item in stock during processing an order from a customer. For this purposes, the user needs to know the full brand and model identification of the item that he wishes to find out about. The new facility that is to be offered for the query removes this restriction. The user can now find out the details of the items on various fields, and can supply partial information for the query.

## *Systems Analysis*

---

### 3. SYSTEM ANALYSIS

#### 3.1 Existing System

1. Currently On-Line Inquiry is accessed by pressing the INQ key where the Brand & Model has to be completely entered in order to query an item.
2. The result of the query is the quantity available in different dispositions ( such as closed box, open box , display etc.)
3. Partial values for Brand & Model is not allowed & no other fields can be used for querying.

#### 3.2 Problems of the existing system :

- Search on inventory should be made more flexible by providing options so that query can be based on Class and Category apart from the Brand and Model, and also allow partial values .
- User will not be able to query the inventory using partial values of Brand/Model.
- To know the details of several items user has to go through the entire search process each time.
- User has to remember the entire code for Brand/Model for querying.
- User is not allowed to query the items using Category and Class.
- Extended Search facility is not available in the existing system.

#### List of fields & their combinations allowed for a Query on Inventory :

Existing combination	Expected combinations
<i>Brand, Model</i>	<i>Category</i>
<i>Complete Values Should be entered for</i>	<i>Class</i>
<i>both</i>	<i>Class, Brand*</i>
	<i>Class, Brand, Model*</i>

fields in order to Proceed Search on  
inventory

Brand\*  
Brand, Model\*  
Model\*

'\*' Allows Partial Values in these Fields.

### 3.3 Requirements of the New System

#### Entry point for application :

One entry for this application will be by pressing the function key F2 while in the item inquiry screen invoked from order entry. If the F2 key is pressed in the item inquiry screen from release acknowledgment application , then the message 'Extended search facility is not available from this application' will be displayed. The adjusted and modified item inquiry screen are shown figure 2.1 and 2.2 respectively.

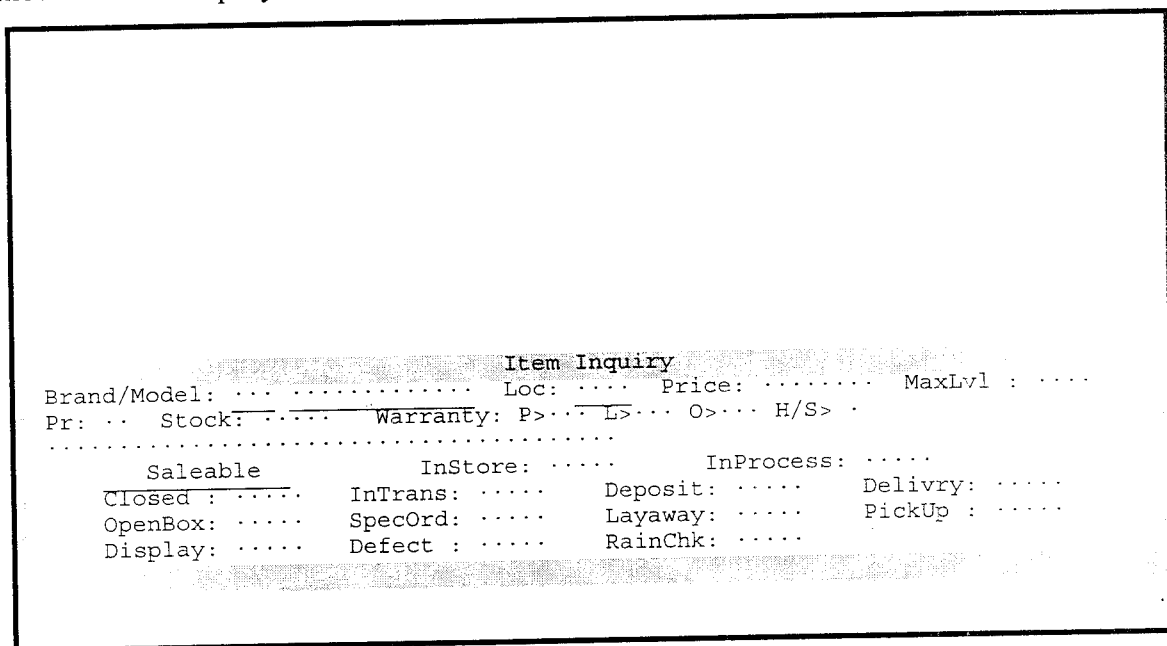


Figure 4.1. Existing item inquiry screen

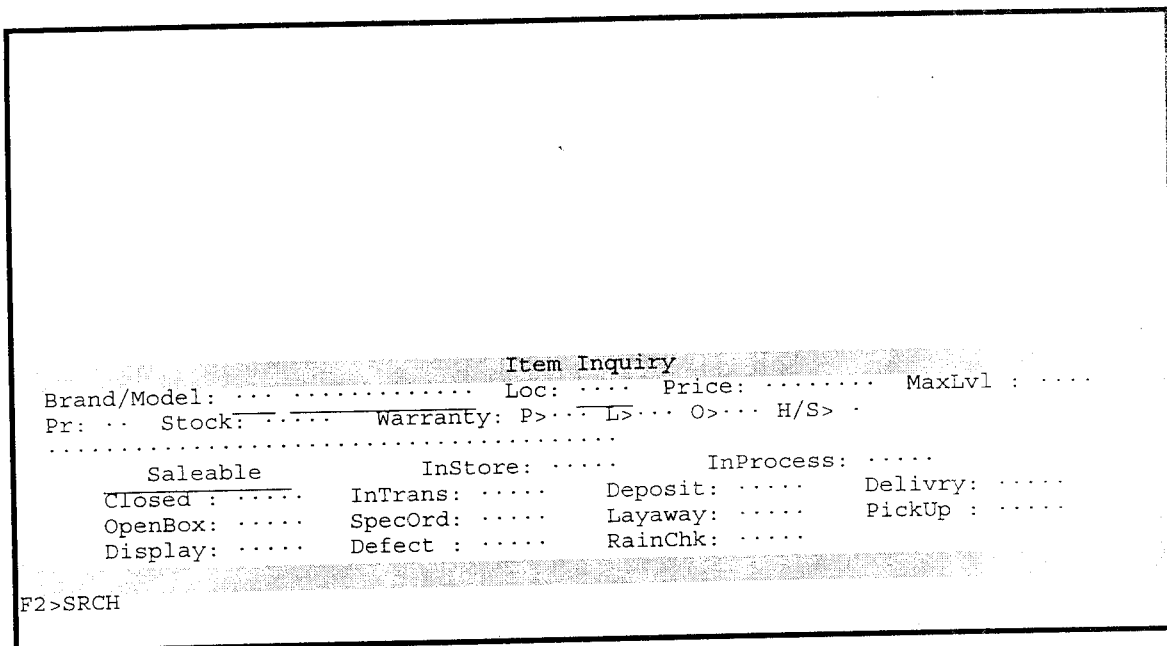


Figure 4.2. Modified Item inquiry screen with function key for extended search

Another entry point for this application will be from a menu selection from the inventory maintenance menu. The existing and the modified menus are shown in figure 4.3 and figure 4.4.

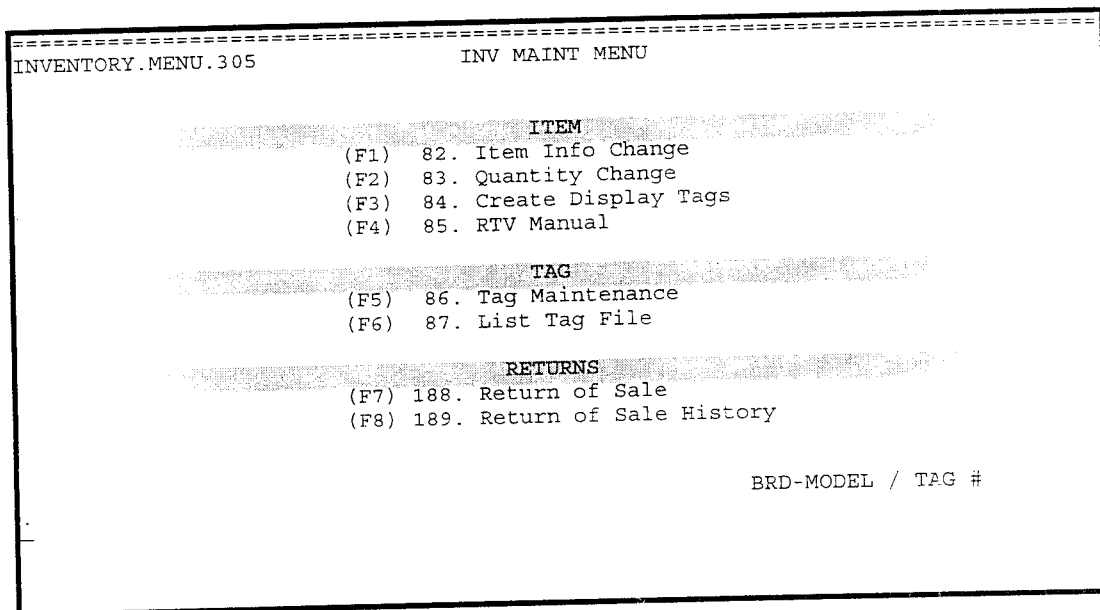


Figure 4.3. Existing inventory maintenance menu

```
=====
INVENTORY.MENU.305                               INV MAINT MENU
=====
                                     ITEM
(F1) 82. Item Info Change
(F2) 83. Quantity Change
(F3) 84. Create Display Tags
(F4) 85. RTV Manual

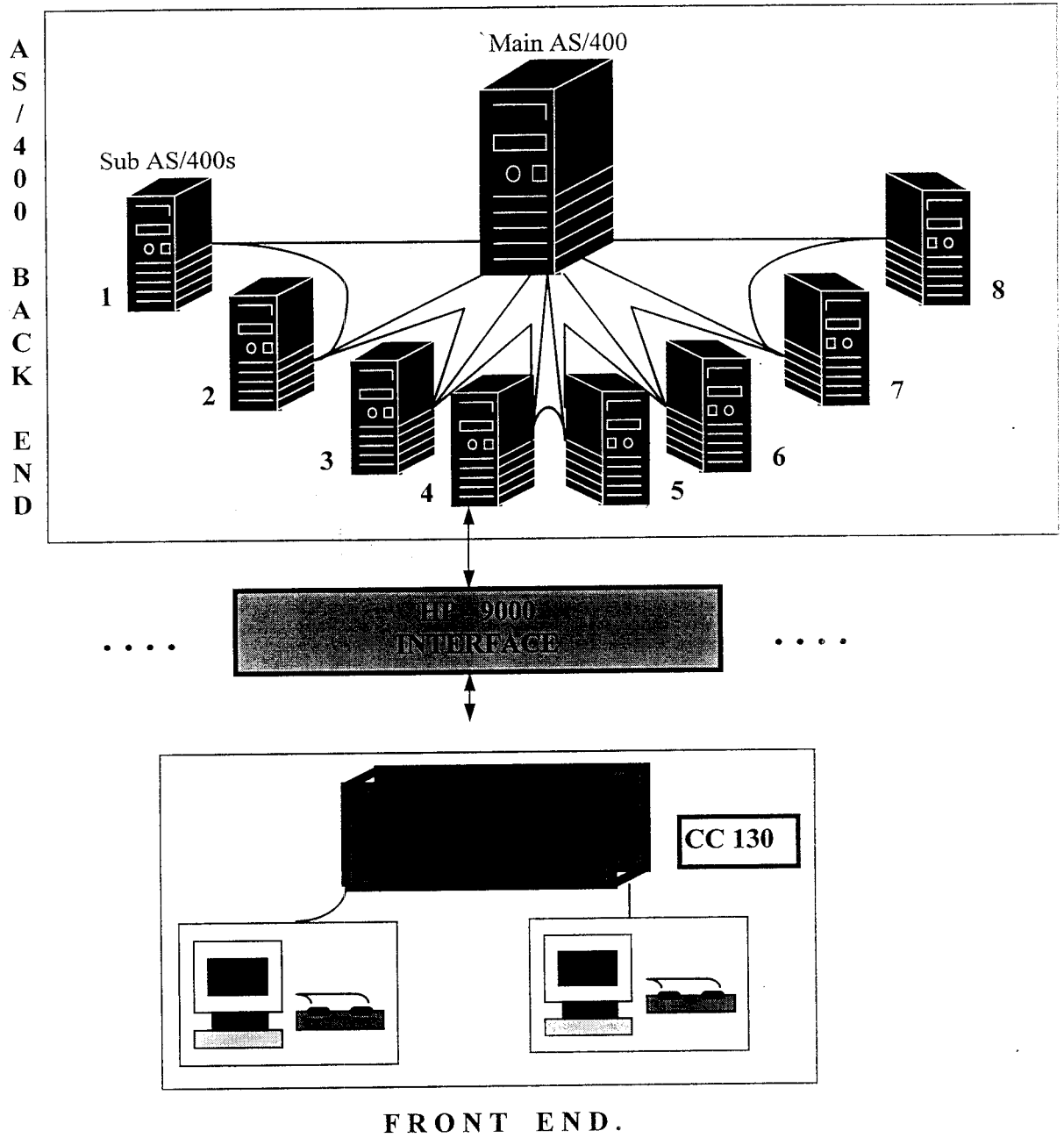
                                     TAG
(F5) 86. Tag Maintenance
(F6) 87. List Tag File

                                     RETURNS
(F7) 188. Return of Sale
(F8) 189. Return of Sale History

                                     SEARCH
(F9) 295. Search On Inventory          BRD-MODEL / TAG #
=====
```

Figure 4.4. Modified inventory maintenance menu

### 3.4.1 System Architecture





The system architecture of the Client is divided into 3 subsystems ,

1. *The Back End* - The Back End has 8 AS/400 machines which act as sub AS/400. These 8 machines are interconnected and are also connected to the Main AS/400.
2. *The Front End* - The Front End has a Proprietary Motorola System (CC130s) . A Max of 64 terminals can be connected to the CC 130.
3. *The Interface* - Each CC130 is connected to an HP-9000 which acts an interface to the Central Database on the AS/400 at the Client's place.

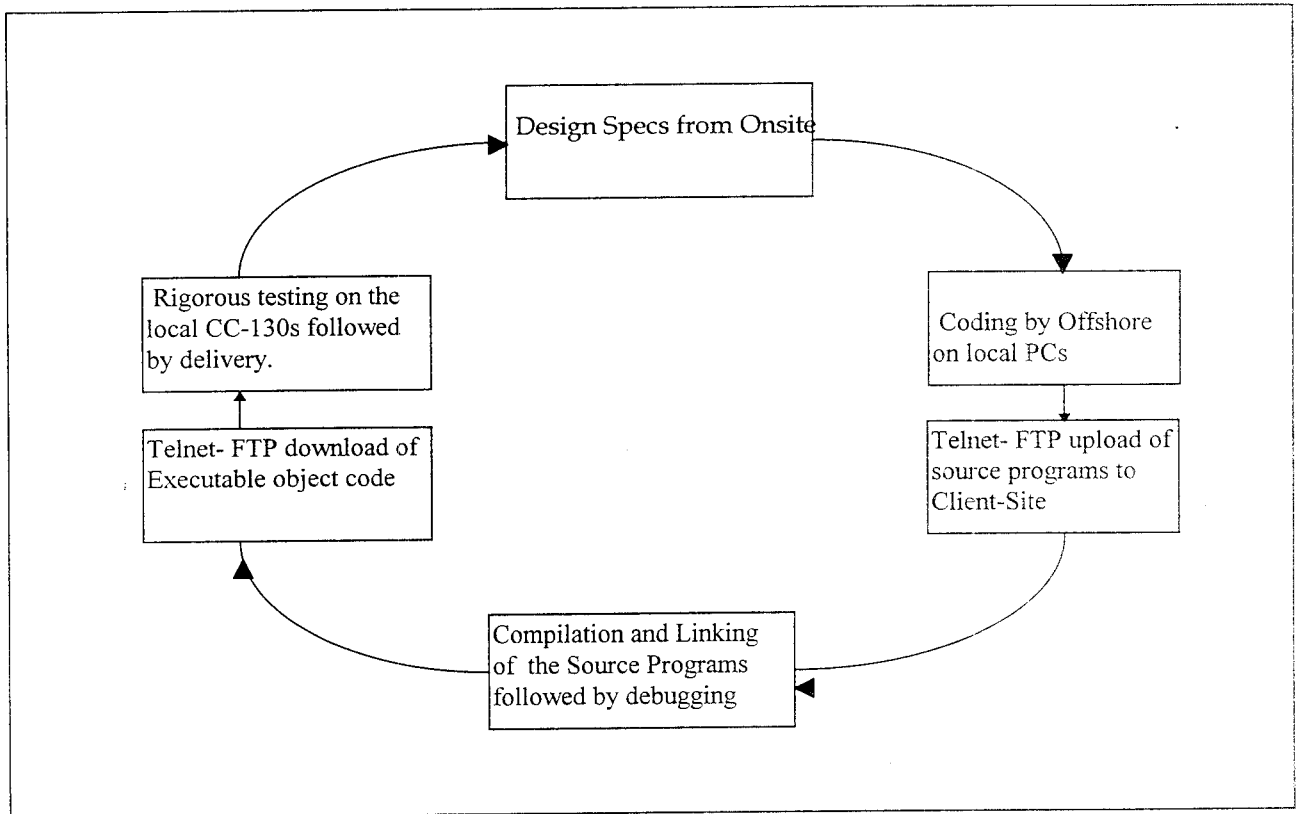
Most of the real time interaction is between the terminal and the application within the local node at the outlet. Inventory and Order Processing Databases are distributed locally onto each CC130. All transactions as done in the store are put up onto a log file, each of which in turn go to the HP-9000. This is a process totally masked from the terminal user (CSA/SC).

End Of Day (EOD) processing is done at the store close every day that consumes two hours. Stores are to be closed by the specified time set for the EOD processing else an auto close is done. EOD process consists of a clean up job and reports are generated.

At EOD, a two way Transaction occurs between the AS/400 and the CC 130s at the outlets indicating all the necessary information updation required over the stock, employee etc related information are done.

## Development Environment

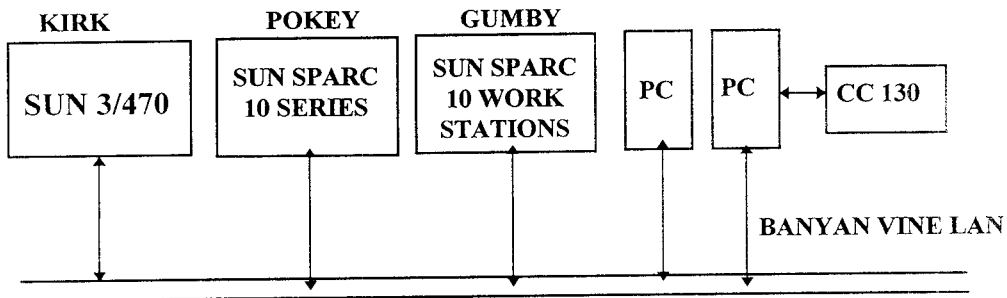
The development environment is summarized below.



1. The System has been developed using Oregon Pascal ( Ver 2.2 ) on VersaDOS a proprietary O.S. exclusively developed for the Client.
2. The PC's offshore will be used as development platforms. The development tools such as, text editor, text search, text compare and documentation tools should reside on these PC 's.
3. The VT/HP emulator on the PC's will be used to connect and communicate to and from the CC130 .

4. The DPS terminal and peripherals (Bar code scanner, MICR{check} reader, Thermal printer, Signature capture pad and Magnetic stripe reader {credit card reader}) will be used as DPS interface.
5. Mascot LAN which has connectivity to the Client's Link will be used at Client's location to connect a new SUN Work Station dedicated for DPS offshore development.
6. The SUN w/s will be used as source control repository and compiling and linking platform. This SUN w/s will not have an opening into the Client's network for security reasons.
7. The SUN w/s will be configured with SCAM (Source Control Applications Module) which is a layer on the UNIX SCCS ( Source Code Control System ).
8. The DPS application modules/configuration files/include files will be available in the SCAM and will be updated every night with the latest files (VIA tape media from other SUN w/s that are used for development work at Circuit City).
9. The Mascot users will be provided with SUN accounts on the SUN dedicated for offshore work.
10. The developers at Mascot will get connected to the dedicated SUN (using a TCP/IP tool for remote login and file transfer), browse out the files required for a particular project, copy the files to their PC, and carry out the required modifications.
11. Once the modifications are complete, the files will be put back to the SUN work area and Compilation and Linking will be done using the compiler and linker embedded into the SCAM.
12. The compiled and linked executable (and other related files) will be transferred back to the PC.
13. The developers will transfer these files to CC130 for unit testing.

14. Compiling, linking and unit testing will continue until unit testing is satisfactory.
15. After unit testing, the developers at Mascot will browse out the latest files from the SUN SCAM, and port the changes from their files to the latest files on SUN.
16. Compiling, linking and unit testing will be done again to insure proper porting.
17. After initial porting the latest files will be sent to Mastech on site personnel (VIA Lotus Notes over Internet) for final porting, integration and system testing,
18. Beta tracking following the software release and hand over to Circuit City support.
19. The overall architecture for the system is based primarily upon the interplatform connectivity made possible by the Banyan Vines network:



- The Sun workstations are implemented as file servers, plus provide compilation services. Object link edits are performed on CC100's set up for that purpose. The development work of editing and unit testing is distributed to programmer PCs, with or without local connection to CC100s or CC130s for unit testing.
- We have implemented this system using both the new Sparc stations (gumby and pokey) and one of the old Sun 3 systems (kirk). All user data and sources resides on gumby and pokey. All Sun SCAM access is through kirk in order to use the old compiler.

- Files on the Sparc are accessed through NFS (Network File System). That means that a drive is mounted on your PC to access the files on the Sparc (Unix file system).

### **3.4.2 LANGUAGE CHOSEN :**

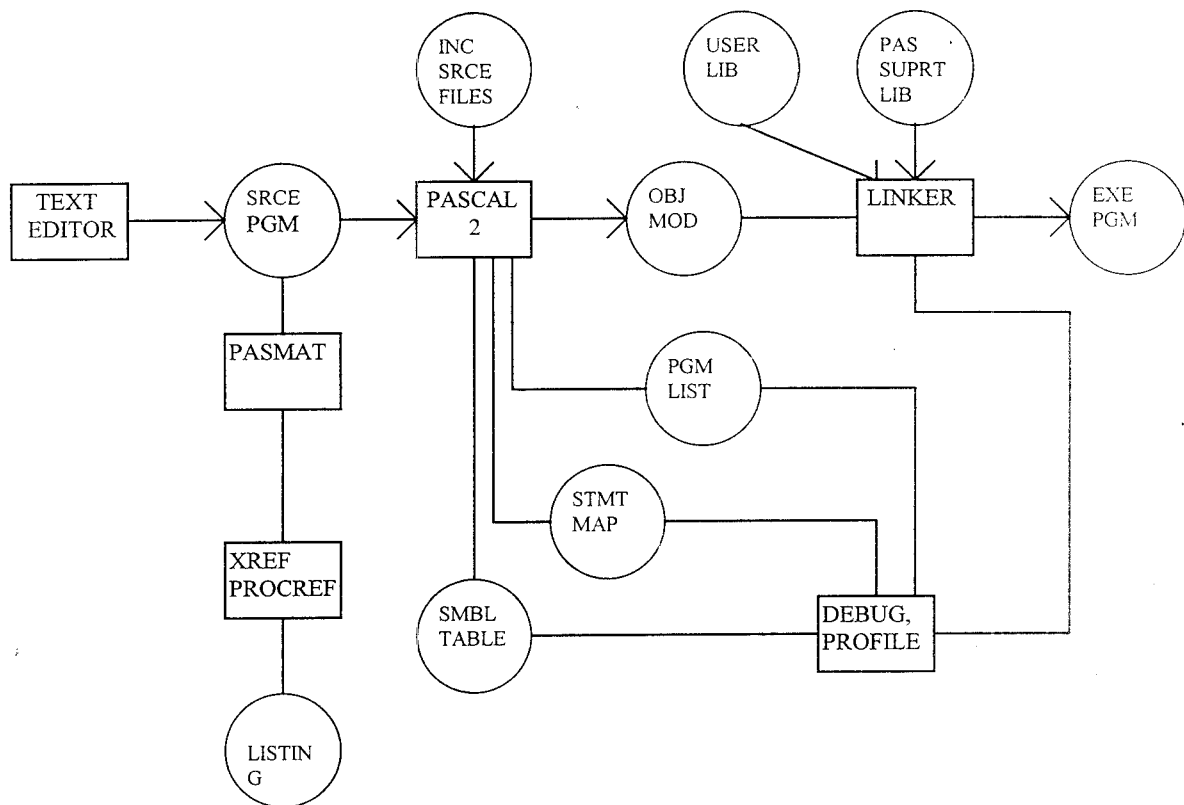
#### **Introduction to oregon pascal**

The Oregon Pascal Software Development System is an integrated system for software development. At the heart of the system is transportable multipass compiler that adheres to the pascal standard while performing optimizations to generate compact, fast code. The Pascal system also offers sophisticated error checking during compilations, extensive error reporting and recovery at run-time, a Debugger to examine the dynamic state of a running program in high-level pascal context, and other development utilities.

Oregon Pascal has many components namely language, compiler, debugger, and utilities. The other optional components such as the Cross-Compiler, Oregon Linker/Assembler, or the Pascal concurrent Programming Package. Together these components offer the professional programmer a structured and unified environment in which to design, code, test, maintain, and improve software. This system allows the speedy production of highly reliable programs.

The Oregon Pascal compiler produces fast, compact, portable programs that are easily transported to other computer systems with a minimum of change, thereby accelerating future software development.

The version used is Pascal-2 system and is designed to run under the VERSAdos operating system on Motorola computers that are based on the MC68000, MC68010, MC68020, and MC68030 processors, optionally utilizing the MC68881 coprocessor.



*The Pascal - 2 Software Development system*

### 3.4.3 DATA BASE SCHEMA STRUCTURE

The schema provides the information necessary for the database to compress and expand application records. A schema must contain one and only one header. The header provides information about the entire record. A record is comprised of one or more segments. All records must contain a segment type X'00'. The maximum number of segment types allowed in a record is 256. Each segment type must contain at least one field definition. The maximum number of field definitions allowed in a segment is 253. The schema is contiguous and in the form:

HEADER  
 SEGMENT 00

FIELD DEF 00

FIELD DEF nn  
SEGMENT 01

SEGMENT nn

*Detailed schema format:*

File header, 12 bytes.

Bytes 0-7 : File ID.

Byte 8 : Key length , attributes.

Bit 7 - Set to indicate this is a V-DOS managed file.

Bit 6 - Set to indicate duplicate keys are permitted. Bit 6 requires bit 7 to be set.

Bits 5-0 - Key length (even,  $0 \leq k-len \leq 60$  bytes).

Byte 9 : Schema version number.

Byte 10-11 : Schema length.

Byte 12-13 : Maximum buffer size (words).

Segment descriptor, 12 bytes. May occur multiple times in a record.

Byte 0 : Segment attributes.

Bits 7-6 : Set to B'10' to indicate this is a segment header.

Bits 5-2 : Unused, should be set to B'0000'

Bit 1 : Set to B'1' if this to be a unique segment .

Byte 1 : Segment ID.

Byte 2 : Number of fields in this segment.

Byte 3 : Expanded segment length (words).

Bytes 4-11 : Segment name string.

Field descriptor, 10 bytes. May occur multiple times in a segment.

Byte 0 : Field attributes.

Bits 7-6 : Set to B'01' to indicate a field descriptor.

Bits 5-0 : Unsigned binary integer indicating the type of this field. Valid field types are shown below.

( 0) Transparent field.

( 1) Transparent Ascii field.

( 2) Boolean field.

( 3) Date field.

( 4) Time field.

(10) Signed binary field.

(11) Unsigned binary field.

(20) Numeric field.

(21) Ascii field.

Byte 1 : Field length (bytes).

Bytes 2-9 : Field name string.

### **Segment x'00' requirement**

Every record type must contain a segment type of X'00. The segment must have the attributes of required and unique and must contain at least one field. For keyed files this segment's first field must be the record key. It may contain additional fields and is otherwise subject to normal segment structure.



### 3.4.4 OPERATING SYSTEM

#### VersaDos session:

This document describes the VersaDos session. CC130, which is the Client proprietary computer, runs the VersaDos operating system. The CC130 console port is connected to your PC COM1 port and the 'Reflection' software is configured to connect to CC130, providing a VersaDos console window. 'Reflection' is the terminal emulator software which should be installed on your PC to allow you an access to CC130.

VersaDos is a multi user operating system and a login is required to get in to the VersaDos OS. The following is the instructions to log into the system.

#### Logging into the system:

1. Check up with you system administrator to locate the PC connected to the CC130 system.
2. Invoke 'Reflection' from the PC to which CC130 (console) is connected. Check up with your system administrator to find out the right 'Reflection' icon to be used.
3. You will see a 'Reflection' window with nothing displayed in the text area of the window. Press **Cntrl-C** (Control key and the 'C' key together on your PC keyboard).
4. You will get a message something like

```

Ä CC130 SCSI: ver A.9.3      9407221200
ENTER USER NO.=
    
```

5. Enter '0' (without quotes) after the '=' sign. You will get a message similar to the following

```

Ä
9/4/96 START SESSION 0005 USER = 0
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!                                     !!!!!
!!!!      Anant at NODE 895             !!!!!
!!!!                                     !!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
=:.PRIV.USER.CF
    
```

```
=OPT ;O
OPTION(S) SET = O
=TIME
9/4/96
=7000.DPS99.SESSID
=ARG 7,21,5,0,7000,ERRLOG.SCREEN.TX

\6:ERRLOG.SCREEN.TX
=ARG 7,21,5,0895,7000,ERRLOG.SCREEN.TX,SYS:,SYS1:,2,00

\6:ERRLOG.SCREEN.TX
\7:SYS:
\8:SYS1:

=END
=USE 7000.DPS99
SYSTEM VOLUME = SYS:
USE DEFAULT VOLUME = SYS:7000.DPS99.
USER NUMBER = 0
USER TASK =
SESSION = 0005
TERMINAL = CN01
OPTION(S) SET = O
=@END
END CHAIN
```

6. The '=' sign is the VersaDos operating system prompt. You are now logged into the system successfully. Your default volume is SYS and default directory (catalog) is 7000.DPS99.

**Logging out of the system:**

After you are done with using the system, you can log out of the system by typing **bye** at the VersaDos prompt. To log into the system again follow the steps in the previous section.

### 3.4.5 OVERVIEW OF SEDIT

Sedit is a screen editor suitable for preparing, on an IBM compatible personal computer running MS-DOS, and displaying, on a DPS terminal as well as on the PC, DPS screens and/or screen mock-ups. It provides some basic editing functions that support data input, modification, repositioning, etc.; it allows the specification of certain "special effects" (reverse video, underline, blink, dim) supported by DPS terminals; and it provides a limited facility for defining Block Mode fields.

Having been developed in-house for a fairly small group of cooperative users, a significant effort has been made to limit the resources required to implement it and to support its use. Efforts to make it more "user friendly" have been expended only in

proportion to its current and anticipated near term use. Thus it has been implemented as a DOS application rather than a WINDOWS application, it does not provide even rudimentary mouse support much less elaborate multimedia presentations, and it is unlikely to ever support auto-formatting wizards or multi-level undo functionality. It is hoped that it will be regarded as a relatively simple utility that is reliable and relatively easy to use -- even when only used on an infrequent basis.

As currently implemented, Sedit supports the following functions:

- Data entry (in overstrike mode) of text with associated video display attributes supported by DPS terminals (reverse video, underline, blink, and dim)
- Field definition and/or modification via a simple field editing window (subject to some field validation rules not presently enforced by the forms compiler)
- Marking of rectangular regions containing text and/or fields (to allow support of operations that apply to regions)
- A variety of operations that apply to marked regions including block moves (drag through or overlay), copy or cut to clipboard, horizontal centering of text and/or fields within a marked region (as independent lines or as a group), framing (reverse video border) a marked region, deletion of all text and fields within a marked region, setting identical field attributes or scan orders for all fields in a marked region, etc.
- Save and load operations that can be used to preserve and restore screen definitions (in a format convenient for Sedit)
- Import and export operations that interpret and/or generate forms files compatible with those used by the forms compiler
- Export operations that generate Rich Text Format (RTF) files that represent screen images with special effects (reverse video as shading, etc.) and that can be imported by word processors such as MS Word 6.0
- Importing and/or exporting of simple ASCII text (provided as a limited facility for transferring data between Sedit and other applications not supported by more complete facilities)
- An alternate operating mode (invoked by the command line option -s) that supports conventions used by the Service team
- Displaying the screen image on a DPS terminal attached to a COM port of the PC

- A limited help facility that provides an on-line reminder of the keystrokes used to invoke various functions
- A DOS gateway

*Systems Design*

---

## **4.1 INPUT DESIGN**

On Order entry, to aid the User ( CSA/SC) to avail information about the specific Brand Model, pressing the Inquiry ( INQ) Key which leads to the Item Inquiry Screen ( Refer Appendix ) revealing all the information as per the screen which the user is entitled to know. Query On Inventory Application can be invoked at this screen. The application can also be accessed from the Inventory Maintenance Screen ( Refer Appendix ). The process of entry further explained in detail below.

### **Entry point for application**

One entry point for the QOI application will be by pressing the function key F2 while in the Item Inquiry screen invoked from Order Entry.

Another entry point for this application will be from a Menu Selection from the Inventory Maintenance Menu.

The Query On Inventory Application would be available to any user if and only if the following conditions are satisfied

- a) Store is open
- b) Order entry is ON or OFF - The application can be accessed from the Inventory Maintenance Screen
- c) DPS state is on-line - Indicating that DPS is not presently carrying EOD Processing
- d) Terminals with or without cash drawers - Sales Counselors are not entitled to carry out cash transactions but are entitled to carry out transaction rather than sales so this facility is to be extended to them and the terminals ,a Sales Counselor works on does not have Cash Drawers.

e) Password is entered correctly .

f) User accessing the Application has the rights to do so.

The Query On Inventory application interacts with the user with the first by putting forward an User Input Screen ( Refer Appendix ) the Screen displaying the options for the user to select as per the options put forward to the user on the Screen.

The Item in each Store is maintained in the four different levels these being as follows

Category

Class

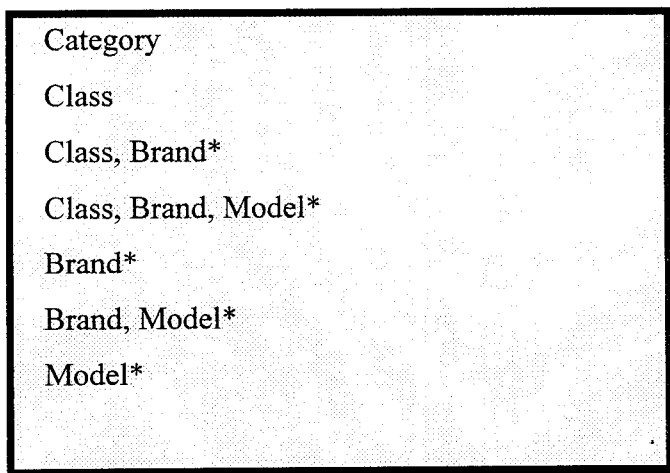
Brand

Model.

### **User input**

A list of allowed field(s) for query is displayed in the User Input Screen.

The allowable fields on which the user may carry out the Query On Inventory are as shown below.



An '\*' next to the field indicates that partial value is allowed for that field. To use the partial value facility, the other fields in the search criteria MUST be input completely. For instance, for the search criteria number 4 (class, brand, model), class and brand values must be input completely to use partial value in the model field.

Category and the description will be displayed on the right side of the screen. User must first select the search criteria by entering a number through 1..7 corresponding to the required search criteria and pressing the <ENTER> key. Now the user will be allowed to enter values in the fields corresponding to the selection number. For instance, if the selection number is 3, then only the CLASS and BRAND fields will be active and the other fields will be inactive.

If the search criteria is category (selection # 1), then a category value has to be entered from the category list displayed on the right hand side of the screen.

Pressing the F1 key will start the query processing. Input fields are validated for the following.

- a) Category value to fall in the categories list displayed on the right hand side.
- b) Class value to fall in the range 1..999
- c) Brand value not to contain "UPC", when search criteria is brand only (i.e. selection 5 )
- d) Brand should be more than one character for the search criteria #4 and #6.

Pressing the F2 key while entering the values for the fields will enable the user to select a different search criteria by entering a different number. Pressing the EXIT key will take the user back to either Inventory maintenance menu or item inquiry screen, depending on from where the application was invoked.

On the selection of the option from 1..7 and on press of < Enter > the input to be Validated and fields as per selection should be activated for the user to enter the Search Criteria values. Later on the input of the value and press of function key F1 or < Enter > key will lead to the execution of the query as per user Criteria. All the inputs have to be validated before the execution.



### **User confirmation**

If the search criteria is other than category and the number of items retrieved is reaches 200, then the User Conformation Screen ( Refer Appendix ) will be displayed. This will allow the user to decide whether the application should continue retrieving more items or stop retrieving and display the items already retrieved.

User can input a 'Y' or 'N' and press the <ENTER> key. Entering 'Y' will continue searching. The confirmation screen will be displayed whenever a multiple of 200 (400, 600..) items are retrieved. Entering 'N' will stop further processing and display the items already retrieved. <EXIT>, <CANC> and <DELETE TKT> keys will be disabled for this screen.

### **Sorting the results:**

After having retrieved the information as per the User Search Criteria the user is given with the options of sorting the final result as per the options provided to the user in the Sort Results Screen ( Refer Appendix ). User will be allowed to sort the displayed results by pressing the F5 key while in final results screen. On the fly sorting can be done on one or more of the following fields

CLASS  
BRAND  
MODEL  
CURRENT PRICE and/or  
ELEKTRA PRICE

User can assign a value in the range 1..5 for the fields or leave it blank if sorting is not required for the field.

For the sample input shown in Sort Results Screen, as shown below

1 CLASS

- 3 BRAND
- 2 MODEL
- CURRENT PRICE
- 4 ELEKTRA PRICE

Sorting for the above input will be done on CLASS (assigned value 1), MODEL(assigned value 2), BRAND (assigned value 3) and ELEKTRA PRICE(assigned value 4). There will be no sorting on CURRENT PRICE as the field is left blank. Note that a lower number indicates a higher level of sorting. Pressing the F1 will sort the result in the ascending order whereas F2 key will sort the result in descending order. EXIT key will take the user back to the Final Results Screen (Refer. Appendix ).

#### **4.2 PROCESS DESIGN :**

The Process of Query On Inventory majorly involves creation of 3(three) ISAM files. The record format of the ISAM files is given in Table 1.

ISAM FILE	FILE NAME	RECORD FORMAT	No. OF FIELDS(KEY LENGTH)
ISAM file #1	BMISAM.DA	STORE NBR, BRAND, MODEL, SKU NBR	4 (24 bytes)
ISAM file #2	CBMISAM.DA	STORE NBR, CLASS, BRAND, MODEL, SKU NBR	5 (27 bytes + 1 filler byte)
ISAM file #3	MBISAM.DA	STORE NBR, MODEL, BRAND, SKU NBR	4 (24 bytes)

**Table 1 - ISAM files for establishing relationship**

All these ISAM files will be created during the End of Day (EOD) processing by reading the records from the item master database (IM.DA).

After the query inputs are entered by the user, one of the three relevant ISAM files, will be accessed to get the brand and model combination(s) for the field(s) selected. This brand and

model value(s) will then be used to access the Item Master database ( IM.DA )to get the inventory information. All the SKUed brand models will be retrieved.

Before the query results are displayed, the information, if any, in the Item Reserve database (IR.DA) will be used for computing the net salable quantity for each item.

The Query On Inventory Application was to be developed based on the following Assumptions/Constraints.

**Assumptions/Constraints**

1. Query will be allowed only on the field or field combinations listed in earlier i.e. Category ,Class , Class and Brand\*, Class- Brand and Model\*, Brand\*, Brand and Model\*, Model\* ( \* indicating that partial values are allowed ).
2. This extended query facility will be available only for the location assigned to the user using this application. No query will be allowed on a location out of the local CC130 using the new query facility.
3. The query will be possible only on the items present in the Item Master Database during the End of Day (EOD) processing. The Items that are added or removed from the Item Master Database ( because of down load processing ) after the ISAM files are updated, will not be available for query through out the day.
4. UPCs will not allowed as a search input for query based on brand field only, as this would retrieve too many records. UPC/ Universal Product Code existing for each and every item equivalent to bar code hence not allowed
5. Scanning of UPCs (for the brand model fields) will not be allowed.
6. There will be a confirmation screen every time the search results in multiple of 200 items. User will have an option to continue further search or stop further processing and display the results.

7. The input screen will have a provision to display up to 17 categories. The program will need modification to accommodate more categories if they exceed 17.

### **Process flow**

This section describes the process flow for the new query application from the user perspective.

- The Entry point for the Query On Inventory application will be by pressing the function key F2 while in Item Inquiry Screen - invoked from the Order Entry ( Refer Appendix ). The Query On Inventory Application can also be accessed Using a Menu selection in the Inventory Maintenance Menu.( Refer Appendix ) by pressing F9. Access to the application via menu selection will be possible under the following circumstances.
  - a) Store is open
  - b) Order entry is ON or OFF
  - c) DPS state is on-line
  - d) Terminals with or without cash drawers
  - e) Password is entered
- An error messages screen Generic Message Screen ( Refer Appendix ) will be displayed , if the application cannot start due to some reason as unavailability of any of the databases or the ISAM files. The user has to press <EXIT> key to end the query.
- Next, an User Input Screen ( Refer Appendix ) for query will appear. This screen displays a list of valid field/field combinations for query. User is expected to select a combination and then enter the values for the field(s). These field values will be used for performing the query. Processing will start once the F1 key is pressed. Pressing the EXIT key will end the new Query On Inventory application and return to the Item Inquiry Screen ( Refer Appendix ) or Inventory Maintenance Menu ( Refer Appendix ) depending on how the application was invoked.

- UPC will NOT be accepted in the brand field when the query is done on brand field only. The SKU numbers for the brand models will be used to extract all the SKUed brand models from the SKU database.
- If the query results in more than 200 items, an User Confirmation Screen ( Refer Appendix ) will be displayed. User will have an option to continue search or display the retrieved results. If the user opts to continue search, the confirmation screen will be displayed every time the number of times cross a multiple of 200 (i.e. 400, 600 ...).
- If the query is done on category the result will be all the classes in that category with their description. This information will be displayed in Category Results Screen ( Refer Appendix ). User can select a particular class from this screen to get more details for each brand model in the class.
- If the query is successful, that is if at least one item is retrieved for the specified search criteria, an output screen Final Results Screen ( Refer Appendix ) is presented to the User. User can browse through the information using function keys [ F1.. F4 ].
- Facility will be provided to sort the result on the fly on CLASS, BRAND, MODEL, CURRENT PRICE and ELEKTRA PRICE fields which can be opted for the user by pressing F5 on the Final Results Screen.
- The Sort Results Screen ( Refer Appendix ) takes in the Validated inputs as per user criteria and sort the results and present the sorted result on the Final Output Screen.

### **4.3 DATABASE DESIGN**

#### **CS DATABASE**

CS means Class. CS database contains the details about the items categorized under class.

## **CSSHEMA**

CSSHEMA gives you the details about the variable name used, length, type and other details about the CS database

```
SCHEMA NAME (CSSHEMA) KEYLEN(4) VERSION(0) BUFSIZE(8000);
SEGMENT ID(00) NAME(CS00) UNIQUE(NO) REQUIRED(YES);
FIELD NAME(CLASS) LEN(4) ATTR(TA) HP(CLASS);
FIELD NAME(CLS_DESC) LEN(26) ATTR(A) HP(CLASS_DESC);
FIELD NAME(Q_PICK) LEN(1) ATTR(A) HP(Q_PICK);
FIELD NAME(CLS_MASK) LEN(1) ATTR(BO) HP(CLASS_MASK);
FIELD NAME(WHS_LOC) LEN(2) ATTR(BS) HP(WHS_LOC);
FIELD NAME(CAT) LEN(2) ATTR(BU) HP(CATEGORY);
FIELD NAME(MIX) LEN(2) ATTR(BU) HP(MIX);
END;
```

## **IM DATABASE**

IM means Item Master. IM database contains information about the each item that is merchandised by the Client

## **IMSCHEMA**

IMSCHEMA gives you the details about the variable name used, length, type and other details about the IM database

```
SCHEMA NAME (IMSCHEMA) KEYLEN(18) VERSION(0) BUFSIZE(8000);
SEGMENT ID(00) NAME(IM00) UNIQUE(YES) REQUIRED(YES);
FIELD NAME(STORE) LEN(2) ATTR(BS) HP(STORE_NBR);
FIELD NAME(BRAND_M) LEN(16) ATTR(A) HP(BRAND_MODEL);

FIELD NAME(CLASS) LEN(3) ATTR(N) HP(CLASS);
FIELD NAME(CAT) LEN(1) ATTR(BU) HP(CATEGORY);
FIELD NAME(PRICE_B) LEN(3) ATTR(BS) HP(PRICE_B);
FIELD NAME(PRICE_A) LEN(3) ATTR(BS) HP(PRICE_A);
FIELD NAME(SPIFF_B) LEN(2) ATTR(BS) HP(SPIFF_B);
FIELD NAME(SPIFF_A) LEN(2) ATTR(BS) HP(SPIFF_A);
FIELD NAME(COST) LEN(3) ATTR(BS) HP(COST);
FIELD NAME(WAR_PRTS) LEN(1) ATTR(BU) HP(WARR_PARTS);
FIELD NAME(WAR_LBR) LEN(1) ATTR(BU) HP(WARR_LABOR);

FIELD NAME(WAR_OTHR) LEN(1) ATTR(BU) HP(WARR_OTHER);
FIELD NAME(QTY_PROG) LEN(3) ATTR(BU) HP(QTY_PROGRAM);
FIELD NAME(QTY_DISP) LEN(2) ATTR(BS) HP(QTY_DISPLAY);
FIELD NAME(TRG_MRGN) LEN(2) ATTR(BU) HP(TARGETED_MARGIN);
FIELD NAME(ITM_MASK) LEN(4) ATTR(BO) HP(ITEM_MASK);
FIELD NAME(ESPPLNCD) LEN(3) ATTR(TA) HP(ESP_PLAN_CD);
```

```
FIELD NAME(ALTINLOC) LEN(2) ATTR(BS) HP(ALT_INV_LOC);
FIELD NAME(PRIORITY) LEN(1) ATTR(BU) HP(PRIORITY);

FIELD NAME(SKU_NBR) LEN(6) ATTR(A ) HP(SKU_NBR);
FIELD NAME(PRCB_P) LEN(3) ATTR(BS) HP(PRICE_P);
FIELD NAME(SPIFF_P) LEN(2) ATTR(BS) HP(SPIFF_P);
FIELD NAME(PR_END) LEN(2) ATTR(DA) HP(PR_END);
FIELD NAME(PR_EVNBR) LEN(7) ATTR(A ) HP(PR_EVENT_NBR);
FIELD NAME(WHSE_LOC) LEN(8) ATTR(A ) HP(WHSE_LOC);
FIELD NAME(TIER1) LEN(1) ATTR(TA) HP(TIER1);
FIELD NAME(TIER3) LEN(1) ATTR(TA) HP(TIER3);
FIELD NAME(NEW_COST) LEN(3) ATTR(BS) HP(NEW_COST);

FIELD NAME(TR_ACT) LEN(2) ATTR(BS) HP(TRANS_ACTIVE);
FIELD NAME(ITM_MSK2) LEN(4) ATTR(BO) HP(ITEM_MASK2);
END;
```

## **IR DATABASE**

IR means Item Reserve. IR database contains information about the item that are kept as reserve.

### **IRSCHEMA**

IRCHEMA gives you the details about the variable name used, length, type and other details about the IR database

```
SCHEMA NAME (IRSCHEMA) KEYLEN(18) VERSION(0) BUFSIZE(8000);
SEGMENT ID(00) NAME(IR00) UNIQUE(YES) REQUIRED(YES);
FIELD NAME(STORE) LEN(2) ATTR(BS) HP(STORE_NBR);
FIELD NAME(BRAND_M) LEN(16) ATTR(A ) HP(BRAND_MODEL);

FIELD NAME(CLASS) LEN(3) ATTR(N ) HP(CLASS);
FIELD NAME(CAT) LEN(1) ATTR(BU) HP(CATEGORY);
FIELD NAME(PRICE_B) LEN(3) ATTR(BS) HP(PRICE_B);
FIELD NAME(PRICE_A) LEN(3) ATTR(BS) HP(PRICE_A);
FIELD NAME(SPIFF_B) LEN(2) ATTR(BS) HP(SPIFF_B);
FIELD NAME(SPIFF_A) LEN(2) ATTR(BS) HP(SPIFF_A);
FIELD NAME(COST) LEN(3) ATTR(BS) HP(COST);
FIELD NAME(WAR_PRTS) LEN(1) ATTR(BU) HP(WARR_PARTS);
FIELD NAME(WAR_LBR) LEN(1) ATTR(BU) HP(WARR_LABOR);

FIELD NAME(WAR_OTHR) LEN(1) ATTR(BU) HP(WARR_OTHER);
FIELD NAME(QTY_PROG) LEN(3) ATTR(BU) HP(QTY_PROGRAM);
FIELD NAME(QTY_DISP) LEN(2) ATTR(BS) HP(QTY_DISPLAY);
FIELD NAME(TRG_MRGN) LEN(2) ATTR(BU ) HP(TARGETED_MARGIN);
FIELD NAME(ITM_MASK) LEN(4) ATTR(BO) HP(ITEM_MASK);
FIELD NAME(LSPPLNCD) LEN(3) ATTR(TA) HP(ESP_PLAN_CD);
```

```
FIELD NAME(ALTINLOC) LEN(2) ATTR(BS) HP(ALT_INV_LOC);
FIELD NAME(PRIORITY) LEN(1) ATTR(BU) HP(PRIORITY);
```

```
FIELD NAME(SKU_NBR) LEN(6) ATTR(A ) HP(SKU_NBR);
FIELD NAME(PRICE_P) LEN(3) ATTR(BS) HP(PRICE_P);
FIELD NAME(SPIFF_P) LEN(2) ATTR(BS) HP(SPIFF_P);
FIELD NAME(PR_END) LEN(2) ATTR(DA) HP(PR_END);
FIELD NAME(PR_EVNBR) LEN(7) ATTR(A ) HP(PR_EVENT_NBR);
FIELD NAME(WHSE_LOC) LEN(8) ATTR(A ) HP(WHSE_LOC);
FIELD NAME(TIER1) LEN(1) ATTR(TA) HP(TIER1);
FIELD NAME(TIER3) LEN(1) ATTR(TA) HP(TIER3);
FIELD NAME(NU_COST) LEN(3) ATTR(BS) HP(NEW_COST);
```

```
FIELD NAME(TR_ACT) LEN(2) ATTR(BS) HP(TRANS_ACTIVE);
FIELD NAME(ITM_MSK2) LEN(4) ATTR(BO) HP(ITEM_MASK2);
END;
```

## **SF DATABASE**

SF means SKU items. SF database contains information about the items which are SKU'ed.

### **SFSCHEMA**

SFSCHEMA gives you the details about the variable name used, length, type and other details about the SF database

```
SCHEMA NAME (SFSCHEMA) KEYLEN(8) VERSION(0) BUFSIZE(8000);
SEGMENT ID(00) NAME(SF00) UNIQUE(YES) REQUIRED(YES);
FIELD NAME(SKU_KEY) LEN(8) ATTR(T) HP(SKU_KEY);
END;
```

## **SM DATABASE**

SM means Store Master. SM database gives information about the Stores that are handled by the Client.

### **SMSHEMA**

SMSHEMA gives you the details about the variable name used, length, type and other details about the SM database

```
SCHEMA NAME (SMSHEMA) KEYLEN(2) VERSION(0) BUFSIZE(8000);
```



```
SEGMENT ID(00) NAME(SM00) UNIQUE(YES) REQUIRED(YES);  
  
FIELD NAME(STR_NBR) LEN(2) ATTR(BS) HP(STORE_NUMBER);  
FIELD NAME(LOG_REN) LEN(4) ATTR(BU) HP(LOG_RRN);  
FIELD NAME(LST_UPDT) LEN(2) ATTR(DA) HP(LAST_UPDATE);  
FIELD NAME(SRT_TYPE) LEN(2) ATTR(A) HP(STORE_TYPE);  
FIELD NAME(LOC_STAT) LEN(2) ATTR(BO) HP(LOC_STATUS);  
END;
```

## **4.4 OUTPUT DESIGN**

### **Generic messages screen**

The Query On Inventory Application requires accessing of Databases namely,

IM.DA - Item Master which holds information of items with Brand/Model ( Combination )  
as it's Primary Key

CS.DA - Class Database which holds the information of items as per Class Division

SK.DA - SKUed Database holding information of the Skued Items

IR.DA - Item Reserve Database holding information of the reserved items

Apart from these 3 ISAM created during End Of Day Processing namely

BMISAM.DA

CBISAM.DA

MBISAM.DA

The Application will thereby run if and only if these are available for it's processing.

Error/information messages that are not related to other screens, such as unable to open data(ISAM) files or database files prior to accepting user inputs. User will have to press the <EXIT> key to end the search and return to the calling application.

### **Category results screen**

The result of the query on category ,as selected by the User in the User Input Screen Selection Number 1 ( Refer Appendix ) , is presented to the User in the screen format as in the Category Results Screen ( Refer Appendix ). All the classes in the category will be retrieved. Class number and the class description will be displayed. 34 classes will appear on one display page in two rows.

If the query results in more then one page of display, the user may browse through the information using the following function keys.

- F1 (NXTPG) to display the next page, if it exists. If the current page is the last page and the user presses the F1 key the message **'YOU ARE CURRENTLY VIEWING THE LAST PAGE'** will be displayed.
- F2 (PREVPG) to display the previous page, if it exists. If the current page is the first page and the user presses the F2 key the message **'YOU ARE CURRENTLY VIEWING THE FIRST PAGE'** will be displayed.
- F3 (FRSTPG) to get to the first page. If the current page is the first page and the user presses the F3 key the message **'YOU ARE CURRENTLY VIEWING THE FIRST PAGE'** will be displayed.
- F4 (LASTPG) to go to the last page. If the current page is the last page and the user presses the F4 key the message **'YOU ARE CURRENTLY VIEWING THE LAST PAGE'** will be displayed.

The details of the brand models for a class can be obtained by entering the class number and pressing the F5 key. Only the Class Numbers in the Current Screen will be accepted as a valid input. Pressing the EXIT key will take the User back to the User Input screen . The Class Number should and will range from 001 to 999.

**Final results screen**

Upon selection of a valid Class in the CATEGORY RESULTS SCREEN ( Refer Appendix ) or for Search Criteria 2..7 from the USER INPUT SCREEN for query the final results will be displayed in the FINAL RESULTS SCREEN ( Refer Appendix ) .

All fields in the FINAL RESULTS SCREEN will be display only fields from the user perspective. However for the programming requirements, there will be a single Character dummy input field. One page of display can accommodate 17 brand models only.

For Category as the Search Criteria by the User, the output will be sorted by Brand, Model and Current Price. For other selections by the User, the output will be sorted by the field(s) on which the Search is being done and the Current price. For instance if Class is used as the Search Criteria, then the result will be sorted by Class and Current Price, if Class and Brand is the Search Criteria, then the output will be sorted by Class, Brand and Current price.

A “Y” will be displayed in the OOP (Out of program) field if the item is out of program, it will be left blank otherwise. The STOCK field will display the incentive compensation (i.e. spiff + commission) and the gross margin, the same way as in order entry applications. The ELEKTRA PRICE field will display the price of the item if it is sold as an Elektra item.

The quantity fields indicate the following:

NS	-	Quantity - Net salable
OB	-	Quantity - open box
DP	-	Quantity - display
XIT	-	Quantity - in transit

The location is displayed at the top of the screen. The PAGE field on the bottom border line will indicate the current page and the total number of display pages in the format ‘n of m’.

The TOTAL NUMBER OF Brand-Models field on the bottom border line displays the count of the brand models as a result of the query.

If the query results in more than one page of display, the user may browse through the information using the following function keys.

- F1 (NXTPG) to display the next page, if it exists. If the current page is the last page and the user presses the F1 key the message **'YOU ARE CURRENTLY VIEWING THE LAST PAGE'** will be displayed.
- F2 (PREVPG) to display the previous page, if it exists. If the current page is the first page and the user presses the F2 key the message **'YOU ARE CURRENTLY VIEWING THE FIRST PAGE'** will be displayed.
- F3 (FRSTPG) to get to the first page. If the current page is the first page and the user presses the F3 key the message **'YOU ARE CURRENTLY VIEWING THE FIRST PAGE'** will be displayed.
- F4 (LASTPG) to go to the last page. If the current page is the last page and the user presses the F4 key the message **'YOU ARE CURRENTLY VIEWING THE LAST PAGE'** will be displayed.

There will be a facility to sort the displayed results by pressing the F5 key. Press of F5 leads to the SORT RESULTS SCREEN ( Refer Appendix ) Details of the sorting procedure are discussed in Input Screen design. If the Search Criteria was other than category, pressing the EXIT key will return control to the USER INPUT SCREEN( Refer Appendix ) . If the Search Criteria was category, then pressing the EXIT key will take the user back to category results screen .

## 4.5 SYSTEM DEVELOPMENT

The development of this design requires the following new modules.

SOINMAIN.PA	- Main module
SOININP.PA	- Module to handle input for query
SOINREAD.PA	- Module to read the ISAM and the database files.
SOINCAT.PA	- Module to handle categories results
SOINPROC.PA	- Module to process the query and generate final results
SOINOUT.PA	- Module to handle the query results.
SOINSORT.PA	- Module to handle sorting of query results.

- Declare all the variable/constants/types in the include files except declarations/definitions that are local to a routine.
- SOINMAIN.PA will be linked with OE (order entry) and RA (Release acknowledgment) applications, the minimum set of include files, will be the ones that are existing in ORINVINQ.PA module (this is a common module for both OE and RA). Include all the files in all the modules (whether used or not) in the same order and place the new include files at the end of existing ones.
- This application uses a generic messages screen (#742) for some messages. The messages should be displayed in the generic messages screen only if it is explicitly specified, all other messages should be displayed in the current screen.

### **SOINMAIN.PA**

This will be the main module for the executable file SOINMAIN.LO. This module will have just two calls to

- (i) TMTinit to initialize the dynamic task table
- (ii) Search\_On\_Inventory routine in SOININP.PA.

All other routines/declarations required for the application will be included in other modules, as they will also be linked with OE and RA applications.

**Algorithm**

1. Call TMTinit(10000,TRUE) to create a dynamic task table entry. This will give access to the TM\_Menu\_Recd that gives information about the location number and port number from where this application is being accessed. Whenever required, the location number can be obtained from TM\_Menu\_Recd.Emp\_Ptr^.assigned\_loc.
2. Call Search\_On\_Inventory routine in module SOININP.PA.

**SOININP.PA**

\*\*\*\*\*

File: SOININP.PA

Purpose : This module accepts inputs from the user through the user input screen and validates them. This information is then passed to SOINREAD .

\*\*\*\*\*

**DESCRIPTION.**

The main routine in this module will be named Search\_On\_Inventory and will be called from two different modules

1. from the module ORINVINQ . PA on pressing F2 key
2. from the module SOINMAIN . PA on pressing F9 key in INV MAINT MENU.

This module will fill up the CATEGORY window, display the input screen and validate the user inputs. The new query application will require the ISAM files (BMISAM.DA, CBMISAM.DA, MBISAM.DA in the SYS:7000.DATA catalog) created by the modified ITEMBAT2.PA module.

**Data Structures**

```

TYPE
    Cat_Rec_Type = RECORD
        Desc : Pac10;           {Description}
        Cat  : shortus;         {Category }
    END;
    Cat_Rec_Type_Array = Array [1..Max_Cat] of
Cat_Rec_Type;

```

```
VAR  
    Cat_Data          : Cat_Rec_Type_Array;
```

### Algorithm

1. Open the ISAM files BMISAM.DA, CBMISAM.DA, MBISAM.DA. If there is an open failure on any file do the following
  - a. Log an error message using El\_write (refer to section)
  - b. Display message “Unable to open ISAM files” in the Generic Message Screen (#742).
  - c. After the user presses <EXIT> key, close the open files and return to the calling routine.
2. Close the ISAM files and continue.

3. Open store master database file (SM.DA). If open fails, log an error message using El\_write. Display message “Error in accessing SM Database” in the Generic Messages screen (#742), return to the calling routine after the user presses <EXIT> key.

4. Make the following initialization of variable Cat\_Data:

```
    For i:=1 to Max_Cat do  
    Begin  
        Cat_Data[i].DESC := '          '; {ten blanks}  
        Cat_Data[i].CAT  := 0;  
    End;
```

5. Read (in EXamine mode) the SM database segment 30 for Str\_Nbr 0 (segment 30 of database SM contains four fields: DESC, CAT, SUB\_DEPT and DEPT. Out of these, only the first two i.e., DESC and CAT are required for our purpose) and get the values of the fields DESC and CAT. Fill the Cat\_Data array elements DESC and CAT for all the occurrences. Currently there will be 15 occurrences of segment 30 and consequently Cat\_Data[16] and Cat\_Data[17] will still contain initialized values (see Step #3). These 2 array elements are kept in case some new categories are introduced in the future. If the number of categories exceed Max\_Cat, log an error message using El\_write, display message “Number of Categories exceeds the application , cannot proceed” in the Generic.

Messages screen (# 742), after the user presses <EXIT> key close the SM database and return to the calling routine.

6. Close the SM database.
7. Populate the description and category fields in the category window of the user input screen using the data stored in the array `Cat_Data`.
8. Display the user input screen with the following
  - a: Only the selection number field active and all other fields protected.
  - b: cursor positioned in the selection number field.Note that F1 and F2 keys are irrelevant before a valid search criteria is selected. So if the user presses F1 or F2 key at this point, display message "Please select a number and press <ENTER>". If the user presses <ENTER>, validate the input (valid range 1..7). If invalid, display message "Invalid Selection , Please input a number 1 through 7" and go to step 8.
9. If the user presses <EXIT> or <CANC> key, return control to calling routine. Trap the <DELETE TKT> (Delete ticket) key and go to step 8.
10. Display the input screen with the following:
  - a. make only the allowed input fields active depending on the search criteria selected in step 8. protect the other fields.
  - b. position the cursor on the first fieldFor instance:

If the user has selected number 4. the search criteria is CLASS, BRAND and MODEL fields. So, the fields for CLASS, BRAND and MODEL will be active. CATEGORY field will be protected. The cursor will be positioned in the CLASS field.
11. If the user presses <EXIT> or <CANC> key, return control to calling routine. Trap the <DELETE TKT> (Delete ticket) key and go to step 10.



12. If the user presses the F1 key go to step 8. If the user presses the F2 key, validate the input values. The following is the list of invalid inputs:

- User inputs a CATEGORY which is not displayed on the input screen
- User inputs a CLASS not in range 1..999.
- User inputs 'UPC' as BRAND for search criteria 5 (i.e. brand field only).
- User inputs a single character for BRAND and the selection number is 4 or 6.
- User does not input in all the necessary fields.

13. If validation fails, use BMERROR and BMLNMODE to display the relevant Error Message "Invalid input , please select from the list displayed" and go to step 10. If validation of input succeeds, display message "Search in progress , please wait".

14. Initialize the variable User\_Search\_Criteria as shown below:

```
With User_Search_Criteria Do
Begin
  Selection_Number := 0;
  Category :=0;
  Class := '   ';           {3 blanks }
  Brand := '   ';          {3 blanks }
  Model := '           ';  {13 blanks}
End;
```

15. Assign the data from user input screen (i.e., selection number and Category, Brand, Model etc. depending upon the selection number) to the respective fields of the variable User\_Search\_Criteria.

16. Call Procedure Process\_Search in the module SOINREAD.PA with User\_Search\_Criteria as the parameter. Go to Step 8.

**SOINREAD.PA**

\*\*\*\*\*



FALSE for the second parameter. Message "SEARCH IN PROGRESS , PLEASE WAIT" should remain displayed.

2. If the function `Process_Category` returns a class number (i.e. a value greater than -1), then continue as if search criteria is 2 (i.e. Class only) and the return value as the class for further processing. If the function returns -1, return to calling routine.

3. If the selection is 2..7, determine the relevant ISAM file to be used. Use the following pseudo-code :

```
CASE selection of
    2, 3, 4 : File_to_Access := CBMISAM.DA;
    5,6     : File_to_Access := BMISAM.DA;
    7       : File_to_Access := MBISAM.DA;
END;
```

4. Get the store number assigned to the user from the variable `TM_Menu_Recd.Emp_Ptr^.assigned_loc`.

5. Open database file `SF.DA` , log an error message using `EI_write`. Display message "Error in accessing SF database" in the generic messages screen and return to calling routine after the user presses the <EXIT> key.

6. Access the first (next) record from the relevant ISAM file matching the user selection criteria and the store number. If there are no more records matching the user inputs, go to step 11.

7. If the `SKU_Nbr` in the record accessed in step 6 is '000000' (six zeros) or (six blanks), it means there are not SKUed items. Create a new node for `Search_Results` pointer variable, initialize the elements, assign values to `Brand` and `Model` (obtained from step 6) . Insert the new node in the linked list. Keep track of number of nodes in the linked list (variable `No_Of_Items`). If the number reaches a multiple of `Min_Items_Before_Confirm` (i.e. 200) go to step 9, otherwise go to step 6.

8. If SKU\_Nbr in the record accessed in step 6 is anything other than '000000' (six zeros) or ' ' (six blanks), then get all the SKUed Brand/Model for this SKU number from the SF database. (from all the occurrences of segment 20). Create a new node (Search\_Results) for each Brand/Model, initialize the elements, assign values to Brand and Model fields. Insert new node into the linked list. Keep track of number of nodes in the linked list (variable No\_Of\_Items). If the number reaches a multiple of Min\_Items\_Before\_Confirm (i.e. 200, 400, 600..) go to step 9 otherwise go to step 6.
9. Call the routine User\_Confirm. Here, Display the confirmation screen. Below are some details for this step.
  - Clear the messages line (row 24) and display the user confirmation screen. No function keys will be enabled. Cursor will be the on the input field (not that there is only one input field on this screen). Please refer to the line **SEARCH RESULTED IN MORE THAN nnn ITEMS** in this screen Display the value of No\_of\_Items in place of **nnn** in this line.
  - If the user inputs anything other than 'Y' or 'N' and presses <Enter> key, display message "Missing data files , application cannot run until tomorrow".
  - If the user inputs 'Y' and presses <Enter>, the Function should display message "Invalid selection , please input a number 1..7" and return TRUE .
  - If the user inputs 'N' and presses <Enter>, the Function should return FALSE.
10. IF User\_Confirm returns FALSE go to step 11, otherwise repeat steps 6 to 9 until all the matching records are retrieved from ISAM.
11. Close SF . DA database and the ISAM file.
12. If not even a single item is retrieved (i.e. no nodes created for variable Search\_Results), then display message 9 and go to step 14.

13. Call Procedure `Get_Inventory` in module `SOINPROC.PA` with `Search_Results` as the parameter.
14. If the search criteria was 1 (i.e. category) then call `Process_Category` with first parameter as 0 and the second parameter `TRUE`. Go to step 2. For all other search criteria (2..7) return to the calling routine.

### **SOINCAT.PA**

```
{*****  
File: SOINCAT.PA
```

#### **Purpose:**

This file processes the query on category . All the classes & their Corresponding description are retrieved. The retrieved records are then sorted on class.

```
*****  
DESCRIPTION:
```

The main routine of this module will be

Function `Process_Category(Category : Shortus, Display_Only : Boolean)`. This module will retrieve all the classes for the given category and display the category results screen. The classes in the category results screen will be sorted in ascending order.

The first parameter to `Process_Category` routine will be the category to retrieve the classes for(from the `CS.DA` database). The second parameter indicates whether to actually retrieve the data or to use an already retrieved data. The value in the first parameter will be ignored if the second parameter is `TRUE`. The function will return the class number which the user has selected or a -1 (minus one) if no class is selected. It will also return a -1, if there are no classes for that category under consideration.

#### **Algorithm**

1. If the second parameter is `TRUE`, then go to step 7.

2. Open the class database CS.DA. If database-open fails, log an error message using El\_Write and display a message "Error in accessing CS database" in the generic messages screen and return -1 (minus one) to the calling routine after the user presses the <EXIT> key.
3. Read first (next) record of the class database in EXamine mode. Go to step 5 if u reach end of database.
4. Retrieve segment 0. If the database field CAT does not match the category then go to step3. If it matches , keep track of number of matching record & do the following :
  - a. Allocate a new node for Category Results.
  - b. Assign the values for Class and Desc fields.
  - c. Insert the new node into the linked list for Category results to form a list sorted on Class.
5. Close the class database.
6. If the number of matching records (from step 4) is zero display message "There are no Classes in this Category" in the Generic messages screen and return control to the calling routine (with return value -1) after the user presses <EXIT> key.
7. Fill up the total classes and page number (current page and total number of pages), class and its description (using the Category\_Results linked list). Display the category results screen.
8. If the user presses <EXIT> key at any point of time, return control to the calling routine with a return value -1 (minus one).
9. If the user presses F1 key, and if the currently displayed is not the last page, then fill the fields with data for the next page. The linked list and redisplay the screen. If the last page

is currently displayed, display message "You are currently viewing the last page". Update the current page number field on the screen if required.

10. If the user presses F2, and if the currently displayed page is not the first page, then fill the fields with data for the previous page from the linked list and redisplay the screen. If the first page is currently displayed, display message "You are currently viewing the first page". Update the current page number field on the screen if required.

11. If the user presses F3, and if the currently displayed page is not the first page, then fill the fields with data for the first page from the linked list and redisplay the screen. If the first page is currently displayed, display message "You are currently viewing the first page". Update the current page number field if required.

12. If the user presses F4, and if the currently displayed page is not the last page, then fill the fields with data for the last page from the linked list and redisplay the screen. If the last page is currently displayed, display message "You are currently viewing the last page". Update the current page number field if required.

13. If the user presses F5 key, validate the input class value by checking if it is one of the classes displayed in the current page. If yes, then return the class number else display message "Invalid Class , select a Class from the current screen".

## **SOINPROC.PA**

\*\*\*\*\*

File : SOINPROC.PA

Purpose : The purpose of this module is to retrieve the details for each item from item master database. Necessary calculations for current price, stock, elektra price will be carried out. Quantity adjustments for net saleable and display buckets will be done by reading the item reserve database.

\*\*\*\*\*

**DESCRIPTION.**

The main routine in this module will be Procedure Get\_Inventory. This procedure will be called from Procedure Process\_Search in module SOINREAD.PA. This module will get the details (i.e., Class, OOP, Stock, Qty\_NS, Qty\_OB, Qty\_DP, Qty\_Xit, Cur\_Price and Elektra) of all the brand/models in the Serach\_Results\_Rec data structure. Some of these details, such as Class, OOP, Qty\_NS, Qty\_OB, Qty\_DP, Qty\_Xit, are readily available from item master database (IM.DA). Qty\_NS and Qty\_DP have to be adjusted with the corresponding values from item reserve database (IR.DA). The other fields, i.e.. Stock, Cur\_price and Elektra\_Price will be calculated.

\*\*\*\*\*

**Algorithm**

1. Open IM.DA and IR.DA. If open fails, log an error message using EL\_Write. Display message "Error in accessing IM & IR database" in the generic messages screen and return to calling routine after the user presses the <EXIT> key.
  
2. Get the Brand/Model information from the first (next) node of the linked list Search\_Results. At end go to step 10.
  
3. Get the class and quantity by doing the following:
  - (a) Build the IM key using the brand, model and the store number (get the store number assigned to the user from TM\_Menu\_Recd.Emp\_Ptr^.assigned\_loc).
  - (b) Access the IM record(in EXamine mode).



(c) Retrieve Segment 01 to get the Class, Qty\_NS, Qty\_DP, Qty\_OB, and Qty\_Xit.  
Update the corresponding fields in Search\_Results.

4. Get the Cur\_Price by doing the following:

(a) Retrieve Segment 0.

(b) In Segment 0. if database field Price\_P is greater than zero then Cur\_Price is Price\_P. Else Cur\_Price is Price\_A. Update the corresponding field in Search\_Results.

5. Get the OOP, by using the following pseudo code:

If IM\_PROGRAMMED IN ITM\_MSK2 then { in Segment 0 of IM.DA }

OOP:='Y'

Else

OOP:='';

Update the corresponding field in the Search\_Results.

6. Get the stock: To calculate stock, make a local copy of the Function Cal\_Stock from ORINVINQ.PA module. Modify the local function to adapt to the module (such as variable references, parameters, function name etc.). Update the corresponding field in the Search\_Results.

7. Get the Elektra price: To calculate Elektra Price make a local copy of the Procedure Calc\_Processing\_Fee from module THRDEXT2.PA. Update the corresponding field in the Search\_results.

8. Build the IR record using the store number, brand and model. Get the record (in Examine mode) from IR.DA. If the record does not exist go to step 9. If the record exists, then do the following

(a) Retrieve Segment 02 of the record to get the values for the database fields

RSVNTSAL and RSVDISP. Use the following pseudo code to adjust the Qty\_NS and Qty\_DP obtained in step 3c.

$Qty\_NS = Qty\_NS - RSVNTSAL$

$$\text{Qty\_DP} = \text{Qty\_DP} - \text{RSVDISP}$$

(b) Update the corresponding fields in the Search\_results.

9. Go to step 2.
10. Close IM.DA and IR.DA databases.
11. Call Procedure Display\_Results. in module SOINOUT.PA.
12. Return to calling routine.

\*\*\*\*\*

### **SOINOUT.PA**

{\*\*\*\*\*

File: SOINOUT.PA

Purpose:

This module will display the Final results screen of the query application. It displays Brand, Model Class, NS, OB DP,XIT,Current Price,Stock,OOP and the Elektra Price. This also displays Location # and total # of BMs selected as the result of the query.

\*\*\*\*\*}

DESCRIPTION.

The purpose of this module is to display the search results, handle the function keys for the final results screen. The main routine in this module will be Procedure Display\_Results.

### **Algorithm**

1. Call the function Sort\_Result(FALSE) in module SOINSORT.PA. The parameter value (FALSE) tells the function to sort the results based on the initial search criteria selected by the user.
2. If the Sort\_Result function returns FALSE, then go to step 4.

3. Populate the form record for the final results screen with the data from the variable Search\_Results. Update the Page number (current and total) and total brand/models fields in the form record.
4. Display the final results screen. Handle the function keys F1 to F4, redisplay the results screen with appropriate date. Display the appropriate messages if required.
5. If the user presses the <EXIT> key at any point of time, return to the calling routine.
6. If the user presses the F5 key (Sort) then call function Sort\_Result(TRUE). The parameter value (TRUE) tells the function to get the user sort criteria and sort the results.
7. If the Sort\_Results function returns TRUE, then go to step 3. Otherwise go to step 4.

**SOINSORT.PA**

\*\*\*\*\*  
FILE : SOINSORT.PA

**PURPOSE :**

This module sorts a linked list, Search\_Results, based on the inputs provided by the user using Form 741. In the absence of such inputs, User Search Criteria provides this information.

\*\*\*\*\*  
**DESCRIPTION.**

This module is to sort the data in Search\_Results linked list. The main routine in this module will be Function Sort\_Result(Get\_User\_Input : Boolean) . If the parameter is TRUE, the function will display the sort input screen to get the sorting input from the user. If the parameter is FALSE, it will sort the data as per the original search criteria from the user. If the sorting was actually done the function will return TRUE. If the sorting was not done because of :

- a. Sort order requested was same as the previous sort order.
- b. The user did not want the sorting and pressed <EXIT> key from the sort input screen.

**Algorithm**

1. If the parameter value is FALSE then initialize Prev\_Sort\_Order and Cur\_Sort\_Order to blanks (spaces).go to step 4.
2. Display the sort input screen. Display the relevant error messages, if required. If <EXIT> key is pressed at any time return FALSE to the calling routine
3. Validate the inputs as per Section. Update the Cur\_Sort\_Order variable in the following format:  

```
Cur_Sort_Order = '      '
```

If the F1 key (sort ascending) is pressed  
Cur\_Sort\_Order[1] = 'A'

If the F2 key (sort ascending) is pressed  
Cur\_Sort\_Order[1] = 'D'

If the sort order for class/brand/model/current\_price/elektra\_price is in the range 1..5 (say 'n'), then update the n+1th element in Cur\_Sort\_Order with the corresponding value of the constants as explained below

```
Sort_On_Class for class  
Sort_On_Brand for brand  
Sort_On_Model for model  
Sort_On_CurPrice for cur_price  
Sort_On_ElektraPrice for elektra_price
```
4. Display the message “Sorting is in progress , please wait” using BMLnmode and BMSendio.
5. Compare the values in Prev\_Sort\_Order and Cur\_Sort\_Order.If they are same then there is no need for sorting. Sorting is also not required if the Cur\_Sort\_Order is a subset of Prev Sort Order (ex. Cur\_Sort\_Order = 'ACBM' and Prev\_Sort\_Order = 'ACBMPE'). Return FALSE to the calling routine.
6. If they are different, compare the values in each position starting from 1. if few positions match, then sorting is required only for the rest of the fields.

For instance,

- a) If Prev\_Sort\_Order contains ACBMP (Ascending, sorted on Class, brand, model and current price) and the Cur\_Sort\_Order contains ACBME (Ascending, sort on Class, brand, model and elektra price), then we need to sort only on the Elektra price as the existing list is already sorted on class, brand and model.
- b) If Prev\_Sort\_Order contains ACBMP (Ascending, sorted on Class, brand, model and current price) and the Cur\_Sort\_Order contains ACMBP (Ascending, sort on Class, model, brand and current price), then we need to sort only on the model, brand and current price as the existing list is already sorted on class.

After sorting , assign the value of Cur\_Sort\_Order to Prev\_Sort\_Order and return TRUE to the calling routine.



*Implementation*

---

## **5. IMPLEMENTATION**

The purpose of the module is to serve an extended query upon Inventory based on partial values and all possible combinations.

This module is invoked by issuing a call to SOINMAIN program. This program in turn calls SOININP for getting the input from the user. SOININP on getting the input from the user calls SOINREAD or SOINCAT depending upon the option chosen. After the process of SOINREAD or SOINCAT a doubly linked list will be created which would contain all the details regarding the filtered data. Now, SOINPROC is called which populates the fields in the doubly linked list.

After populating all the fields in doubly linked list SOINOUT program is called. This is the interface part of the module with the user after getting the matching records. This calls SOINSORT whenever required to sort the data depending upon the fields and their order to be sorted.

These modules were uploaded to the Sun Workstation located at the Client's place. It was then compiled and linked in the Sun workstation and the executable files of these modules were downloaded into the PC's at development centre here in Bangalore. Then, the executable files were transferred to the CC130 machines and extensively tested with the sample data and cleared all the tests.

*Conclusion*

---



## **6. CONCLUSION**

The module Query on Inventory fulfills the requirements from the field for a flexible query on inventory which is based on Class, Category, Brand/Model and all partial combinations.

The module is invoked by (1) pressing F2 from existing query on inventory and (2) Menu selection from the Inventory Maintenance menu. This module can be accessed by any one with an Order Entry license under the following circumstances :

- Store is Open
- Order Entry is ON/OFF
- DPS State is On-Line
- Terminals with/without cash drawers
- Password is entered correctly.

This module makes query on inventory more flexible and user-friendly than the one that could be done by pressing the INQ key during an Order Entry transaction.

*Appendix A*

---

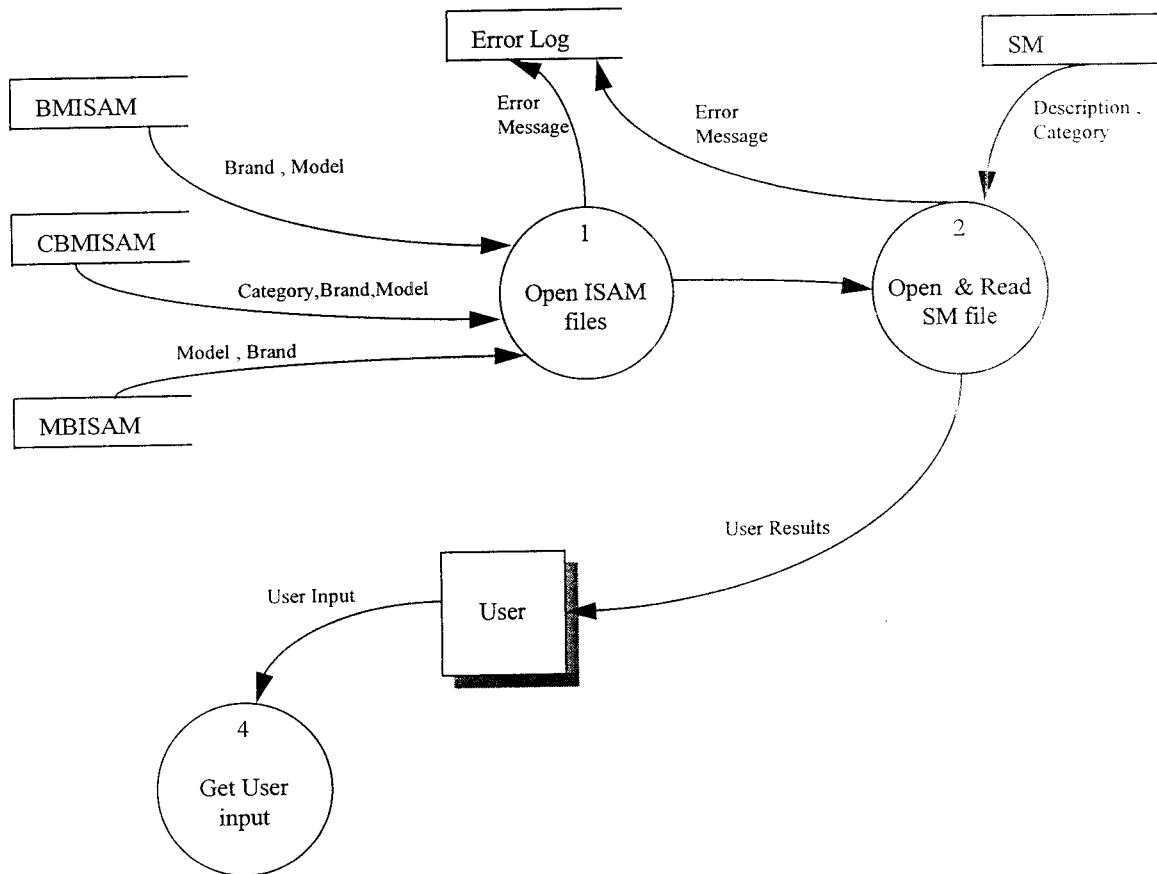


Fig 1.1 : Data Flow Diagram for Soininp Module.

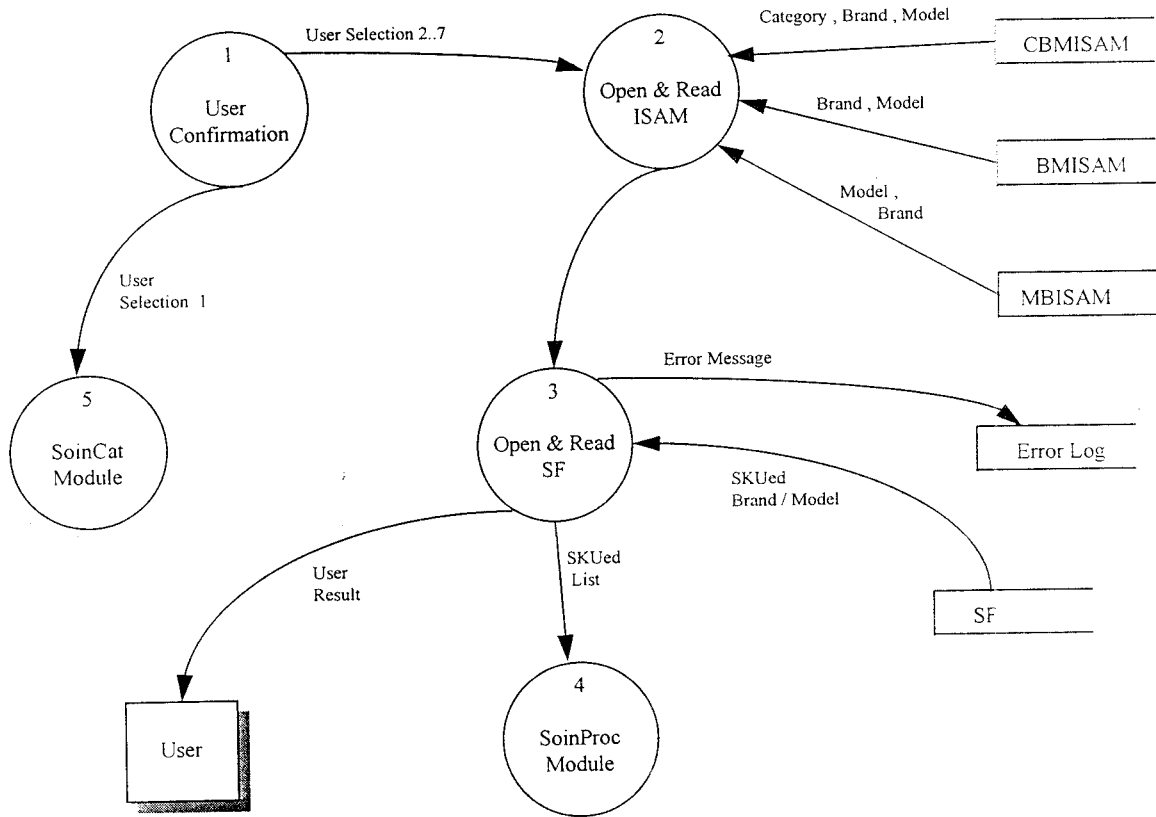


Fig 1.2 : Data Flow Diagram for Soinread Module.

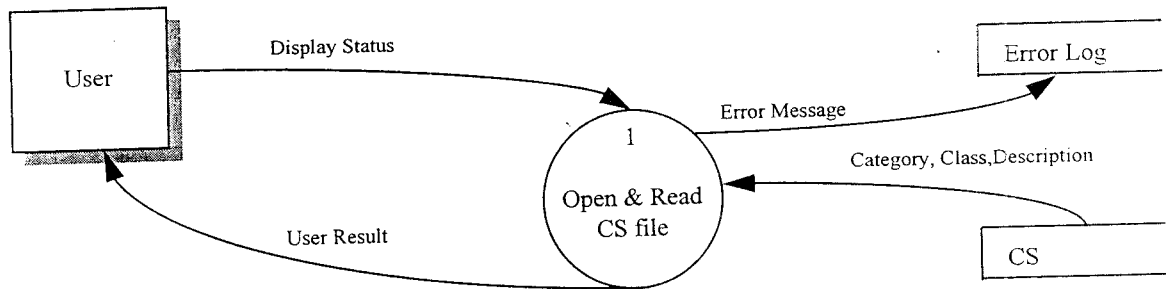


Fig 1.3 : Data Flow Diagram for Soincat Module.

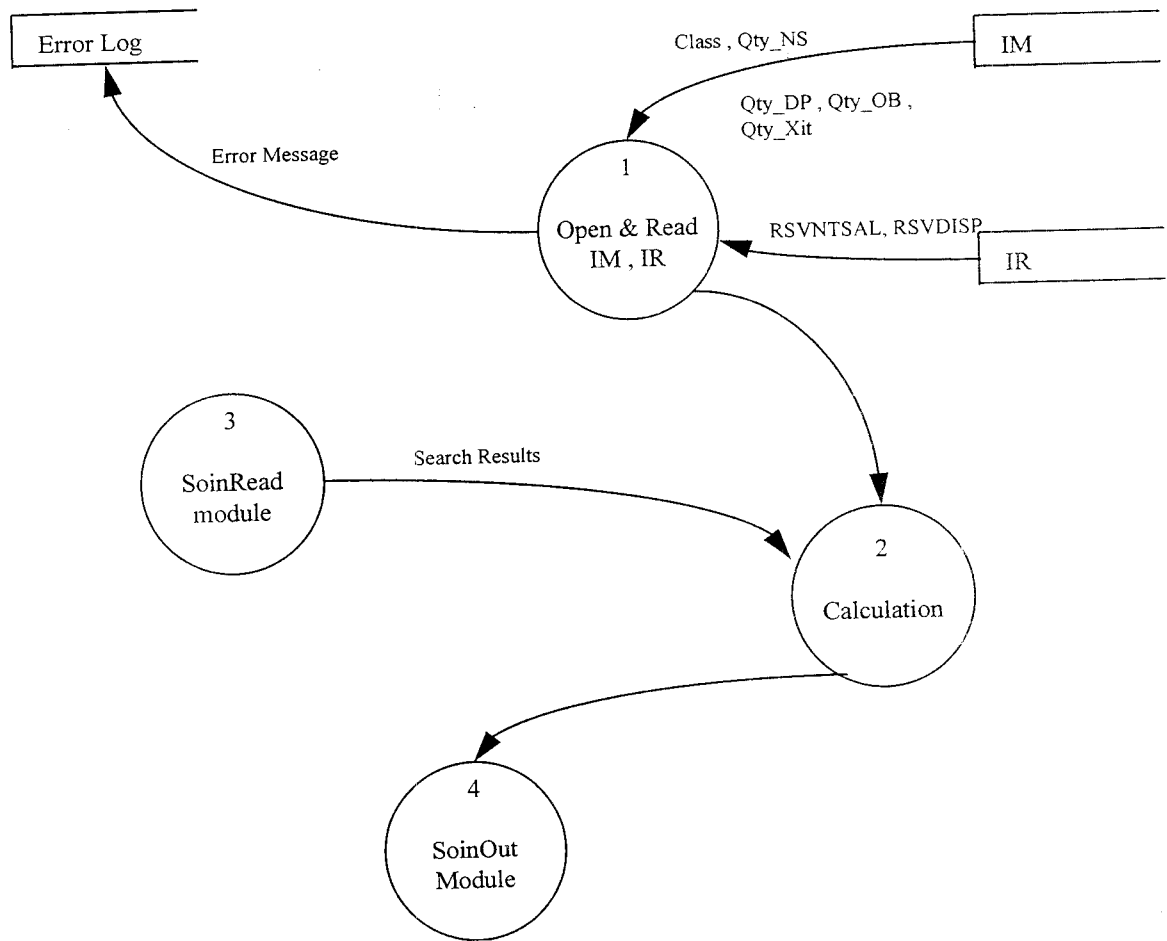
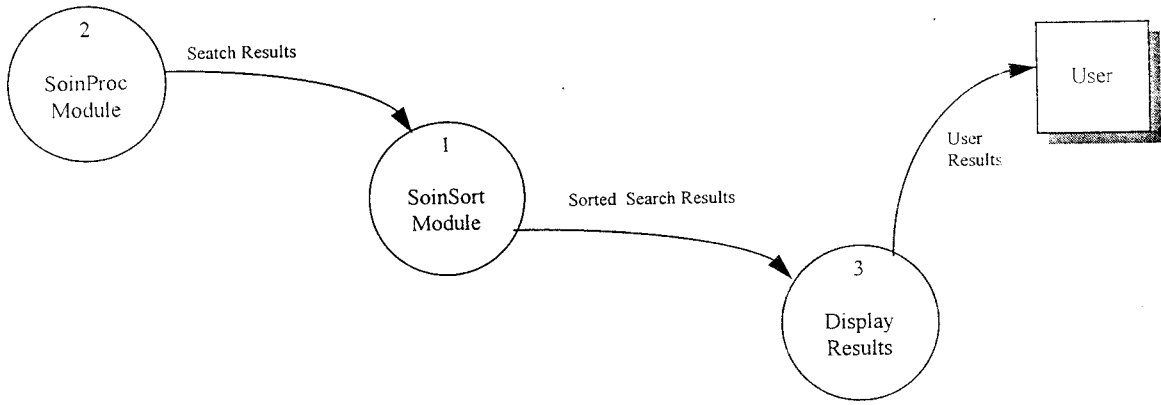
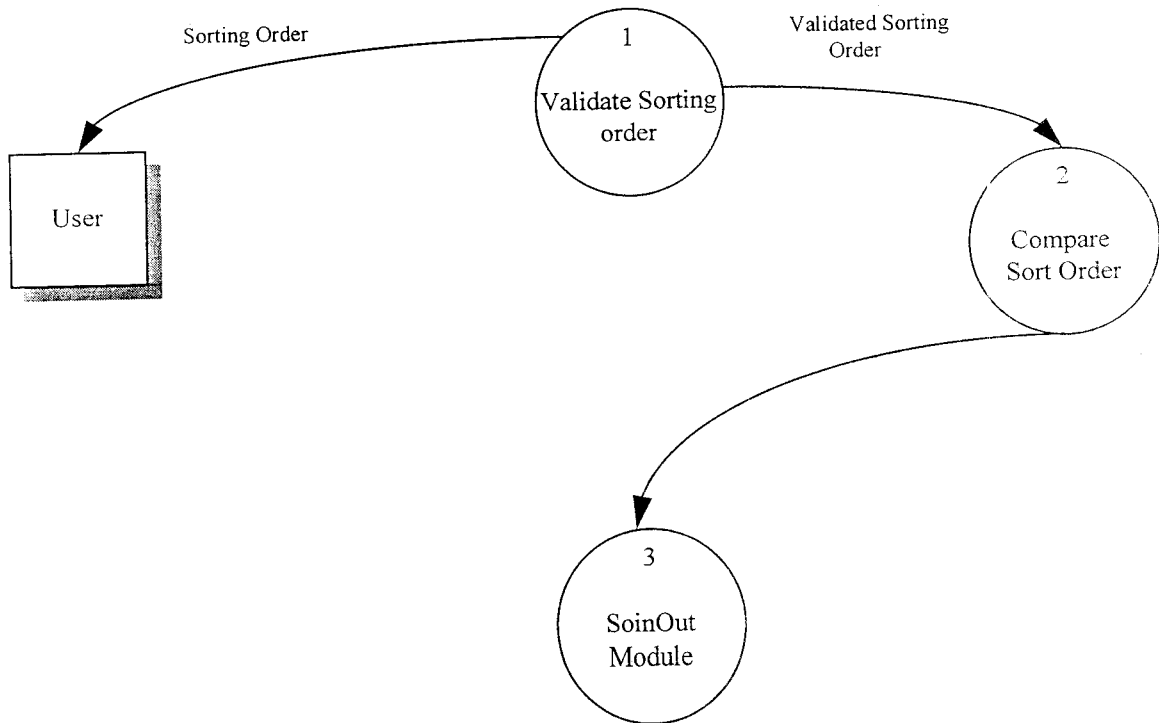


Fig 1.4 : Data Flow Diagram for Soinproc Module.



*Fig 1.5 : Data Flow Diagram for Soinout Module.*



*Fig 1.6 : Data Flow Diagram for SoinSort Module.*

*Appendix B*

---

APPLICATION ENTRY POINT :

ENTRY POINT 1 :

```
Item Inquiry
Brand/Model: ..... Loc: ..... Price: ..... MaxLvl : .....
Pr: .. Stock: ..... Warranty: P>... L>... O>... H/S> .
.....
Saleable          InStore: .....      InProcess: .....
Closed : .....   InTrans: .....      Deposit: .....    Delivry: .....
OpenBox: .....   SpecOrd: .....      Layaway: .....    PickUp : .....
Display: .....   Defect : .....      RainChk: .....
F2>SRCH
```

This screen is same as the existing item inquiry screen but provided with an extra option to invoke the EXTENDED SEARCH FACILITY(Detailed item inquiry screen). By pressing F2 - Function key the detailed inquiry screen is displayed and user can proceed with his further queries.



ENTRY POINT 2 :

```
=====
INVENTORY.MENU.305                INV MAINT MENU
=====
                                ITEM
(F1) 82. Item Info Change
(F2) 83. Quantity Change
(F3) 84. Create Display Tags
(F4) 85. RTV Manual

                                TAG
(F5) 86. Tag Maintenance
(F6) 87. List Tag File

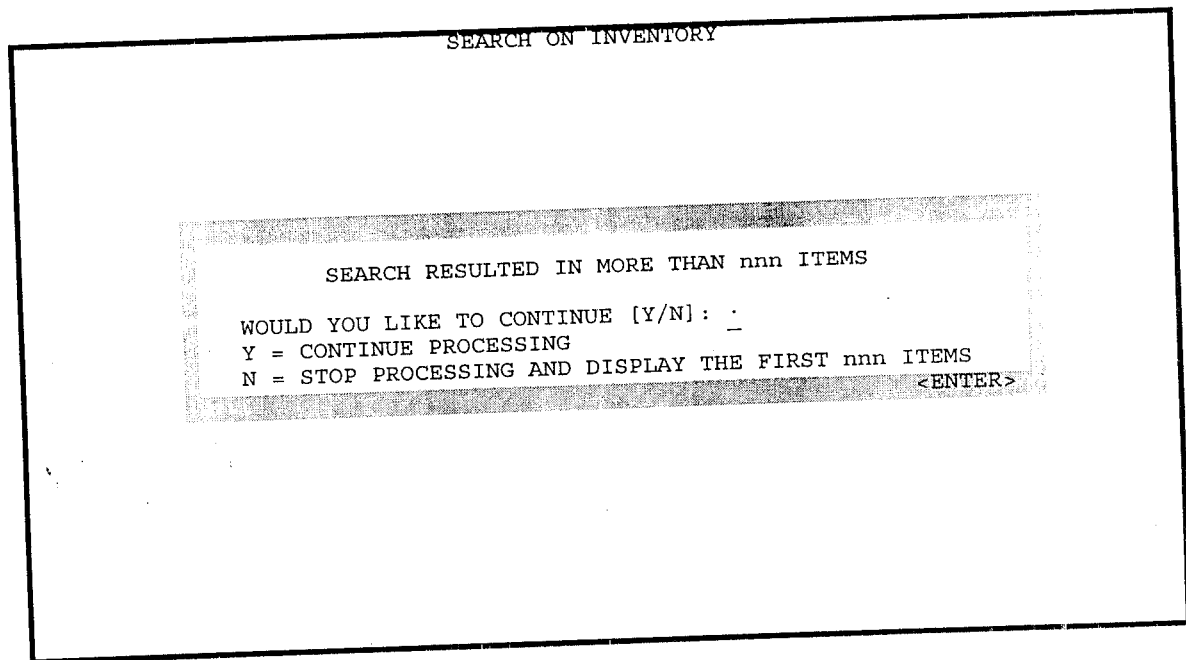
                                RETURNS
(F7) 188. Return of Sale
(F8) 189. Return of Sale History

                                SEARCH
(F9) 295. Search On Inventory BRD-MODEL / TAG #
=====
```

There is another alternative method for invoking the extended search facility. This can be achieved by just pressing a F9 - Function key from the inventory maintenance menu.



USER CONFIRMATION SCREEN :



This is the user confirmation screen displayed when the searching based on any selection criteria exceeds above 200 items. This is just to confirm with the user whether he needs further more search or not.

If the user presses 'Y' then the search procedure will continue until next matching set of 200 items were found .

If the user presses 'N' then it will stop searching and displays the searched items in the screen in the predefined format. The user can scroll up and down the screen and locate the exact item he aimed for and know it's details



SORT INPUT SCREEN :

SEARCH ON INVENTORY

SELECT THE FIELDS AND ORDER FOR SORTING

BLANK=NO SORT 1=FIRST LEVEL 2=SECOND LEVEL...

1	CLASS
3	BRAND
2	MODEL
7	CURRENT PRICE
4	ELEKTRA PRICE
-	

SORT ORDER: PRESS THE F1 KEY FOR ASCENDING (I.E. SMALL TO LARGE,A..Z,0..9)  
PRESS THE F2 KEY FOR DESCENDING (I.E. LARGE TO SMALL,Z..A,9..0)

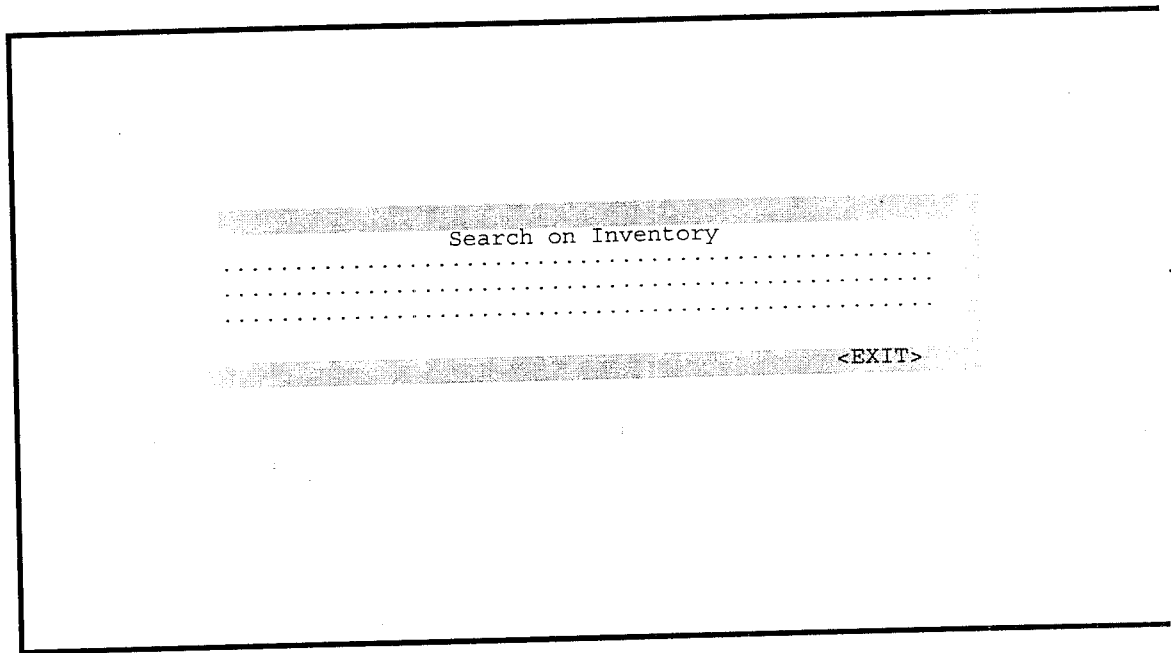
F1>SORT ASCENDING F2>SORT DESCENDING

For any Output screen displayed , if the user wants the screen to be in the sorted form based on any field's on the screen he can just press F5 function key for sorting.

When he presses F5 key for sorting the above screen will be displayed. In the sort input screen the user can select the order of sort based on any field and he also can choose whether the selected sort order should be ascending or descending one.



GENERIC MESSAGE SCREEN :



This is the generic messages screen used by the application to display few error/information messages that are not related to other screens, such as unable to open data(ISAM) files or database files prior to accepting user inputs. User will have to press the <EXIT> key to end the search and return to the calling application.

*Glossary*

---



## GLOSSARY

### A-PRICE

The default price for a brand model. A price is in effect when the brand-model is not affected by a promotion.

### ACCESS KEY

A private, two character, shorthand user identification, used prior to each transaction. The access key allows multiple persons to use a single terminal while providing a minimal level of security.

### ACCOMADATION SALE

A sale made to an employee at cost.

### ACTIVATE

A DPS function that moves a remotely created transfer to the shipping location.

### ALTERNATE SALE

A sale where the selling and inventory locations differ. Alternate sales require DPS to create and manage reservation of merchandise using the network.

### AUTHORIZATION (OF PAYMENT)

The process of verifying and guaranteeing the availability of credit or funds to cover the purchase. An authorization code is recorded in the payment detail.

### B-PRICE

The Normal minimum sales price for an item.

### BACKUP

A copy of a node's database. DPS uses backups to both tape and disk.

### BATCH

A collection of jobs that run after the store closes. Batch performs file clean-ups and produces reports.

### BLOCK MODE

A style of interaction between terminal user and program that issues sets of prompts at a time, some with default data, organised as a form on a terminal screen. The user employs the cursor movement keys to select prompts and then enter or change data in the various fields. The user is restricted as to where on the screen he can type. Function keys are used to signal the system and accept the data typed.

### BRAND-MODEL

A code representing the merchandise brand name and the manufacturer's model number. All items in DPS are identified by brand model.

### CARRY

A release type for merchandise conveyed to the customer at the point-of-sale. DPS assumes that carry releases take place as soon as the item become paid-in-full.

### CASH DRAWER

A device associated with a register terminal which contains the till for one cashier. DPS opens the cashier's drawer automatically when tendering a transaction contains payments.

### CASHIER

A DPS user given tendering licence. The cashier normally operates a register terminal, equipped with a cash drawer and a ticket printer.

### CASHIER CLOSE-OUT

The process by which the cashier drawer is counted and totals compared to that calculated by the DPS. All cashiers must close before the store deposit can take place.

### CATEGORY

A gross distinction of merchandise. typical categories are ACE, Accessories, Video, etc.

**CLASS**

A three digit number code used to group brand-models. Classes are a finer distinction than category.

**COMMENT**

An order detail containing unformatted text. Comments also appear in certain other DPS database records.

**COMMISSIONED EMPLOYEE**

The employees receiving commission and spiff for an item detail. DPS supports splitting of commission between two employees.

**CONFIRMED RAIN-CHECK**

A rain-check for which the customer has confirmed his/her interest in the merchandise, subsequent to the merchandise becoming available. Merchandise is reserved for confirmed rain-check.

**DATA-LINK**

A data path between DPS nodes. The data-link is said to be available when the node can communicate with the central node (HP/3000).

**DATABASE**

The Collection of files on the DPS node.

**DEFECTIVE**

An inventory disposition indicating that the item is not saleable and is in need of repair.

**DELEGATE LICENCE**

A process by which a user awards, temporarily or permanently, a DPS licence to another user.

#### DELIVERY

Transportation of merchandise from a client's delivery location to the customer's site. DPS also supports negative deliveries(fetch) to address returns of delivery merchandise. DPS prints release tickets to support deliveries.

#### DELIVERY LOCATION

A client's location originating and managing deliveries. Prior to the delivery, the item must reside at the delivery location or to be transferred there. A delivery location is a release location for delivered items.

#### DEPOSIT SALE

A deposit sale allows the customer to make down payment in order for an item to be transferred to the selling location for pick-up. No service charges are applied and full payment is expected when the item arrives. Special orders which are filled but are not fully paid are also treated as a deposit sale.

#### DESTINATION CODE

An indicator assigned to stock tag records for defective items. Valid destinations are service, PSC, Write-off and hold.

#### DETAIL

The pieces that make up a transaction. Transactions typically consist of adding line-item, payment and release details to an order.

#### DISPLAY

A disposition describing items that are reserved for display in the store . Display items are not intended to be sold, but DPS provides a facility for doing so, should that become necessary.

**DISPOSITION(INVENTORY)**

A designation pertaining to inventory. A given item can have only one disposition. Dispositions are saleable, open box, display, defective, layaway, pickup, delivery, reserved by deposit and reserved by rain-check.

**DISTRESSED**

A damaged open box item that is considered saleable but not in perfect condition.

**DISTRIBUTION**

The system which controls movement of inventory from the distribution center to the store warehouses. Distribution includes automatic replenishment, allocations and transmittals.

**DISTRIBUTION CENTER**

One of a few large warehouses used to receive vendor shipments and distribute inventory to the stores

**DOWNLOAD**

The movement of data from the central computers to the remote nodes.

**EMPLOYEE NUMBER**

A six digit number uniquely identifying each DPS user. The employee number is required each day to activate the corresponding access key.

**FETCH**

A release type used for returns of merchandise that are picked-up at the customers site. DPS treats fetch as reverse of delivery.

**FIELD**

A single piece of data consisting of one or more characters. Fields appear on the block mode forms are associated with a prompt or column header.

#### **FORM**

A set of prompts and fields used in the block mode. The terminal user is restricted to typing on a single form at any one point in time, even through several forms may be displayed simultaneously.

#### **FUNCTION KEY**

A key on the terminal keyboard with special meaning to the system. Keys may either permanently defined, with meanings on the key itself or "soft" defined by the program. The meanings of the soft keys are displayed on the terminal screen.

#### **HELP**

A DPS feature which displays instructions to the user on request. The system automatically selects the data associated with the form currently on the users terminal.

#### **INTANGIBLE**

A pseudo brand-model used to represent charges for services as opposed to merchandise. ESP's are intangible product.

#### **IN-TRANSIT**

It indicates that the item is expected on a transfer or transmittal but has not yet been received.

#### **INVENTORY LOCATION**

The DPS location of an item at the time of sale

#### **LAYAWAY**

A sale type used to allow the customer to make payments while the item is held for him at the release or inventory location. DPS applies service charge for layaways.

**LICENSE**

A capability to perform a certain DPS function. Licenses are assigned to individual DPS users. Some licenses cannot be used without an accompanying password.

**LINE-ITEM**

An order detail that represents goods or services, usually for which the customer is charged.

**LOCK(RECORD)**

A reservation placed upon a record in the DPS database while a transaction affecting the record is in progress. Lock prevents two users from making conflicting changes to the database.

**MENU**

A DPS subsystem which provides access to interactive functions on DPS. Menu performs basic security validation.

**NET-SALEABLE**

Boxed and unreserved items ready for sale.

**NODE**

A computer in the DPS network. The CC-100 is called a remote node. The HP/3000 is the central node.

**OPEN BOX**

Merchandise that has been removed from its factory package.

**ORDER**

A collection of information that describes a related transactions between a customer and the client.

**ORDER ENTRY**

A DPS subsystem that support the creation and tendering of orders.

**PASSWORD**

A secret code word or phrase required to complete certain transactions and functions requiring special security.

**PAID-IN-FULL**

A status on a written sale line item indicating that full amount of the item is included in the minimum payment due. Thus the item must be fully paid to tender the transaction. Only paid items can be released.

**PROCESSED SALE**

A completed sale and release. Prior to the release of the merchandise, the transaction is called a written sale. Prior to a sale becoming a processed sale, income, spiff, and tax liability are deferred.

**PSC**

Product Service Center. A location where merchandise is staged for return to vendor. The return may be for credit or repair.

**RAIN-CHECK**

A sale type used when normally stocked(Programmed) merchandise is not available at the inventory location. No down payment is required and the merchandise item is expected at the inventory location through routine distribution.

**RECEIVING LOCATION**

The "Ship-to" location on a transfer or transmittal.



**RELEASE**

The act of physically conveying an item to the customer. A sale is not considered processed until released. A item may not be released until it is fully paid.

**RELEASE-DATE**

The date on which a release is expected or scheduled to take place. DPS does not allow a release date for an item until the item is paid-in-full.

**RELEASE TYPE**

A code in a merchandise item detail describing how the item is to change hands. The valid types are: Carry, Pick-up, Today pick-up, Delivery, Service, and Fetch(Retrieval).

**RESERVED ITEM**

An item not available for sale due to some other commitment. Items are typically reserved for open orders.

**RETURN**

A return reverses a processed sale. Only released items may be returned.

**SALES COUNSELOR**

A DPS user given Order entry license but not Tendering license.

**SAVE TICKET FILE**

A file of orders in process or awaiting tender.

**SELLING LOCATION**

The location which creates the transaction on an order for the given line-item detail.

**SERVICE CENTER**

A location where defective items are repaired.

**SHIPPING**

The DPS transaction entered when transfer goods leave the shipping location.

**SKU**

A number which links fungible brand-models together for sales and distribution purposes. DPS automatically alters brand-model store like items when appropriate.

**SPECIAL ORDER**

A line item type used when the item is not available at the inventory location.

**SPIFF**

An Additional compensation associated with a particular brand-model.

**SPOOLING**

A DPS subsystem that manages printers and print files.

**STOCKROOM**

The part of a store where small inventory is kept. Large stores have also a warehouse.

**TENDER**

The final step in the order entry interactive process. It is normally done by the cashier. Following Tender, posting applies the transaction to the database. An order must be tendered to have any permanent affect on inventory, cash or accounting.

**TRANSACTION**

A collection of details on an order created by a single user.

**TRANSFER**

A record created in DPS describing the movement of items between two locations.

**TRANSMITTAL**

A set of records downloaded from the distribution system, describing a shipment of merchandise from the distribution center to the stores.

**USER**

An individual who is licensed to do functions on DPS.

**WAREHOUSE**

The large stock room for a super-store.

*Bibliography*

---

BIBLIOGRAPHY

- DISTRIBUTED PROCESSING SYSTEM USER REFERENCE MANUAL
  - M.I.S DEPARTMENT, CIRCUIT CITY STORES.
- STORE BUILD AND MAINTENANCE
  - M.I.S DEPARTMENT, CIRCUIT CITY STORES.
- CC130 INSTALLATION AND MAINTENANCE
  - M.I.S DEPARTMENT, CIRCUIT CITY STORES.
- DISTRIBUTED PROCESSING SYSTEM CONCEPTUAL DESIGN
  - M.I.S DEPARTMENT, CIRCUIT CITY STORES.
- OREGON PASCAL (VERSION 2.2)
  - M.I.S DEPARTMENT, CIRCUIT CITY STORES.

