BOOKSHOP INVENTORY MANAGEMENT SYSTEM (BIMS)

By,

CHANG LOONG PO
(WET 98018)

Supervisor
PROF. MADYA DR. RODZIATI ZAINUDDIN

SESSION
2000/2001

CONTENTS

Chapter 2 – Literature Review

Chapter 3 – System Analysis

Chapter 4 – System Design

List of Tables

List of Figures

ABSTRACT

All undergraduates of the Faculty of Computer Science and Information Technology, University of Malaya are required to carry out an final year thesis project in computer or information technology related topic during their 3 years course in the university. This subject is one of the conditions must be fulfilled before they graduate. This final year thesis project takes 2 semesters and brings out 9 credit hours of total 120 credit hours.

I, one of the students in the faculty, propose **Bookshop Inventory Management System (BIMS)** as my thesis title. I was assigned to and placed under the supervision of Prof. Madya Dr. Rodziati Zainuddin and moderated by Prof. Madya Dr. Selvanathan.

The project objective is implementing a computerize system for a bookshop to control their inventory (books). This system involved 3 parties, which are bookshop administrator, bookshop employee and customer.

BIMS is divided into 3 main modules: Administrator Module, Employee Module and Customer Module. Basically, administrator module is concerned about the back end support and control. Employee module is a provide a platform for cashier do book selling (Transaction). Finally, customer can use this system to do book searching, browsing to get the information from bookshop database.

ACKNOWLEDGEMENTS

Firstly, I would like to thank Faculty of Computer Science and Information Technology that offer this opportunity to carry out my final year thesis project.

Then, I would like to take this opportunity to express my appreciation to my thesis project supervisor Prof, Madya Dr. Rodziati Zainuddin and project moderator Prof Madya Dr. Selvanathan for their advice, suggestion and support when I carried out my project.

I also wish to thank my course-mate Miss Song Hooi Nak that share her knowledge, experience and useful information with me.

Last but not least, for those encouragement and give helping hands whenever I faced problems.

1.1 Project Overview

The management processes associated with maintaining optimal inventory levels presents numerous and continual challenges to the goods-oriented small business. In each of these firms the design and implementation of a particular inventory planning system should originate with the underlying strategic planning policies within the business. These decisions are conditioned by numerous financial and structural concerns. It is important to note that research has shown that inventory-planning discrepancies have created difficulties for small business such as a bookshop, each of these areas must be carefully considered.

As Robinson, Logan and Salem [20] have reminded both researchers and managers, inadequate inventory planning has been demonstrated to be one of the major causes of small business failure [8; 14]. Indeed, others have underscored the importance to the small firm of effective inventory management. In a study by Wichmann [23], 22 percent of the firms under review encountered inventory control problems in their operations. Without question, these studies suggest that small business managers have numerous inventory planning options available to them. They may select from among techniques that range from judgmental heuristic processes to elaborate computer-based models. For example, the traditional economic order quantity method combines intuitive appeal with straightforward mathematics. However, given the widespread availability of computer technology and a more complete understanding of sophisticated relationships in inventory management, other more complete understanding of sophisticated relationships

in inventory management, other more complex methods have evolved. Among these newer methods are Just-In-Time Production Technology, and Flexible Manufacturing Systems. With this wide range of choices available, it becomes extremely crucial to the small business manager that the most effective inventory management system is implemented.

1.2 Project Definition

Inventory. In its most traditional definition inventory is on-hand quantity of goods and material. Yet, in today's increasingly networked business environment, inventory has become the most vexing problems for wholesale distributors and at the very same time, a golden opportunity. Inventory investment can be the single most distinguishing characteristic of a successful distributorship or it can be the death knell for a less disciplined company. Given the potential of future offers, it is reasonable to expect that the distributors who master the inventory question will survive and thrive.

Management is any contribution to the planning, controlling and decision making to an organization. It includes leadership which is the process of influencing positively the activities of others and assisting their efforts to achieve a particular goals. Under this heading, consideration may be given to activities such as a policy-making, engagement in review activities, engagement in equity activities, promotional activities and public relations, contribution to planning and governance including the holding of management position within the organization.

The bookshop inventory management system is a project, which is developed in order to ease the management of a bookshop either for headquarter or braches especially regarding the inventory. Inventory in this project is not referred to academic books only, but it refers others product such as stationary.

1.3 Objectives

1. To facilitate and smooth the inventory management of a bookshop.
2. To save costs for long term and short term maintenance.
3. To increase efficiency of the inventory management.
4. To facilitate the future maintenance and enhancement.
5. To speed up the data retrieved process.
6. To avoid people carelessness which is not necessary.

1.4 Scopes

The implementation of a Bookshop Inventory Management System Project is to enhance small bookshop inventory management system by using computer-based with some popular technology in business field.

Generally, this system will integrate 3 modules which is:

- Administration module
- Employee module
- Customer module

1.4.1 Administration Module

On the system administrators or manager-level employers side, all information related to their bookshop inventory is presented. This module provides some user-friendly forms for system administrator to input, edit, delete, saving records such as book title, author's name, price, publisher and so on.

The system concerns about inventory level as well as publisher information and customer information. Customer is referred to who registers as a member of the bookshop.

When level of some inventory is below a figure which determined by system administrator, bookshop inventory management system will alert administrator. Beside this alert function, administration module has ability to generate certain report, daily selling report, weekly or monthly best selling report, inventory level for example.

1.4.2 Employee Module

The employee module provides end-user (operation employee) to conduct transaction (book selling). This module interacts with certain hardware such as barcode reader to ease selling process. After a successful transaction being done, the system print out receipt automatically.

1.4.3 Customer Module

The bookshop inventory management system provides end-user (potential customer) do book searching using a PC. System lists book status (available or not available), location of the book (which rack), and a short description to the user.

If Customer cannot get the book (out of stock or bookshop not order at all), they can fill an ordering form through this online system. Administrator makes decision whether order the book from publisher or not.

1.5 Methodology / Software Process

The software process is the set of activities and associated results which produce a software product. These activities are mostly carried out by software engineers. CASE (Computer-Aided Software Engineering) tools may be used to help with some process activities. These are four fundamental process activities which are common to all software processes. The activities are:

1. Software specification
2. Software development
3. Software validation
4. Software evolution

In this software development process, the methodology that adopted is *waterfall* approach. *Waterfall* approach takes the above activities and represents them as separate process phases such as requirement specification, software design, implementation, testing and so on. After each stage is defined it is "signed-off" and development go on the following stage.

1.5.1 The *"waterfall"* model

This first explicit model of the software development process was derived form other engineering processes. This is probably the most widely employed model in software development process. It involves stepwise systematic well-defined steps from starts to finish of the software life cycle. Because of the cascade from one phase to another, this model is known as the "*waterfall*" model. There are numerous variations of the process model. The principal stages of the model map onto the fundamental development activities:

1. Requirements analysis and definition
2. System and software design
3. Implementation and unit testing
4. Integrate and system testing
5. Operation and maintenance

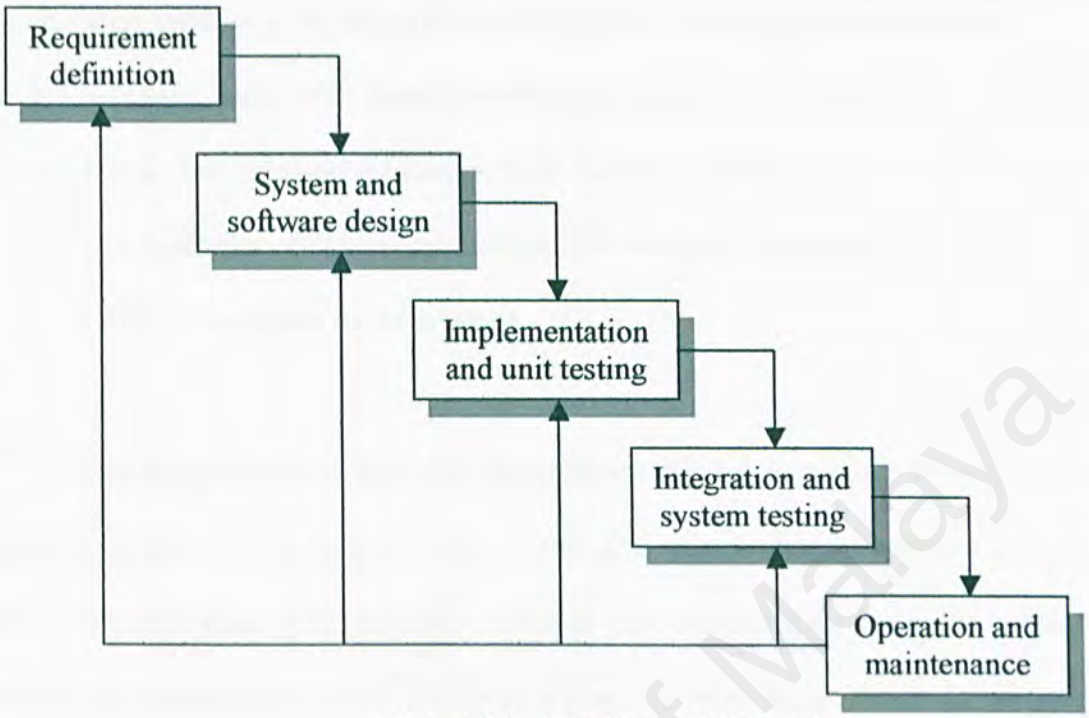

Figure 1.1 The "waterfall" model

The fundamental features of the "*waterfall*" model are:

1. Each phase is culminated by a verification and validation activity whose objective is to eliminate as many problems as possible in the products of that phase.
2. As much as possible, iterations of earlier phase products are performed in the next succeeding phase.
3. A formal product review determines whether or not the product is in satisfactory shape to proceed to the next phase i.e. whether or not a 'milestone' has been reached, if not, the process reverts to phase one.
4. If the product is satisfactory, it is 'Baselined' (i.e. put under a formal change control process)

The baseline product in the Waterfall model has the following main advantages:

1. No changes are made thereafter without the agreement of all interested parties.
2. The higher threshold for change tends to stabilize the product.
3. The controller of the management process achieves the goal of having at any time a definitive version of the product.

The waterfall model presents the software life cycle as a set of sequential sub goals to be achieved in pursuit of the overall goal of a successful software product. In fact, to say that these goals are approached in pure sequence for the entire project is a convenient over-simplification. The picture presented here is just an idealized model. In reality, two refinements of the waterfall model are generally in use. These are:

1. Incremental Development: This is a refinement of both the 'do it twice' full prototype approach and of the 'level-by-level' top-down approach. It holds that rather than these two approaches; software should be developed in 'increments of functional capability'. This method was used successfully as a refinement of the waterfall model on extremely large software products such as the $100m Site Defense project.
2. Advancemanship: This is further divided into anticipatory documentation and Scaffolding. It is analogous to advancemanship in advertisement and political campaign.

1.6 Project Scheduling

Project scheduling is a particular demanding task for software managers. Managers estimate the time and resources required to complete activities and organize them in a coherent sequence. Project scheduling involves separating the total work involved in a project into separate activities and judging the time required to complete these activities. Usually, some of these activities are carried out in parallel.

In estimating schedules, it should not assume that every stage of the project would be problem-free. If the project is new and technically advanced, certain parts of it may turn out to be more difficult and take longer than originally anticipated.

A guideline for estimating is to estimate as if nothing will go wrong then increases that estimate to cover anticipated problems. A further contingency factor to cover unanticipated problems may also be added to the estimate. This extra contingency factor depends on the type of project, the process parameters (deadline, standards, and so on), and the quality and experience of the software engineers working on the project.

The project schedule is usually represented as a set of charts showing the work breakdown, activity dependencies and staff allocations. Bar charts and activity networks are graphical notations which are used to illustrate the project schedule.

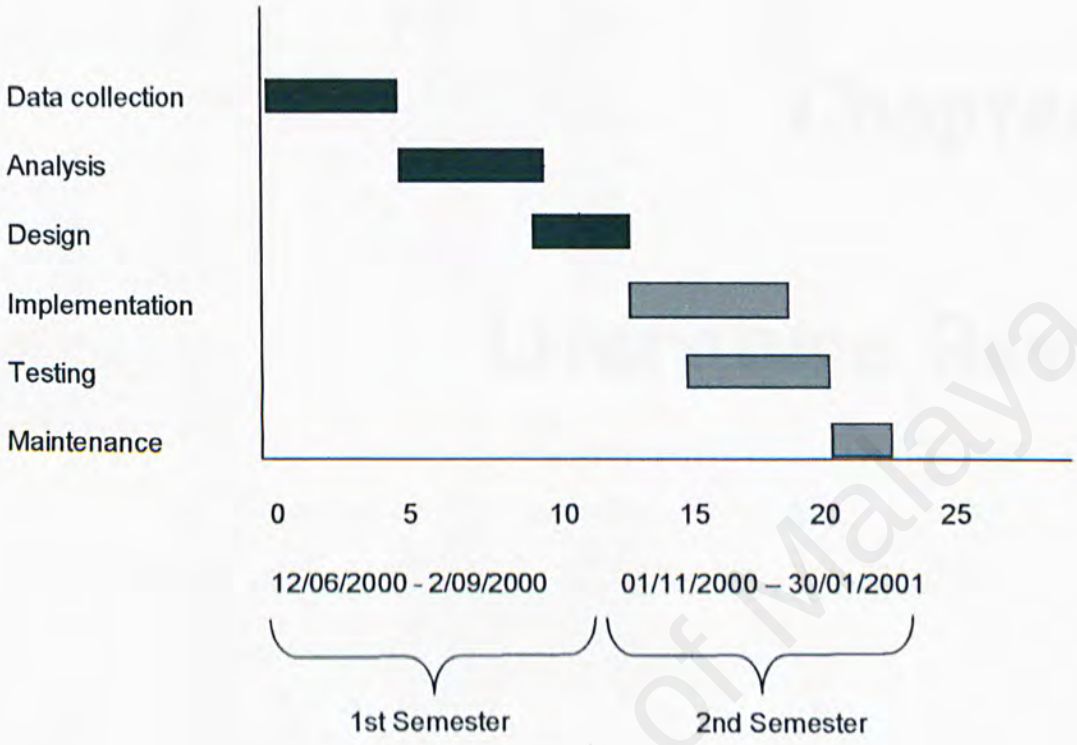

Figure 1.2 - Project Schedule

2.1 Inventory Management

There are several definitions of Inventory Management. Among them are:

(a) Inventory Management is "***the practice of planning, directing and controlling inventory so that it contributes to the business' profitability***". Inventory management can help business be more profitable by lowering their cost of goods sold and/or by increasing sales.

(b) Inventory Management is "***making sure that items are available when customers call for it, but not too much stock so that inventory turnover goals are met***" - Juhi Gonzales, Inventory Management and Systems Consulting-

(c) Inventory Management is "***the art and science of managing to have the RIGHT PRODUCT, at the RIGHT TIME and PLACE, in exactly the RIGHT AMOUNT, at the BEST POSSIBLE PRICE***".

2.1.1 Introduction

The key decision in manufacturing, retail and some service industry businesses is how much inventory to keep on hand. Inventory is usually a business's largest asset. The instant inventory levels are established, they become an important input to the budgeting system. Inventory decisions involve a delicate balance between three classes of costs: ordering costs, holding costs, and shortage costs.

Before venture further, what does inventory mean? According to the Merriam-Webster Dictionary, Inventory is defined as," the quantity of goods or materials on hand"

Inventory is also known as "an itemized list of goods or valuables, with their estimated worth; specifically, the annual account of stock taken in any business" by the online Dictionary.Com.

Stock can be divided into 2 kinds:

a.) Base stock - that portion of inventory that is replenished after it is sold to customers.

b.) Safety stock - the second portion of inventory that is held to protect against the impact of uncertainty.

Without safety stock:

Average inventory = 1/2 order quantity

With safety stock:

Average inventory = 1/2 order quantity + safety stock

There are several types of inventory. Raw materials, purchased parts and supplies work-in process, and component parts are inventories to many businesses. Businesses also need tools, machinery and equipment as part of their capital inventory. In most businesses however, finished goods are mainly the consistent inventory, especially for small businesses.

2.1.2 Reason to Hold Inventory

Most businesses hold inventory for many reasons. Among them are:

- **Meeting unexpected demands:** The chain of supply and demand really comes into consideration here. Business people know that consumers expect goods and services when they need them. Thus, businesses usually stock up their inventories to meet these unexpected demands. These demands may result in overcrowding of inventories because we never know when the storm strikes and consumers would flock to buy the items.
- **Smoothing seasonal demands:** With the comings and goings of major events and the changing seasons, most businesses have inventories at hand to smoothen the seasonal demands. For example, Christmas is just round the corner. With the coming season, retail outlets as well as other businesses are busy meeting and stabilizing the upcoming Christmas demands of consumers. If they do not have any inventory, how can they meet these demands?
- **Taking advantage of price discounts:** When a business purchase goods from the manufacturers and suppliers, they usually get price discounts if they buy in bigger bulks. Manufacturers and suppliers give these discounts to attract and maintain regular buyers. Taking advantage of price discounts is helpful at times but one must always remember not to overstock the inventory because inefficient buying may cause failure of the business.

- **Hedging against price increase:** Businesses usually hold inventory to avoid from the ever-fluctuating market price of inventories. Thus, by having efficient and good inventory system, businesses can control their inventory cost.
- **Getting quality discounts:** When businesses have inventory in store, they can get quality discounts because they know which goods and services to buy from the suppliers and manufacturers. It helps to learn where to get better deals than no deal at all.

2.1.3 Method to Supervise Inventory

The success of a business depends on how well the owner(s)'s ability to maintain adequate quantities of items sold. Records provided by an inventory control system should call attention to the need for reorder when necessary or eliminate "dead wood" inventory when called for. Inventories are controlled and supervised by three (3) methods:

(a) **Perpetual Inventory Control:** The perpetual method is the most frequently used method. It is more costly than the other two but it is an efficient way of keeping count. In this system, complete data records are kept on each item of merchandise and additions or subtractions are made with each transaction. There is an

inventory balance plus a receipt of sale, minus the actual sale to reflect the quantity at hand.

(b) **Actual Counting Piece:** This is another method used to control and supervise inventory. It is used to actually count inventory item-by-item. This is an exhausting task and not many companies or businesses do it. Salespeople are usually involve in this process and there is a large margin of error to be considered as the salespeople go through the monotonous and tiring task of counting everything.

(c) **"Looking It Over":** The third method is "Looking over" the inventory. It is the easiest and cheapest way of controlling and supervising inventory, but there is bound to be errors. With this method, it is hard to pinpoint the inventory levels, the items that need to be ordered, and the items that the store is overstocked with. Almost all financial statements that include inventory figures based on this method cannot be completed accurately.

2.1.4 Cost of Inventory

There are 2 kinds of cost:

1. Visible cost
    i. Ordering cost – Cost of replenishing inventory
    ii. Carrying cost – Cost of holding an item in inventory
    iii. Shortage cost – Temporary of permanent loss of sales when demand can not be met
2. Hidden cost

Having inventory constantly at hand is good but sometimes there are hidden costs that would prove to be a menace for businesses. These costs include could cause:

- Longer lead times
- Reduce responsiveness
- Underlying problems are hidden rather than being exposed and solved
- Quality problems are not identified immediately
- No incentive for improvement of the process

Table 2.1 – Cost of inventory

Cost	Example
Ordering Cost	(a) Clerical costs of preparing purchase orders (b) Some spent finding suppliers and expediting orders (c) Transportation costs (d) Receiving costs (E.g. unloading and inspection)
Carrying Cost	(a) Costs of storage space (E.g. warehouse depreciation) (b) Security (c) Insurance (d) Forgone interest on working capital tied up in inventory (e) Deterioration, theft, spoilage, or obsolescence
Shortage Cost	(a) Disrupted production when raw materials are unavailable: • Idle workers • Extra machinery setups (b) Lost sales resulting in dissatisfied customers (c) Loss of quantity discounts on purchases

2.1.5 Importance of Inventory Management

1) Inventory management can help business be more profitable by lowering their cost of goods sold and/or by increasing sales.

Consider a typical company - ABC Company with the following income statement:

Sales	$	2,000,000
Cost of Goods Sold		1,100,000
Gross Profits		900,000
Gen. Administrative Expenses		402,000
Marketing Expenses		350,000
Net Income before taxes	$	148,000
		==========

Not bad -- return on sales is over 7%.

Now, suppose that through application of sound inventory management principles, ABC Company was able to reduce the cost of goods sold by 3%. And because there are fewer inventories, let's say that carrying costs (warehouse storage charges, insurance, finance charges, etc) is reduced by 2% of the general administrative expense. Those minimal cost reductions result in significant increase on net income:

Sales	$	2,000,000
Cost of Goods Sold		1,067,000
Gross Profits		933,000
Gen. Administrative Expenses		394,000
Marketing Expenses		350,000
Net Income before taxes	$	189,000

Small costs reductions due to application of sound inventory management principles resulted in very significant increase (28%) in net income!

Lower cost of goods sold is achieved by making the inventory smaller and therefore turn more often; while making sure that stocks are large enough will result in increased sales because products are available when Customers call for it. Inventory management is balancing those two opposing factors for optimum profitability.

2) *Conduct an inventory audit that answers the following questions:*

- What is your inventory turnover performance? Are you carrying too much inventory and paying more for interest and/or storage charges?
- If you can improve your inventory turnover performance, how much will your gross profits increase?
- What is your service level performance? Are you losing potential or current Customers (and revenues) because you are out-of-stock of the products they want?
- If you can improve your service level, how much will be the increase in your sales revenues?
- How accurate are your records?
- Are you losing Customers (and sales revenues) because your inventory records are not accurate enough – they show you have some in stock but actually, none?
- Or are you overstocked on some products because your records showed that you didn't have any and you ordered some, when in fact you have lots? (Please note: Quality of answers to these questions will depend on available records or information maintained by your organization.)
- Set up either a manual or computer-based inventory control system to better manage inventory.
- Where required, provide contractual project leadership from installation into conversion and start-up of new system. Where also required, develop a procedure manual and train staff in using new system.

- If an inventory system is in place, review current processes to make it better. Where required, provide contractual inventory management to carry out recommendations and resolve unsatisfactory conditions.

3) *Improve Customer Service*

4) *Reduce Inventory Investment*

5) *Increase Productivity*

6) *Prevent Poor Inventory Record Accuracy*

Inventory record errors are costly. No computer system, be it old or new, will work properly if the transactions are not entered correctly. The costs of poor inventory record accuracy are not always apparent to management. Consider the following results, all of which increase production costs and reduce profits:

- Unanticipated stock-outs
- Decreased production efficiency
- Higher investment in safety stocks
- Requirement for staging of items to determine availability or shortages
- Invalid data for inventory replenishment system
- More obsolete and excess inventory

Some of these costs can be quantified. Others are intangible, but nevertheless do exist and can be substantial. It is important to have inventory records, which are accurate. Most experts agree that this accuracy must be at least 95% and even higher for critical or high unit value items. The key to accurate records is the implementation of a sound cycle counting system.

2.2 Inventory management software

Service centers live or die on how well they control their inventory. Increasingly, service-center operators are relying on computer systems to help them buy at the lowest cost and sell at the highest margin.

Inventory management software can perform many functions, and it can be tailored to each company's needs. For example, many service centers rely on their systems to signal when it is time to reorder each item. Some use bar codes to track the movement of metal into and out of their warehouses. Others use more sophisticated systems that allow them to track the cost associated with each sale so they can set prices that allow them to meet preset profitability goals.

Before investing in a new inventory control system, experts say, service centers should first set goals on what they want to accomplish. The systems they consider should offer real-time data in a flexible manner so that the system can be tailored for the facility's particular needs. The system must also be "scalable" so that it can be expanded as the company grows, without deterioration in performance.

"A lot of companies don't set a target or goal for what they want a system to do for them. That can be defined without a whole lot of investment," says Brian David, national sales manager for Compusource, La Palma, Calif.

"The expectations and priorities set before you even begin to look for an inventory management software system are extremely important," adds Dave Cyrus, product manager, Enmark Systems, Ann Arbor, Mich.

As a starting point, service centers should quantify their current operations in terms of volume moved, number of transactions, average number of invoices issued per month and so on. Then they should set quantifiable goals for improvement, such as additional inventory turns, the elimination of out-of-stocks, or a reduction in man-hours.

The software provider should help define these goals as the initial step in the system implementation, which typically takes three to six months.

Experts agree that software should be flexible enough to run on a variety of platforms. The most common operating system for inventory management is Unix, but some packages also run on Windows.

2.2.1 Software As Sales Support

Salespeople need to know what inventory is on hand in order to maximize their selling opportunities. Today's software systems can give them real-time access to inventory levels, as well as each customer's sales history and buying habits.

"Service-center customers are looking for an immediate response and they want fair pricing," says Scott Deutsch, vice president of marketing at Prophet 21, Yardley, Pa.

The more sophisticated the inventory management system, the more information it offers about what is happening to the service center's stock. Some reports commonly available through these systems include scrap, usage, remnants and reorder points.

Other systems may generate reports that detail each customer's order history and pricing, who bought specific sizes and shapes of metal, the profit made on each sale and the cost of carrying specific sizes and shapes in inventory.

Other reports quantify the number and size of deliveries, material purchasing costs and labor costs.

"By having up-to-date information of what actual yields are, your buying plan is more accurate and you can reduce inventory," says Joel Reed, senior manager of industrial marketing for J.D. Edwards Co., Denver.

2.2.2 Caution over customization

Service centers are often tempted to build inventory management programs that are customized to their particular operation, but experts urge caution. The program might be left without support, especially if a key programmer leaves the company.

System integration may be another problem with customized software. Most systems built for service centers incorporate multiple programs such as accounting, personnel, mill test reporting, sales analysis and on-line order entry/quoting. These programs must be able to share information.

"You can look at it two ways," Cyrus says. "Customization can make you a software orphan. On the other hand, you don't want absolute rigidity."

He recommends avoiding too much customization, except for forms specific to the operation. Reed emphasizes the importance of flexibility and scalability in a software system. "No matter where you think the business is going, things change," he says. "You want software that can change with you."

2.2.3 A Checklist for Choosing Software and Product and Provider

What make for a good vendor to supply your inventory management system? Well, there are more than a score of factors to consider.

- **The company management:** How long has the company been in the business? How long has its management been around?
- **Specialization:** Is metals processing and distribution its primary or sole market?
- **Strategy:** Are revisions a regular occurrence? What percentages of the provider's customers are on the most recent revision? What percentage of these customers subscribe to the full enhancement and support programs?
- **Development:** How much effort is put into research and development? What is the annual budget for R& D>?
- **Technology:** What are the company's plans for e-commerce, the Internet?
- **Staff:** Does the company use its own staff or part-time staff and contact programmers? What is the staff turnover rate? Is support available 24 hours a day, seven days a week?
- **Internet support:** Is it available? Are they plans to develop it?
- **Stability:** Are audited balance sheets available for inspection?
- **Legal story:** Are there are any outstanding legal dispute with customers or suppliers?

- **Track record:** How many successful implementations does the company have? Were these implementations completed within the customer's budget?
- **Assistance:** Are trainers, consultants and support staff available to help when your system goes live?
- **Documentation:** Is the documentation complete? Can it be modified to your company's requirements?
- **Product history:** When was the last major rewrite?
- **Customer lists:** Ask for a customer list. Determine which previous customers share commonalties with your company and talk to them.
- **The product Flexibility:** Will the software be able to handle your unique product situations?
- **Modifications:** Are custom modifications reintegrated into the product or left as a unique module? Database: Does it use an open, non-proprietary database, such as Oracle or Informix?
- **Database access:** How easy is it to access the database? Can standard industry tools, such as Microsoft Excel, Microsoft Access, Crystal Reports and other SQL products, be used to access the information?
- **Open systems:** Does the software and its database operate on any of the major open systems, including Windows NT and Unix?
- **Customer service:** Does the system allow your sales staff to access customer orders with minimal information? How easy is it for your sales staff to follow up on past shipments or shipment history?

2.3 Inventory management system

Traditional issues and currents methods that too many distributors and dealers use when viewing their parts inventory, thus missing the opportunity for an addition profits.

To be more profitable shouldn't we just improve sales?

Certainly sales play a key role in profitability. Without sales, there would be no reason to be in business in the first place. However, the buying center affords the single largest opportunity to have a direct impact on profitability.

Imagine if you could cut the cost of your inventory by 30%, while at the same time reducing out-of-stock conditions, thus improving your ability to deliver what customers need and want to buy. Merely increasing sales will not attain these benefits. On the other hand applying scientific inventory management principles will. Wouldn't you agree: sales generate revenue, buying generates profit?

What's wrong with using "turns" as an inventory management strategy?

If you're using "turns" as your only measurement of inventory management success, that may mean you are foregoing valuable discount brackets, profitable forward buying and economic buying cycles. While these issues may have a negative impact on your turn rate, you need to be able to quickly and consistently analyze a buying situation's profitability-regardless of its impact on your turns.

Doesn't our current ERP (Enterprise Resource Plan) system effectively handle our inventory?

Currently most distributors have some computer software in place. Typically the Inventory/Replenishment component is part of a larger group of software designed to run the full spectrum of an enterprise's system (ERP). Sometimes an equipment manufacturer, on a monthly lease basis, has provided the software. The vast majority of these systems count and track inventory quite well and even provide a basis for replenishment. However, they fail to recognize inventory replenishment as an asset or profit contribution center. Their rather broad-brush focus makes them unresponsive to the rapidly changing marketplace condition occurring today. They don't have the strength or depth to handle and evaluate inventory down to the individual products and locations, which is essential if you are going to squeeze out additional costs and increase profits.

What exactly is scientific inventory management?

Before the computer, inventory was tracked and replenished manually using "stock cards." Buyers kept a file containing parts numbers and descriptions, stock status, little notes, hints and hunches that assisted them in formulating the correct level of parts to stock. Needless to say, the computer has allowed us to provide a great deal of "science" along with the "art" of buying.

Slide rules, mathematical tables, the Wilson Formula, profitability theory, weighted averages, exponential smoothing-all these tools come into play in determining what to buy, and most important, when to buy. Like brain surgery, few care how it happens-only that it does happen.

The biggest single problem we have found is that the people who would benefit the most from the use of scientific inventory management - the owners or top management of a company - typically don't fully understand the mechanics and profit points of inventory replenishment. They are denied the opportunity to measure and realize profit contributions from their buying group.

What kinds of profit results are being obtained?

Our experience shows varying results based on type of industry and dollar amount of inventory. Certainly evidence shows inventory levels being reduced a minimum of 20% and up to 40%. Customer service levels have either been sustained, if they were already in the high 90% range, or have been improved up to those levels. Overstock conditions, as well as obsolete and aged inventory issues are diminished substantially, and the high costs of intercompany stock transfers are being significantly reduced. Many system solutions provide some of these benefits on a one-time ROI but are unable to continue to improve the profit contribution on an annualized basis. (In the next article I will address how to sustain these profit opportunities on an annual basis.)

2.4 A study on successful inventory management system

2.4.1 Norton Super Abrasives Tool Crib

The Norton Super Abrasives Tool Crib houses 220+ types of grinding wheels, plus, all sorts of gloves, nuts & bolts, coffee supplies, safety equipment, pens, markers, rags and an assortment of other low-cost items. The tool crib manager tried his best to keep track of the inventory of all these items in the Tool Crib, but was overwhelmed by the variety and daily flow of stock in and out of the tool crib. As a result of the high traffic and the periodic stock-out situations that arose, the Tool Crib manager began to keep far greater inventory of grinding wheels in the Crib than was actually needed. Needless to state, the yearly spending on grinding wheels inventory rose to over 300K (1996 figure). Norton management decided that an inventory system should be implemented to manage the diverse stock of grinding wheels.

The Inventory System Specifications

The Tool Crib Manager needed a system that was friendly, smart, and low-maintenance. We wanted a system to have the following qualities:

1. The system should be ***user friendly***, equipped with a user interface that can lead the administrator through the various software elements quickly, easily and robustly.

2. The system should be ***scaleable***. Today they want to address the Tool Crib inventory, but tomorrow they want to track/manage other inventories.
3. For each item, the system should ***store information*** on
   (a) Stock levels,
   (b) Re-order points,
   (c) Re-order quantities,
   (d) Lead times,
   (e) Vendor information,
   (f) Usage histories.
4. The system should create and use ***barcode*** labels to identify each item. Bar coding technology is not new, but it is a tremendous efficiency enhancement tool.
5. The system should ***verify, record and track*** the usage of grinding wheels to individual operators. We charge our manufacturers for the grinding wheels they use. Since each operator works for one line, we should be able to assess the individual line grinding wheel cost.
6. We should be able to integrate this system with our AS/400, BPCS MRP system.

These were the more important of our specifications, but there were others.

Choosing the Inventory Management System

With the specifications in hand, it was simply a matter of finding a system that met the specifications, and had a price that was commensurate with its value. Norton Co. looked at several systems..., but after much deliberation..., they chose SumTotal's Material Management Software to solve our inventory problem.

The Results

The MMS implementation process included the following steps:

1. Setting up item locations and the storage room layout are the first aspects to attack.
2. Then, created a database of items (item #, item description, on-hand quantity, vendor, location). The data entry process for 218 items took about 6 hours; this included the time to enter vendor information, etc. It's very important to have all the information about an item on hand at the time of the initial item entry. Leaving out information adds more data entry time later on in the process when you find out its needed. One of the plusses of this system was that you could update the on-hand quantity with a cycle count. We only did a cycle count once to bring the system to an operational status.

3. Norton Co. printed barcode labels for each item and affixed them on racks. We then did the cycle counts (counted each item, entered quantity into a portable hand-held scanner from Intermec, Inc.).

After the item and location labels were set up, and the cycle counts were done, the system was ready to run. The entire process from setting up the tool crib to operational status took approximately a month and a half.

They had some problems along the way. Initially we did not have pricing information, so after we got them (prices) in January, Norton had to go through the process of entering them for each item, a process that took about 5 hrs. Most of our problems were due to coming up the learning curve. Understanding the purchase module took a few hours; this was by far the most complicated module. They approached it thinking it would be a giant and this approach really hurt us. The module was easier to operate than they expected.

The software maintenance side was a breeze. Norton had to do daily back-ups on a floppy, and if they had a software fault, they could call for support at SumTotal. The support was excellent. SumTotal installed remote access software that allows them to do real-time maintenance via a modem connection. Norton kept a duplicate copy of the software on an Omega zip drive. System recovery is as simple as downloading the zip file into a new C: directory, restoring our data from the back-up floppy, and running the

software. They had to do this once and it took approximately 5 minutes to get the system running again in tip-top shape. The Intermec equipment is under a 24 hours replacement policy (which Norton has used on the wedge scanner). The MMS system will run without the wedge, scanner or barcode printer, but operators will have to type and you will not be able to print labels until your new printer comes in. But this should not be an issue; Norton have been running all of the equipment 24 hrs a day, for than a month now with no problems.

2.4.2 Ericsson

Ericsson uses Forklift-mounted Computers with Integrated RangeLAN2 Wireless LAN Adapters to Help Maintain Real-time Information

The efficiency of warehouses and distribution centers is rapidly being acknowledged as a critical factor in the success of a business. Some innovative warehouse managers are gaining a competitive advantage by implementing real-time inventory management systems that utilize wireless local area networks (LANs).

These wireless solutions help increase staff productivity, improve space utilization, and reduce errors in shipments and data-entry. In addition, real-time information can help enhance customer service and assist in key business decisions.

One warehouse that successfully implemented such a wireless inventory management system is the Richardson, Texas facility of Ericsson, Inc., a subsidiary of telecommunications leader L.M. Ericsson Telefon AB. The Richardson facility manages the distribution of millions of cellular phones and accessories per year throughout North and South America.

The Problem

As a result of Ericsson's worldwide success, the Americas distribution facility in Richardson quickly outgrew the paper-based system it used for inventory management. With this original system, employees tracked incoming inventory shipments on paper and transferred the data into a terminal on a traditional wired computer network. When a customer order needed to be filled, a list was printed out and given to forklift drivers in the warehouse, who gathered the products and checked them off the list. The completed list was handed off to a computer operator who then noted that the products were taken from inventory.

Although the process was state-of-the-art when implemented several years ago, it did not provide real-time data, a capability that improves inventory management. It also required redundant data entry, which was inefficient and increased the possibility of human error.

"We wanted to allow our forklift drivers to enter data directly into the network without stopping to pick up and drop off paper-based orders," said Bill Birgbauer, the engineer in charge of inventory management processes at Ericsson's Richardson facility. "This would make us more efficient in the warehouse and give our finance division critical inventory information in real-time."

The Solution

Based on a recommendation from Ericsson's parent company, Birgbauer began working with system integrator MA Systems of Sweden, which had implemented a comprehensive inventory management solution for Ericsson's distribution facilities in Europe.

The solution Birgbauer adopted included a customized application from MA Systems running on Fujitsu Stylistic 500 RF touch screen computers with fully integrated RangeLAN2 wireless LAN adapters from Proxim, Inc., and barcode scanners. The computer, keyboard and scanner are mounted on the forklift, giving drivers easy access to the devices.

As shipments arrive at the warehouse, receiving clerk's type information related to the contents of incoming pallets into the inventory management application on the Windows for Workgroups Ethernet network. They then print out bar-code stickers and

put them on the pallets. A forklift driver picks up the pallet, takes it to an open area, scans the barcode and enters its location into the computer.

Now, when drivers start their shifts, instead of picking up a paper order, they simply log into the computer on their forklift. The first order comes up on the screen, showing the product needed and its location in the warehouse. As products are "picked" from the shelf, drivers scan the associated barcodes, and note that the required numbers of products were taken. The computer prompts the driver to confirm that the correct product and quantity were collected. When an order is completed, the next order comes up on the drivers' computer screen. The drivers no longer need to stop to pick-up and drop off paper orders, and no redundant data entry is required, since it all happens in real-time.

As forklift drivers enter this inventory-change information, the data is sent at a fast 1.6 Mbps to a Proxim RangeLAN2 Access Point, which links the wireless computers to the Ethernet network. Access Points are placed throughout the warehouse to create an in-building cellular-like network that allows drivers to stay connected regardless of their location. "We were very impressed with the range we got from even just one Access Point. It nearly covered the whole warehouse," noted Birgbauer.

From the Access Points, the information goes to an in-building hub that is wired to an AST Pentium-based PC server on the Richardson campus. It then goes to a communication server that sends the data to a Digital Equipment Corporation host computer at the finance division in Lynchburg, Virginia.

Because it is simply an extension of standard Ethernet, RangeLAN2 plugged right into Ericsson's existing network. According to Birgbauer, "The installation of the system was simple, and it's easy to manage."

In addition to being compatible with the standard wired network, RangeLAN2 is an "open" wireless network that gives Ericsson the choice of using a broad set of other wireless mobile computers and RF barcode scanners on the same wireless backbone, in addition to the Fujitsu computers. More devices are compatible with the RangeLAN2 network than any other wireless LAN.

"By completely automating inventory management and using the RangeLAN2 wireless LAN, we simplified the driver's job, while maintaining the most up-to-date information. In the warehouse this improves efficiency, data accuracy and productivity. For our finance group it enables better management of our inventory and customer orders.

"The RangeLAN2 wireless LAN has been a key to the success of this system," concluded Birgbauer.

2.4.3 McKesson

Problems

There must be a better way to manage a massive inventory that's costing you millions in tracking errors.

Solution

McKesson HBOC invested in an automated distribution system that equips workers with wearable handheld computers.

Until recently, the world's, largest health care services provider, McKesson HBOC, was dumping about $28 million a year into a black hole of inaccurate shipping, billing, and bad inventory. Sales of its pharmaceutical and health and beauty aides languished.

The old paper-based system was "inaccurate, slow, and nonproductive," said Tom Magill, vice president of logistics technologies at the San Francisco company. "Mistakes happened all the time. We'd bill someone for item A when we shipped item B, or we'd have shortages. We'd bill them for 10 of something but ship 5. It costs millions of dollars a year to process a return, rebill, restock, and put it on the customer's shelf."

In 1994, Magill embarked on a massive automation plan that is only now coming to a conclusion. To improve accuracy in all facets of its supply chain, the company settled on a warehouse/inventory management package from Uniteq, then customized it and called it Acumax.

The company eventually installed Uniteq's IBM OS/400-based application on each of its 35 AS/400 minicomputers in its distribution centers. Using a spread-spectrum radio frequency (RF) network, infrared scanners, and wearable handheld computers from Symbol Technologies, warehouse workers could more accurately and quickly receive, record, bill, and ship inventory orders. The system reduces human error because employees simply scan the bar code on items and automatically upload the information to the management database.

Ultimately, the company reached its goals—and realized a one-time cost-savings of $17 million, plus an annual return of $28 million. But not without complications: Its time and cost estimates were off by two years and $33 million.

When it went live with the first automated distribution center in August 1994, the company was hit with installation costs, debugging nightmares, costly AS/400 upgrades, and integration and cabling issues.

"It was hell," Magill said. "Invoices, pricing, and billing were all a mess. It took several months to get under control. We evaluated it very hard to see if we wanted to proceed."

After a rocky start and repeated requests to the board of directors for more money – what was estimated to be an $18 million project ended up costing $51 million – McKesson proceeded slowly.

By 1995 only three warehouses were online. The company has since stepped up the pace.

Was it worth the cost and headaches?

"We manage orders better and at a lower cost," Magill said, adding that McKesson's warehouse workers were initially resistant to the drastic change but now appreciate being relieved of the tedium of manually counting inventory.

Symbol's RF network and scanning technology leave little room for error, he said, which lowers the risk and costs linked to bad inventory management. (McKes son has been known to stumble upon $120 million in inventory it didn't know it had.)

The next step is to tie the Acumax inventory system into the company's suppliers and to bring everyone online with electronic purchase orders.

"We're going to be better off the more we bring online," Magill said. "We've already exceeded inventory accuracy and paper savings targets [by about 8 percent], which we now use as a selling tool for prospective customers."

The Result

"Remember to smile as you explain your $51 million budget request to upgrade – of all things" the company's aging inventory management system. It'll make the news easier to swallow. So will pointing to McKesson HBOC's venture, which ultimately took five years but tied a secure tourniquet around the estimated $28 million a year the health care service was losing to waste and inefficiency. It's no wonder: The company's $2.5 billion stockpile inventory, distributed among 35 centers, was overly dependent on paper and human input. The upgrade, however, uses an inventory management app, AS/400 minis, and plenty of symbol networked wireless handheld.

- **Time it took:** Five years to automate 35 distribution centers.
- **Cost:** $51 millions
- **Saving:** A one-time savings of $17 millions and $28 annual savings because of reduced inventory management errors.

- **Lessons learned:** Don't give up just because your project gets off to a rocky start, says Tom Magill, vice president of logistics technologies. McKesson was undeterred by $33 million in cost overruns and two years added to the project. As the world's largest health care services company, its management saw that was an improvement worth investing in.

2.6 Conclusion

Business must have methods and procedures that offer ample flexibility to meet unusual and sometimes unreasonable demands on their resources -- personnel, equipment and facilities and operational. Exceptional customer service also includes providing top quality products at reasonable costs.

Businesses must keep a careful rein on their inventories. Having too much inventory and/or not having enough stock are considered primary direct causes of business failures.

In any business, make it big or small, we must understand that taking good care of our inventory is very important. If we as managers do not understand the concept of good inventory management, we must learn to be familiar with it and its applications. One of the reasons for the failure of a business is its inventory management. There are many ways to fight failure, and we can start from here. There are new technologies that can help us maintain and supervise our inventory. What we can do is learn, implement and evaluate our business. And we can start with our inventory.

Chapter 3

System Analysis

3.1 Introduction

Systems analysis is the study of a current business and information system, and the definition of user requirements and priorities for a new information system. Systems Analysis involves three basic phases:

- Feasibility assessment (Project Survey Phase)
    - Interviews
    - Project scope
    - Problem statement and classification
    - Proposed project plan
- Organization problem statement (Project Study Phase)
    - Project roles
    - Learn current system (use repository)
    - Model the current system
    - Analysis of problems and opportunities
    - New system's objectives
    - New project scope and plan

- Organization requirements statement (Definition Phase)
    - Identify requirements
    - Model system requirements
    - Discovery prototype
    - Prioritization
    - Review requirements

The aim of system analysis is to analyze, specify, and define the system, which is to be built. The models developed in this phase will describe "what" the system will do.

The benefits of system analysis are:

- Provides an overview of the business
- Defines the scope of the system
- Defines "what" the system will do
- Provides a solid understanding of the system

3.2 Requirement Definition

A software requirements definition is an abstract description of the services that the system should provide and the constraints under which the system must operate. It should only specify the external behavior of the system. It should not be concerned with system design characteristics. Consequently, the requirements should not be defined using an implementation model. The definition should be written in such a way that it is understandable by customers without knowledge of specialized notations. The notations used for requirements definition must therefore be based on natural language, forms and simple intuitive diagrams.

System requirements may be either functional or non-functional requirements:

- ***Functional requirement:*** These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do.
- ***Non-functional requirement:*** These are constraints on the services of functions offered by the system. They include timing constraints, constraints on the development process, standards and so on.

3.2.1 Functional Requirements

- Inventory database
    - System should have the ability to store inventory detail (book's title, publisher, author, price, quantity, edition)
    - System should display alert when the inventory level below predetermined numbers.
    - Inventory data should store systematic.
    - System should output inventory information whenever searching by administrator or potential customer.
- Access permission
    - System should recognize people that operate it by checking the login name.
    - System should run the system according login name. For example, administrator has the authorities to access administrator module and same thing for employee and customer.
- Automatic update inventory changes
    - System should reduce inventory level each time after successful transaction.
    - Administrator can do manual edit on the inventory data.

- Book searching and browsing
    - Customer could use this system to find the book they want to buy.
    - System should tell customer that the book status (price, quantity on hand)
    - A short description should be outputted for every success finding.
- Member registration
    - Customer could register to become a member of the bookshop
    - Special discount could be give to those registers as member.
- Online ordering (Place an order through customer module)
    - When the customer finds that the book on hand is not enough, they could do an online ordering.
    - Customer could do the same thing for the book not in the bookshop or new released book.

3.2.2 Non-Functional Requirements

- Data communication
    - Reliable intranet service should be provided. If the network run improperly may impact business deeply
    - The intranet should provide easy access and retrieval of inventory data by administrator, employees and customer.

- User interface design
    - System should have an attractive interface.
    - System may use graphics interface rather than textual interface.
    - System should provide an user-friendly interface
- End-user guideline
    - System should implement with the help file for the new user.
    - The end-user could refer to the help file when they face any problem regarding the BIMS.
- Respond time
    - The data retrieval speed should be fast.
    - Searching process must not take long time.
- Hardware
    - Upgrade processor speed, RAM (Random Access Memory) and disk storage space for a better performance.
- Software
    - Software should compatible with the software that develops the system.

3.3 Technology Consideration

3.3.1 Programming Language

Programming language that being chose should:

1. Have ability of creating user interface easily.
2. Provide a graphic interface.
3. Compatible with the database server that being used to store data.
4. Error handling.

Consideration result: Visual Basic 6.0

3.3.2 Database Server

Database server should have the below characteristics:

1. Store large amount data.
2. Concurrent access.
3. Security element
4. Compatible with the programming language that uses to develop the system.
5. Accessible in LAN (Local Area Network)

Consideration result: Microsoft SQL Server 7.0

3.3.3 Platform

The system should run in a platform which:

1. Provides a network environment
2. Support database server and programming language that develop system
3. Reliability
4. Real-time

Consideration result: Microsoft Windows NT 4.0

3.4 System Requirements

System requirements consist of 2 components:

1. Software requirements
2. Hardware requirements

3.4.1 Software Requirements

- Windows NT 4.0
- Microsoft SQL Server 7.0
- Microsoft Visual Studio 6.0 (Microsoft Visual Basic 6.0)

3.4.2 Hardware Requirements

- Windows NT Server
- Microsoft SQL Server
- PC with the specification:
    i. 450 MHz processor or higher
    ii. 64 MB RAM or higher
    iii. 4.0 GB of Hard disk or higher capacity

4.1 Introduction

System design is the evaluation of alternative solutions and the specifications of a detailed computer-based solution. It is also called Physical Design. System design is composed of three phases:

- Select a design target from a candidate solutions (Selection Phase)
- Acquire necessary hardware and software (Acquisition Phase)
- Design and integrate the new system (Design and Integration Phase)
    - General design
    - Detailed design

System design can be represented into 2 components:

1. Program design
2. Database design

4.2 Program Design

BIMS has 3 main modules:

1. Administrator module
2. Employee module
3. Customer module

4.2.1 Login Module

Every time when BIMS is required to run then the program will prompt out a login form for end-user (administrator and employees) to do login by enter user name and their password. The program will run on module that is allowed to access. But, for a customer, they do not need to do login. So, customer module is default module for this system.

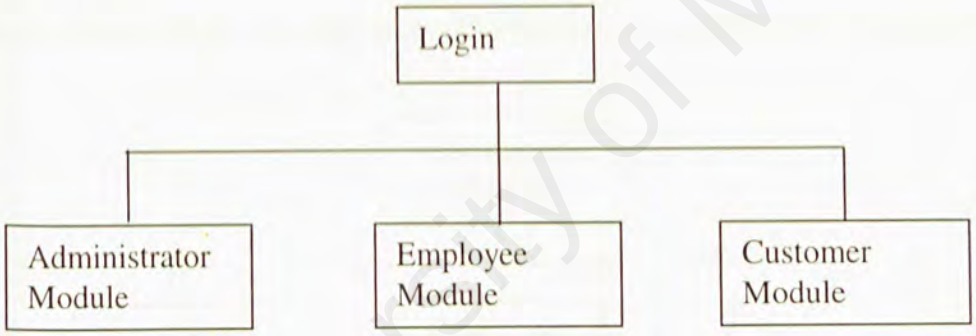

Figure 4.1 - Login Module

4.2.2 Administrator Module

This module is especially for those administrator to do controlling and analysis. Administrator used this module to help them in data entry and editing process. The data involved in this feature are book title, author, selling price, retail price, publisher detail, quantity and so on. Then, administrator uses this system to keep track bookshop transaction and inventory level for internal controlling and make sure bookshop running in a good condition. Besides, administrator module can generate several reports like daily transaction selling report, monthly best selling report, and book below determined level.

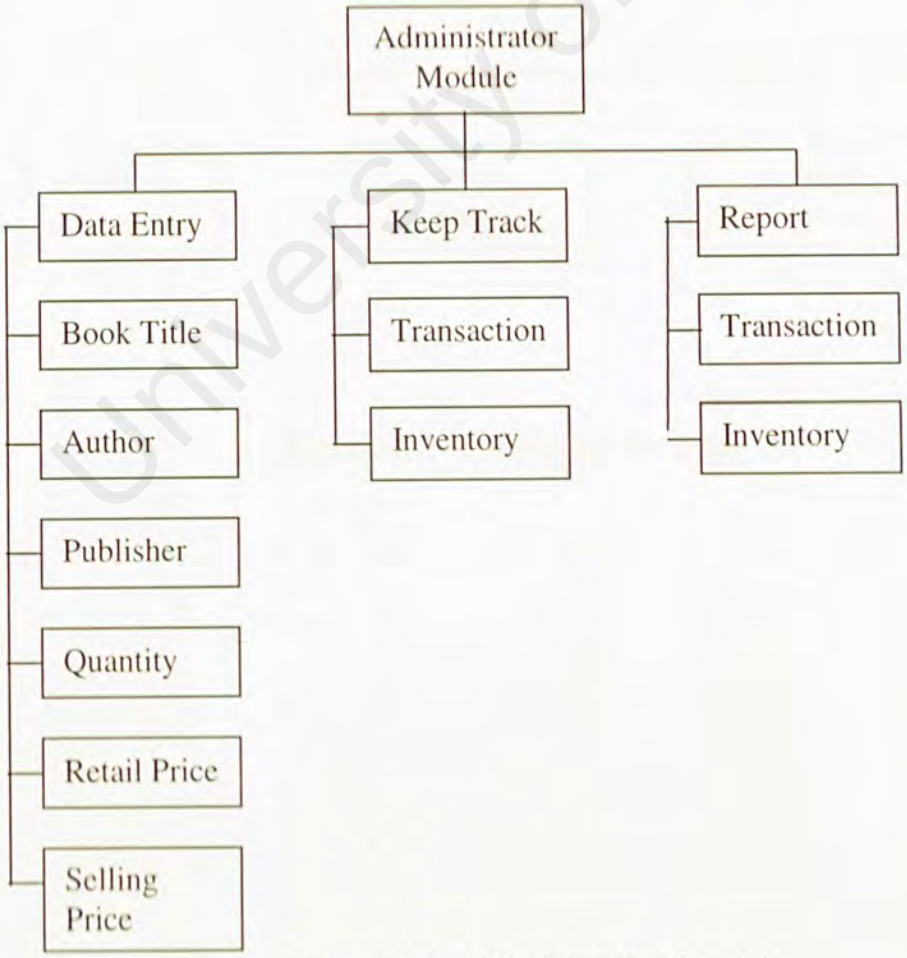

Figure 4.2 - Administrator Module

4.2.3 Employee Module

Employee module is designed especially for the employee those work as cashier. Cashiers use this module to do transaction (refers to the book sale process). Whenever there is a successful transaction has been done, the system will store the transaction. Besides, employee module will automatically reduced inventory number in the database according the sold item through this module.

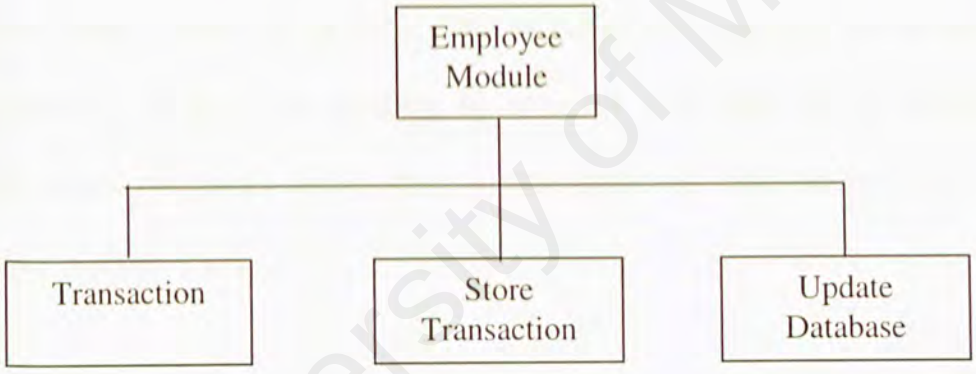

Figure 4.3 - Employee Module

4.2.4 Customer Module

Customer module is a program that is developed according customer needs. Customers use this system to do book searching to get the book they want to purchase. Moreover, all potential customers could browse the bookshop by entering the book title, subject, author and publisher. Then the system will do a searching on database. If the system finds the book that customer want to purchase, it will output some information like book description, price and location of the book. On the system could not find the book or the book already out of stock, then, an online ordering form will be prompt out for customer to do an online ordering by enter the book title, author, publisher and quantity. When the books arrive, they will be informed. But, administrator has the authorities to order or not.

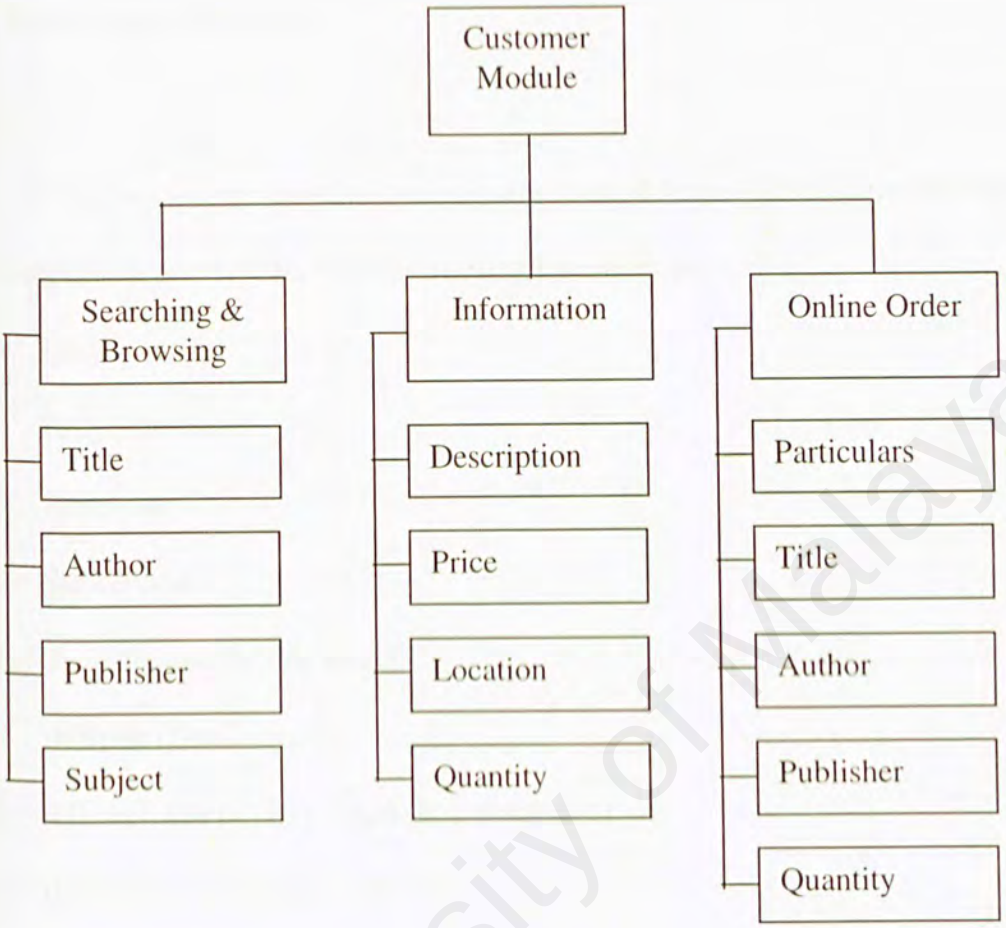

Figure 4.4 - Customer Module

4.3 Database Design

Database – large store of computerize data. It keeps all the important data inside computer or server. The benefits of using database are:

1. Easy to store
2. Easy to retrieve
3. Save cost
4. Save space

12 tables are created for this BIMS:

1. tblBook (Book details)
2. tblEmployee (Current bookshop employee)
3. tblMember (Customer details)
4. tblPublisher (Publisher details)
5. tblCategory (Book category and description)
6. tblOrder (Store customer order)
7. tblTransaction (Transaction details)
8. tblOrderItem (Order item details)
9. tblSaleItem (Sale item details)
10. tblPaymentMethod (Payment method description)
11. tblShipMethod (Ship method description)
12. tblShipStatus (Status of online ordering item)

4.3.1 Data Dictionary

a) **Members table – tblMember**

Table 4.1 - tblMember

Column Name	Datatype	Length	Description
ID	Int	4	Customer's identification (Primary key)
FirstName	Varchar	50	Customer's first name
LastName	Varchar	50	Customer's last name
Phone	Varchar	20	Customer's contact number
Email	Varchar	50	Customer' s email address
Address	Varchar	50	Customer' s address
Street	Varchar	50	Street where customer reside
City	Varchar	50	City where customer reside
State	Varchar	50	State where customer reside
Country	Varchar	30	Country where customer reside
PostalCode	Varchar	20	Customer's address postal code
RegisterDate	Datetime	8	Customer's registered date
MemberStatus	Char	1	Status of the member
LoginName	Varchar	20	Customer's login name
LoginPassword	Varchar	20	Customer's login password

b) **Books table - tblBook**

Table 4.2 - tblBook

Column Name	Datatype	Length	Description
ISBN	Varchar	20	Book identification (Primary key)
Title	Varchar	500	Book title
Publisher	Int	4	Book publisher
Edition	int	4	Book edition
Description	Varchar	1500	Book description
Quantity	Int	4	Quantity of book available
Category	Char	3	Book category
RetailPrice	Money	8	Book retail price
SellingPrice	Money	8	Book sell price
Reorder	Int	4	Quantity that have to reorder

c) **Books Category table - tblCategory**

Table 4.3 - tblCategory

Column Name	Datatype	Length	Description
ID	Char	3	Identification of book category (Primary key)
Description	Varchar	50	Description of the category

d) **Orders table – tblOrder**

Table 4.4 - tblOrder

Column Name	Datatype	Length	Description
ID	int	4	Identification of the customer's order (Primary key)
Member_ID	Int	4	Customer's identification
OrderDate	Datatime	8	Date of the order is placed
ShipAddress	Varchar	50	The shipping address
ShipStreet	Varchar	50	Street where recipient reside
ShipCity	Varchar	50	City where recipient reside
ShipState	Varchar	50	State where recipient reside
ShipCountry	Varchar	50	Country where recipient reside
ShipPostalCode	Varchar	20	Recipient's address postal code
ShipMethod	Char	1	Customer shipment method
ShipStatus	Char	1	Status of the customer's shipment
PaymentMethod	Char	1	Customer paid method
PaymentStatus	Char	1	Status of the customer's payment
CCNumber	Varchar	16	Credit card number
CCExpire	Datetime	8	Credit card expire date

e) **Payment Method table – tblPaymentMethod**

Table 4.5 - tblPaymentMethod

Column Name	Datatype	Length	Description
ID	Char	1	Payment identification (Primary key)
Description	Varchar	50	The types of the payment

f) **Shipment Method table – tblShipMethod**

Table 4.6 - tblShipMethod

Column Name	Datatype	Length	Description
ID	Char	1	Shipment identification (Primary key)
Description	Varchar	50	The types of shipment

g) **Shipment Status table – tblShipStatus**

Table 4.7 - tblShipStatus

Column Name	Datatype	Length	Description
ID	Char	1	Shipment status identification (Primary key)
Description	varchar	50	Status of the shipment

h) **Ordered Item table – tblOrderItem**

Table 4.8 - tblOrderItem

Column Name	Datatype	Length	Description
Order_ID	Int	4	Identification of the customer's order
Book_ID	Varchar	20	Book identification (ISBN)
OrderQuantity	Int	4	Quantity of books has been ordered

i) **Publishers table – tblPublisher**

Table 4.9 - tblPublisher

Column Name	Datatype	Length	Description
ID	Int	4	Identification of the publisher (Primary key)
Name	Varchar	50	Publisher's name
Email	Varchar	50	Publisher's email address
Address	Varchar	50	Publisher's address
Street	Varchar	50	Street where the publisher reside
City	Varchar	50	City where the publisher reside
State	Varchar	50	State where the publisher reside
Country	Varchar	50	Country where the publisher reside
PostalCode	Varchar	20	Publisher's address postal code

j) **Employee table – tblEmployee**

Table 4.10 – tblEmployee

Column Name	Datatype	Length	Description
ID	Varchar	10	Identification of the employee (Primary key)
Name	Varchar	50	Employee's name
Phone	Varchar	20	Employee's phone number
Address	Varchar	150	Employee's address
Email	Varchar	30	Employee's e-mail address
Birthday	Datetime	8	Employee's birthday
StartDate	Datetime	8	Employee's start work date
UserName	Varchar	20	Employee's user name for login
Password	Varchar	20	Employee's password
Category	Varchar	30	Employee's category

k) **Transaction table – tblTransaction**

Table 4.11 - tblTransaction

Column Name	Datatype	Length	Description
ID	Int	4	Identification of the transaction
Date	Datetime	8	Time that transaction is made
Employee_ID	Varchar	10	Employee that done the transaction
Total	Money	8	Sum of the customer purchase

l) **Sale Item table – tblSaleItem**

Table 4.12 - tblOrderItem

Column Name	Datatype	Length	Description
Transaction_ID	Int	4	Identification of the customer's order
Book_ID	Varchar	20	Book identification (ISBN)
SellQuantity	Int	4	Quantity of books has been ordered

4.4 Data Flow Diagram (DFD)

4.4.1 Preface

Data flow diagrams were developed in the early sixties. In that particular period, there has been a lot of writing in professional magazines and books. Because there has been so much writing in the literature about it, nowadays it is a trivial topic. As a consequence of it, it is hard to find any article about Data flow diagrams. Especially about the history and the development throughout the time, there is hardly any information.

4.4.2 General information

Data flow diagrams are diagrams which show the flow of data from one place to another. DFDs describe the processes of a system, showing how these processes link together through data stores and how the processes relate to the users - the outside world. They are used to record the systems analysis as a part of the design documentation. At their lowest level of detail, as we shall see, DFDs are often included in a programmers working specification when the systems analysis is complete and the system is being programmed. [MORT93]

4.4.3 Importance of DFD

Why data flow diagrams? In Dutch there's an expression: ',,n blik zegt meer dan duizend woorden' (one view says more then a thousand words). This is also true for a graphical description model. It's difficult, if not impossible, to describe a model in words and still being clear and not complex. This was one of the main reasons to develop a graphical modeling technique like Data flow diagrams.

Data flow diagrams usually are made after a Context Diagram has been made, because the Context diagram functions as the basis of a Data flow diagram. It's important not to forget that Data flow diagrams are not a model of flow of control or sequence of processing in a system. Data flow diagrams must be seen as a model which shows the flow of data through a system.

Table 4.13 – DFD Symbols

	External Entity	Source or destination of data that is external to the system.
	Process	Manual or computer process that changes data. In the following text a circle is used to indicate a process
	Data Flow	Data transfer in the direction indicated by the arrow. Each arrow should be labeled to indicate what data is being transferred
	Data Store	Manual or computer storage of data

4.4.4 Inventory Data Flow Diagram

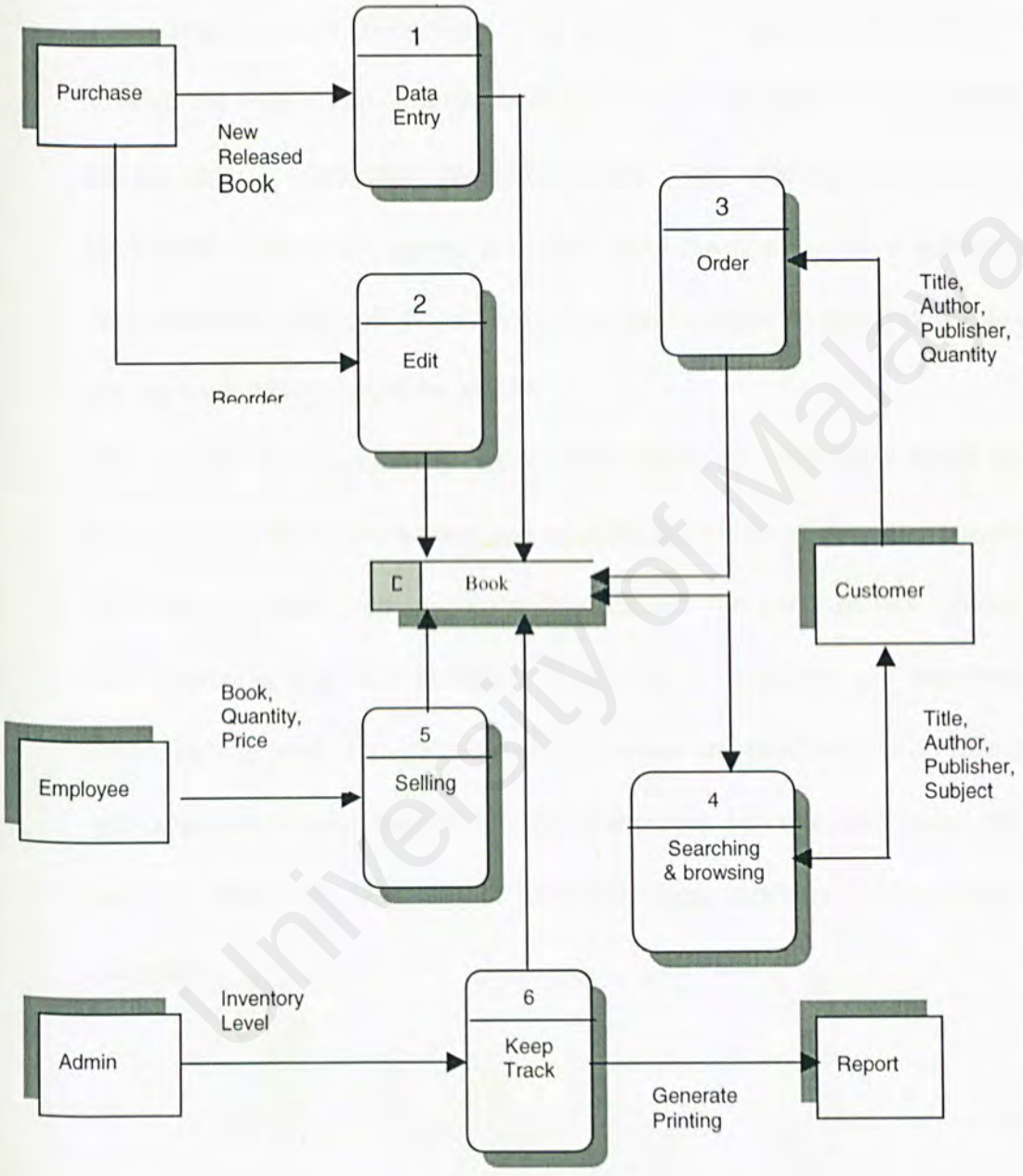

Figure 4.5 - Inventory data flow diagram

Note:

1. When the bookshop purchases new books, the administrator will run administrator module to input the book data. First of all, administrator will check whether the book detail is in the database or not. If the book is found in database, administrator just increase inventory number else administrator will input the book detail such as title, author, publisher, retail price, selling price and so on.
2. Administrator can check inventory level in the database by using query and print out the book which should be reorder.
3. On the other hand, employee can do book selling by input book ISBN (primary key) to search for the book price and calculate total price of customer purchase.
4. Customer involved with inventory data because the will use this system to do book searching and book ordering. When they do browsing and searching, they could find the book through title, subject, author and publisher. Customer module will help them do searching in database and output data like description, price and location. While the book is not available, then customer could do an online ordering.

4.4.5 Transaction Data Flow Diagram

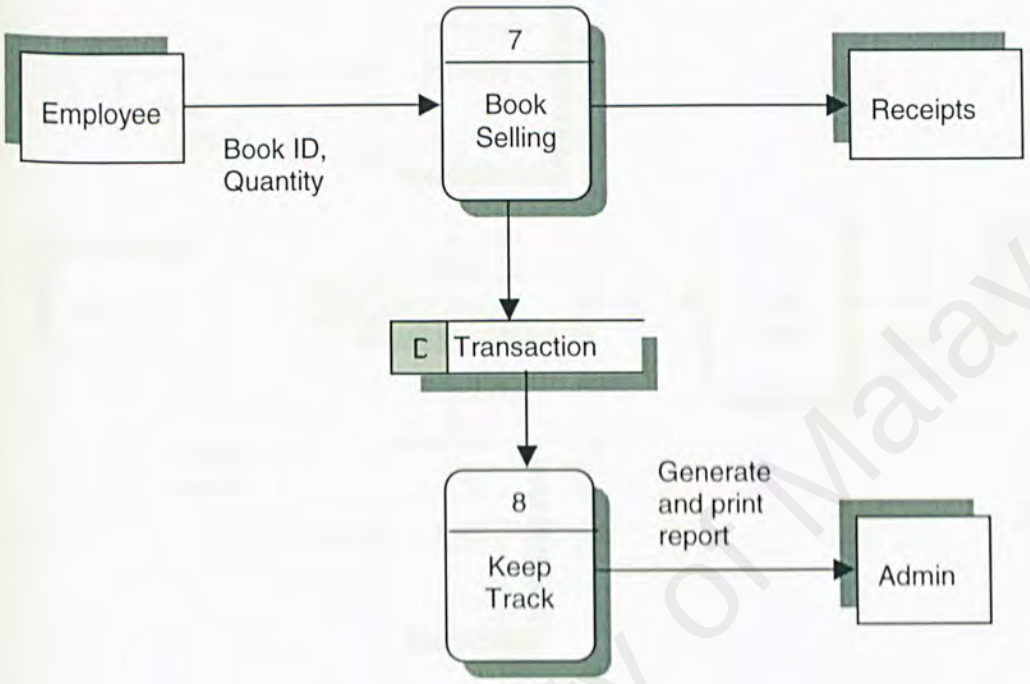

Figure 4.6 - Transaction data flow diagram

Note:

1. Whenever customer purchases books, employee uses employee module to de transaction. The transaction will be store inside database. And BIMS will print out a receipt to the customer.
2. Transaction that stored inside database could be accessed and retrieved easily by bookshop administrator if they want to do auditing.

4.4.6 Member Data Flow Diagram

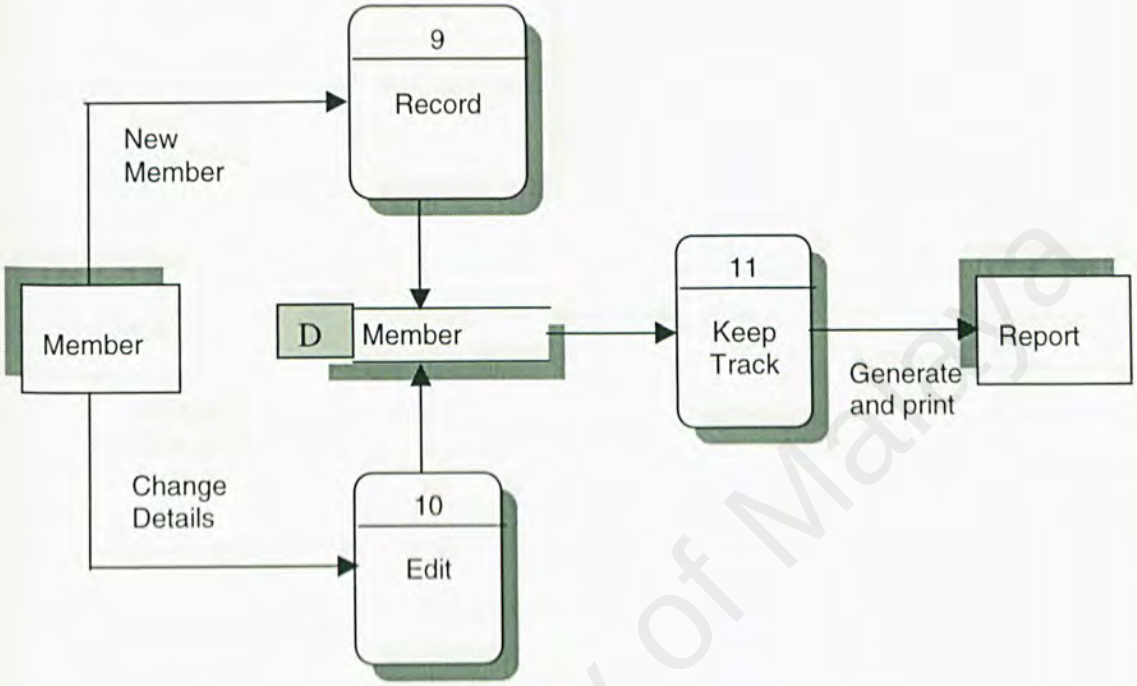

Figure 4.7 - Member data flow diagram

Note:

1. If there is a new member registration, bookshop administrator will run administrator module to input new member data.
2. If a member wants to edit his/she personal details then he/she can ask bookshop administrator to do the changes.
3. Administrator can edit member data according company needs.

4.4.7 Employee Data Flow Diagram

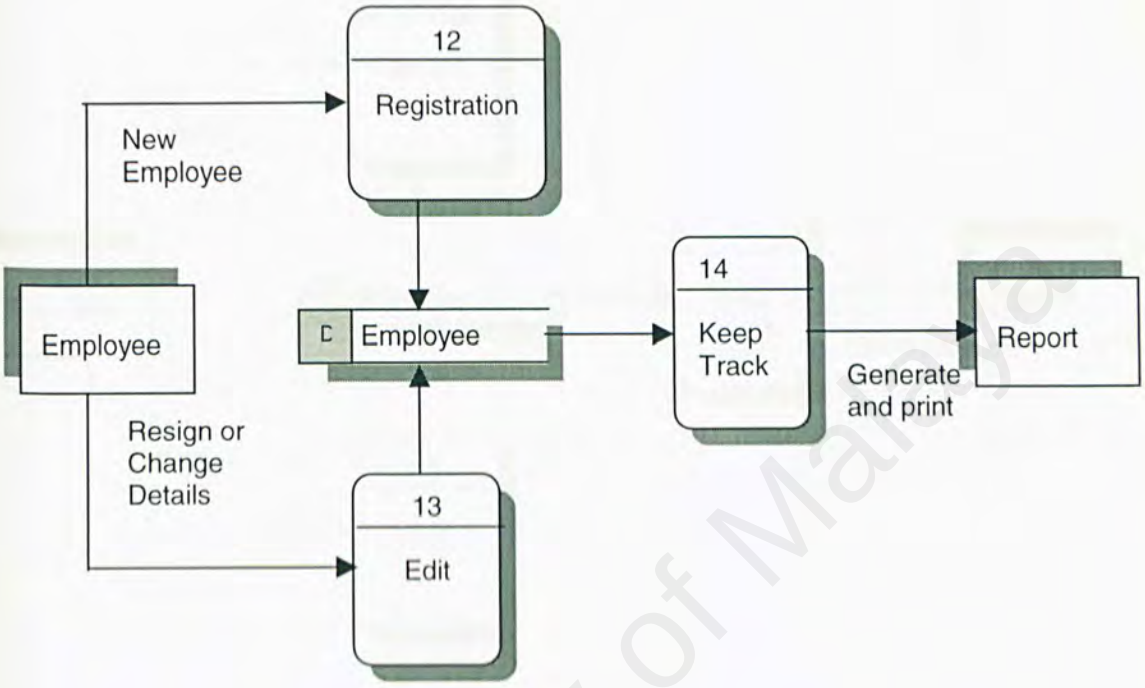

Figure 4.8 - Employee data flow diagram

Note:

1. If a new staff is employed, the administrator will key in he/she personal details and assign a login name for them and give permission for them to access certain function.
2. If the employees ant to change their particular, they can ask administrator to do it.

4.4.8 Publisher Data Flow Diagram

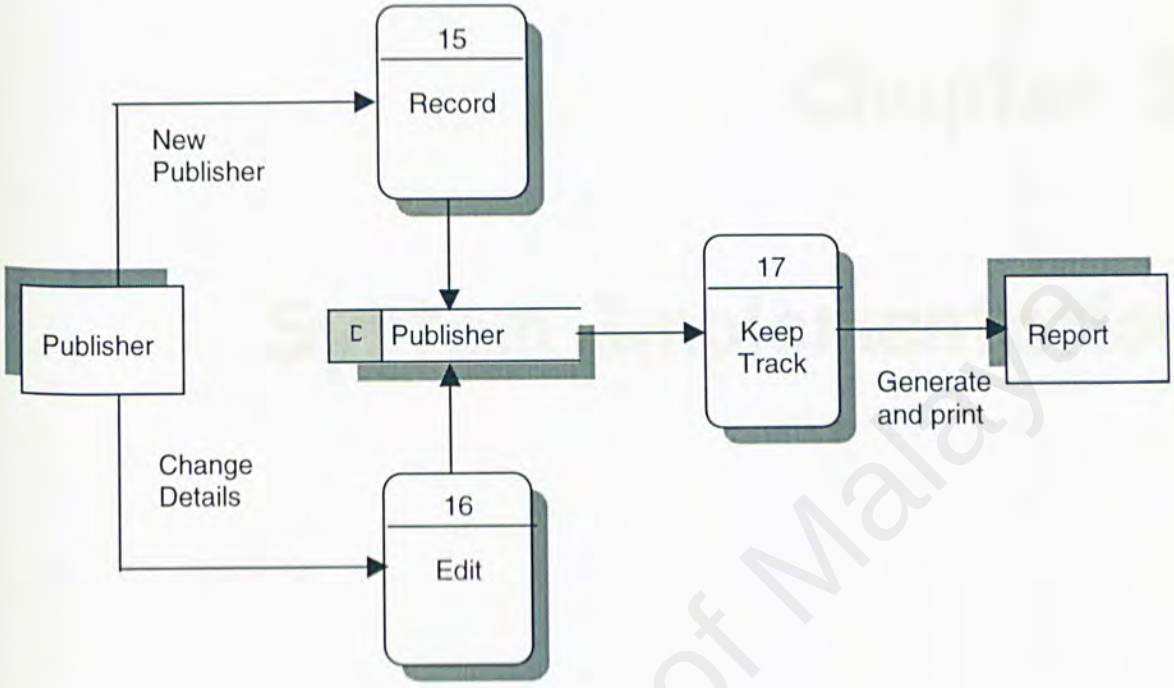

Figure 4.9 - Publisher data flow diagram

Note:

1. If the bookshop has orders with new publisher, administrator will input the publisher detail into database.
2. They also have the authorized to edit publishers' file.

5.1 Introduction

Under this stage, we transformed the design model of the Bookshop Inventory Management System into workable software. The system implementation of Bookshop Inventory Management System will be divided into two components, which are platform development and the modules implementation.

5.2 Platform Implementation

The platform implementations include setting up operating system and database server, which are Microsoft Windows NT Server and Microsoft SQL Server.

5.2.1 Setting Microsoft Windows NT Server

Before the system is being developed, it needs to run under Windows NT server. During the installation of Windows NT 4.0, the hard disk is formatted using NT File system format to ensure a more stable and secured NT transaction across the platform. Main reason to set up this operating system is the multi-users feature. It allows concurrent process by different users.

5.2.2 Setting Microsoft SQL Server

The SQL server is installed in another Compaq server. It is separated from the IIS and Exchange Server. After the Microsoft SQL server has been installed successfully, a database name *Bookshop* is created. Then, we create the table according to the database design. This database will become the database storage for the system. The tables are created for keeping the data used in all modules of the Bookshop Inventory Management System.

We allocated the hard disk space for the database to maximize the performance of the SQL server and to ensure there is enough of space to store the record. The file growth of the database is set to 10% of the original database size.

5.3 Modules Implementation

The Bookshop Inventory Management System has 3 main modules:

- Customer Module
- Employee Module
- Administrator Module

Each module is implemented using Visual Basic, one type of programming languages.

5.3.1 Customer Module

Customer module is designed specially for the bookshop potential customer. With this module, customers can search the book record in database instead of rack by rack. Potential customers are informed that whether we have the book in our stock. Besides searching feature, they can do an order through the system.

For the searching function, customers can search database by book's:

- ISBN
- Title
- Author
- Publisher
- Category

For the order feature, customer can do an order on the book, which is not in stock. Customers are required to input their personal detail such as:

- Name
- Identity Number
- E-mail address
- Telephone Number
- Address

5.3.2 Employee Module

Employee Module is designed specially for the bookshop cashiers. This module not only have the entire feature that customer module has but also transaction function. Transaction function is where the cashier does the sales module. In the transaction function cashier will input:

- Book's ISBN
- Quantity

Then, the system will calculate the total amount that customer should pay and the system will automatically update database after the transaction has stored.

5.3.3 Administrator Module

Administrator module is the place that administrator updates data and keep track inventory. The functions are:

- Edit inventory record
- Edit Employee record
- Edit Publisher record
- Edit Book Category record
- View customer order record

6. 1 Computer system testing

Computer system testing is the analysis and validation of the product as a whole. Computer system testing encompasses all product components, including both hardware and software. Its purpose is to verify that all essential functions and features are present and working properly. Computer system testing often includes testing the system's performance and response times, and validating that all system components function as intended. Computer system testing can also often include a wide range of industry standard tests. These may be performed separately on each system component, jointly on multiple system components, or both.

Standard tests commonly used to validate a computer system may include, but are not limited to:

- Functionality testing to determine how well your product functions based on its stated design specifications.
- Performance testing to identify whether your product functions within the expected performance parameters, and to allow you to compare your product's performance against its competitors.
- Stress/Load testing to verify your system's ability to successfully handle excessive amounts or challenging types of input data.
- Compatibility testing to validate your system's ability to function smoothly with each of the various hardware devices and software programs for which you designed and built it.

6.2 Why use computer system testing?

"If you build it, they will come," use to be the marketing norm. In today's competitive marketplace, however, just building a product is no longer enough. Now we must also:

- Prove that our product does everything we say it does
- Show that our product fits seamlessly into existing and planned networks
- Give the buyer the most for their money, and make sure they know that's exactly what they are getting
- Meet or beat our competition in any number of tests that may be thrown at our product
- Protect our company from potential liability by taking every reasonable step we can to make sure that our product is safe and sound.

6.3 Definition of unit, component and integration testing

Unit. The smallest compilable component. A unit typically is the work of one programmer (At least in principle). As defined, it does not include any called sub-components (for procedural languages) or communicating components in general.

Unit Testing: In unit testing called components (or communicating components) are replaced with stubs, simulators, or trusted components. Calling components are replaced with drivers or trusted super-components. The unit is tested in isolation.

Component: A unit is a component. The integration of one or more components is a component.

Note: The reason for "one or more" as contrasted to "Two or more" is to allow for components that call themselves recursively.

Component testing: the same as unit testing except that all stubs and simulators are replaced with the real thing.

Two components (actually one or more) are said to be integrated when:

a. They have been compiled, linked, and loaded together.
b. They have successfully passed the integration tests at the interface between them.

Thus, components A and B are integrated to create a new, larger, component (A, B). Note that this does not conflict with the idea of incremental integration -- it just means that A is a big component and B, the component added, is a small one.

Integration testing: carrying out integration tests.

Integration tests (After Leung and White) for procedural languages.
This is easily generalized for OO languages by using the equivalent constructs for message passing. In the following, the word "call" is to be understood in the most general sense of a data flow and is not restricted to just formal subroutine calls and

returns -- for example, passage of data through global data structures and/or the use of pointers.

Let A and B be two components in which A calls B.

Let Ta be the component level tests of A

Let Tb be the component level tests of B

Tab The tests in A's suite that cause A to call B.

Tbsa The tests in B's suite for which it is possible to sensitize A

-- The inputs are to A, not B.

Tbsa + *Tab* == the integration test suite (+ = union).

Note: Sensitize is a technical term. It means inputs that will cause a routine to go down a specified path. The inputs are to A. Not every input to A will cause A to traverse a path in which B is called. *Tbsa* is the set of tests which do cause A to follow a path in which B is called. The outcome of the test of B may or may not be affected.

There have been variations on these definitions, but the key point is that it is pretty darn formal and there's a goodly hunk of testing theory, especially as concerns integration testing, OO testing, and regression testing, based on them.

As to the difference between integration testing and system testing, system testing specifically goes after behaviors and bugs that are properties of the entire system as distinct from properties attributable to components (unless, of course, the component in

question is the entire system). Examples of system testing issues: resource loss bugs, throughput bugs, performance, security, recovery, transaction synchronization bugs (often misnamed "timing bugs").

6.4 Unit Test

Unit test is the test of the software elements at the lowest level of development. Units may be aggregates of software elements. Planning for unit test should occur concurrently with the software design activity. Reused software will probably not undergo unit test; unless changes were made to the units. Then, appropriate testing is performed as in regression testing.

General

- Test planning - Establish the objectives of the unit test, the strategies to be employed, the coverage requirements, reporting and analysis, and close-out of anomalies.
- Generate, monitor, and update the unit test plan to accomplish objectives.
- Trace test design, cases, procedures, and execution results to the unit designs.
- Confirm that anomalies during test are software anomalies, and not problems detected for other reasons.
- Generate test cases and procedures - Develop test cases and procedures for unit test and continue tracing as required by software test plans.

- Perform unit test - Check individual software units for typographical, syntactic, and logic errors to ensure that each correctly implements the software design and satisfies the software requirements; execute the test cases; analyze results to verify anomalies; recommend changes to software design or code; and conduct re-testing as necessary.
- Document test activities and results.

6.5 Software Integration Test

The software integration test activity is performed to examine how units interface and interact with each other with the assumption that the units and the objects (e.g., data) they manipulate have all passed unit tests [BEIZER]. Software integration tests check the interaction with other software (e.g., libraries) and hardware. The software integration test schedule depends upon the development and integration schedules for software units, hardware, and other components. For large systems, software integration test planning may require close coordination among all system personnel to ensure that the overall test objectives are achieved by the selected test strategy. For each major integration that has successfully undergone interface and interaction testing, functional tests may be developed and executed [BEIZER]. When all system components have been integrated and have successfully undergone software integration tests, then the system moves into software system test. During software integration test, reused software units are

integrated into the system. It is critical to test that the interfaces are correct, and that the resulting software meets operating requirements.

General

- Test planning - Establish the objectives of the software integration test, the strategies to be employed, the coverage requirements, reporting and analysis, and close-out of anomalies. Ensure that interface testing of reused software to other system software is planned.
- Generate, monitor, and update a software integration test plan to accomplish identified objectives.
- Trace test design, cases, procedures, and execution results to software requirements.
- Generate test cases and procedures - Develop test cases and procedures for unit test and continue tracing as required by software test plans.
- Perform software integration test.
    - Check the inter-unit communication links and test aggregate functions formed by groups of units.
    - Confirm that anomalies during test are software anomalies, and not problems detected for other reasons.
    - Ensure any changes to software requirements, software design, or codes are made. Conduct retesting as necessary.

- Conduct functional, structural, performance, statistical, and coverage testing of successfully integrated units after each iteration of software integration and successful testing of interfaces and interactions.

- Document test activities and results.

6.6 Software System Test

Software system test, in the context of software V&V, involves the conduct of tests to execute the completely integrated system. Software system test is the validation that the software meets its requirements. Validation of the complete system may involve many tests involving all system components. The software system tests exercise those system functions that invoke software to determine whether the software behaves as intended relative to complete system performance. These tests must be conducted in such a manner as to stress the system based on software responses to system inputs (e.g., from sensors, operators, databases). Tests and data collected from the tests are designed to provide an operational profile of the system which support a statistical analysis of the system reliability [MUSA87, MUSA89, BUTLER]. This section of the report addresses only the tests that validate that the software implements the system requirements; other tests for other components and perspectives are necessary for complete system validation.

While software system tests are conducted after the system has been built, it is imperative that planning for these tests is conducted concurrently with the software requirements activity because:

- Analyzing the software requirements for test requirements may result in finding software requirements errors and/or discovery of untestable requirements.
- Establishing test facilities (e.g., model of operational environment) and Computer-Aided Software Engineering (CASE) tools (e.g., test case generators, test database) may require as much time as development of the system.

For reused software, software system test is performed to assure that the software is correct, consistent with prior documentation, complete for use and/or modification, and accurate. At the system level, reused software should be considered part of the system. Tests are in accordance with test procedures. Results are documented and traced as required by the software system test plan.

General

- Test planning - Establish the objectives of the software system test, the strategies to be employed, the coverage requirements, reporting and analysis, and close-out of anomalies.
- Generate, monitor, and update a software system test plan to accomplish objectives.

- Trace system and software requirements to test software design, cases, procedures, and execution results.
- Generate test cases and procedures - Develop test cases and procedures for unit test and continue tracing as required by software system test plans.
- Test the operation of the software as an entity (sometimes a simulated environment may be used); confirm that anomalies during test are software anomalies, not problems detected for other reasons; ensure any changes to software (software requirements, software design, code, or test cases) have been made; and conduct retesting as necessary.
- Document test activities and results.

6.7 Software Installation Test

The software installation test activity is the final step before launching full customer acceptance testing. The purpose of installation test is to demonstrate that the correct software has been delivered and that the software interfaces are correct relative to any interfaces at the installation site. Acceptance testing, which involves the user/customer, is outside the scope of this document.

General

- Conduct an installation configuration audit.

- Determine that all software outputs needed to operate the system are present.
- Check that the software installed in the system is the software that underwent software V&V.

- Develop and execute tests that will examine and stress site-unique parameters (e.g., printer interface, operating system interface, monitor interfaces).
- Generate applicable documentation.
- Generate an SVVR (or generate it at the end of the software V&V process).

6.8 Software QA & Testing FAQ

1. *What are some recent major computer systems failure caused by software bugs?*

- News reports in September of 2000 told of a software vendor settling a lawsuit with a large mortgage lender; the vendor had reportedly delivered an online mortgage processing system that did not meet specifications, was delivered late, and didn't work.
- In early 2000, major problems were reported with a new computer system in a large suburban U.S. public school district with 100,000+ students; problems included 10,000 erroneous report cards and students left stranded by failed class registration systems; the district's CIO was fired. The school district decided to reinstate it's original 25-year old system for at least a year until the bugs were worked out of the new system by the software vendors.

- In October of 1999 the $125 million NASA Mars Climate Orbiter spacecraft was believed to be lost in space due to a simple data conversion error. It was determined that spacecraft software used certain data in English units that should have been in metric units. Among other tasks, the orbiter was to serve as a communications relay for the Mars Polar Lander mission, which failed for unknown reasons in December 1999. Several investigating panels were convened to determine the process failures that allowed the error to go undetected.
- Bugs in software supporting a large commercial high-speed data network affected 70,000 business customers over a period of 8 days in August of 1999. Among those affected was the electronic trading system of the largest U.S. futures exchange, which was shut down for most of a week as a result of the outages.
- In April of 1999 a software bug caused the failure of a $1.2 billion military satellite launch, the costliest unmanned accident in the history of Cape Canaveral launches. The failure was the latest in a string of launch failures, triggering a complete military and industry review of U.S. space launch programs, including software integration and testing processes. Congressional oversight hearings were requested.
- A small town in Illinois received an unusually large monthly electric bill of $7 million in March of 1999. This was about 700 times larger than it's normal bill. It turned out to be due to bugs in new software that had been purchased by the local power company to deal with Y2K software issues.

2. *Why does software have bugs?*

- Miscommunication or no communication - as to specifics of what an application should or shouldn't do (the application's requirements).
- Software complexity - the complexity of current software applications can be difficult to comprehend for anyone without experience in modern-day software development. Windows-type interfaces, client-server and distributed applications, data communications, enormous relational databases, and sheer size of applications have all contributed to the exponential growth in software/system complexity. And the use of object-oriented techniques can complicate instead of simplify a project unless it is well-engineered.
- Programming errors - programmers, like anyone else, can make mistakes.
- Changing requirements - the customer may not understand the effects of changes, or may understand and request them anyway - redesign, rescheduling of engineers, effects on other projects, work already completed that may have to be redone or thrown out, hardware requirements that may be affected, etc. If there are many minor changes or any major changes, known and unknown dependencies among parts of the project are likely to interact and cause problems, and the complexity of keeping track of changes may result in errors. Enthusiasm of engineering staff may be affected. In some fast-changing business environments, continuously modified requirements may be a fact of life. In this case, management must understand the resulting risks, and QA and

test engineers must adapt and plan for continuous extensive testing to keep the inevitable bugs from running out of control

- Time pressures - scheduling of software projects is difficult at best, often requiring a lot of guesswork. When deadlines loom and the crunch comes, mistakes will be made.
- Egos
- Porly documented code - it's tough to maintain and modify code that is badly written or poorly documented; the result is bugs. In many organizations management provides no incentive for programmers to document their code or write clear, understandable code. In fact, it's usually the opposite: they get points mostly for quickly turning out code, and there's job security if nobody else can understand it ('if it was hard to write, it should be hard to read').
- Software development tools - visual tools, class libraries, compilers, scripting tools, etc. often introduce their own bugs or are poorly documented, resulting in added bugs.

3. *Kinds of testing should be considered?*

- Black box testing - not based on any knowledge of internal design or code. Tests are based on requirements and functionality.
- White box testing - based on knowledge of the internal logic of an application's code. Tests are based on coverage of code statements, branches, paths, and conditions.

- Unit testing - the most 'micro' scale of testing; to test particular functions or code modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. Not always easily done unless the application has a well-designed architecture with tight code; may require developing test driver modules or test harnesses.
- Incremental integration testing - continuous testing of an application as new functionality is added; requires that various aspects of an application's functionality be independent enough to work separately before all parts of the program are completed, or that test drivers be developed as needed; done by programmers or by testers.
- Integration testing - testing of combined parts of an application to determine if they function together correctly. The 'parts' can be code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.
- Functional testing - black-box type testing geared to functional requirements of an application; this type of testing should be done by testers. This doesn't mean that the programmers shouldn't check that their code works before releasing it (which of course applies to any stage of testing.)
- System testing - black-box type testing that is based on overall requirements specifications; covers all combined parts of a system.
- End-to-end testing - similar to system testing; the 'macro' end of the test scale; involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network

communications, or interacting with other hardware, applications, or systems if appropriate.

4. *What are 5 common problems in the software development process?*

- Poor requirements - if requirements are unclear, incomplete, too general, or not testable, there will be problems.
- Unrealistic schedule - if too much work is crammed in too little time, problems are inevitable.
- Inadequate testing - no one will know whether or not the program is any good until the customer complains or systems crash.
- Featuritis - requests to pile on new features after development is underway; extremely common.
- Miscommunication - if developers don't know what's needed or customers have erroneous expectations, problems are guaranteed.

5. *What are 5 common solutions in the software development process?*

- Solid requirements - clear, complete, detailed, cohesive, attainable, testable requirements that are agreed to by all players. Use prototypes to help nail down requirements.

- Realistic schedules - allow adequate time for planning, design, testing, bug fixing, re-testing, changes, and documentation; personnel should be able to complete the project without burning out.
- Adequate testing - start testing early on, re-test after fixes or changes, plan for adequate time for testing and bug-fixing.
- Stick to initial requirements as much as possible - be prepared to defend against changes and additions once development has begun, and be prepared to explain consequences. If changes are necessary, they should be adequately reflected in related schedule changes. If possible, use rapid prototyping during the design phase so that customers can see what to expect. This will provide them a higher comfort level with their requirements decisions and minimize changes later on.
- Communication - require walkthroughs and inspections when appropriate; make extensive use of group communication tools - e-mail, groupware, networked bug-tracking tools and change management tools, intranet capabilities, etc.; insure that documentation is available and up-to-date - preferably electronic, not paper; promote teamwork and cooperation; use prototypes early on so that customers' expectations are clarified.

7.0 Introduction

After system testing for the *Bookshop Inventory Management System,* the end product was brought up for system evaluation. After the system had been completed for the first, it had been sent to several end-users to test it. There are many evaluation techniques that use to evaluate the final system. The purpose of the system evaluation is to highlight the strengths, limitations, constraints and possible future enhancement of the developed and completed system. The results gathered from the user testing stage are of a very good means of system evaluation. The following section will explain in detail about the system strengths and its limitations.

7.1 System Strengths

7.1.1 User-friendliness

The graphic interface design of the system was designed to let the users feel comfortable and easy-to-use. The GUI ensured user friendliness. Thus, the users should find it easy to use. The system will return error message if it detect inconsistencies.

7.1.2 Paper-less

Reducing paper use has been the attempt ever since the developments in computerize and much has been mentioned in earlier chapter of this documentation. By reducing paper, it saves long-term cost.

7.1.3 Search Capability

Searching plays an important role in database management. It helps user to store and retrieve data sufficiently and effectively. User could search the books' record, employees' record, publishers' record and customers' order record.

7.1.4 Data Management

In the nature business of a bookshop, it required data edit and update from the database. Sometimes, outdated data should be deleted on order to maintain data integration.

7.2 System Limitations

7.2.1 Order Capability

The system could not order direct from correspond publisher although the customers do an order. The administrator has to manually order from publisher.

7.2.2 Mailing Capability

After the bookshop has purchase the books, which have been order by the customer, administrator must manually contact the customer.

7.3 Future Enhancement

Some functionality of the system can be enhanced in order to improve the quality of the system. The following are the functionality that can be enhanced on this system.

7.3.1 Maintenance of User Interface

The system should provide more useful information and news for the user, like the latest book released, the most popular book and book recommendation. The GUI should be updated for a certain period to give the user a fresh and a greater look.

7.3.2 Support Multiple Language

The current system is only limited to one language. It needs to be enhanced so that it can support more than one language. This is due to different organization is using different type of language.

7.3.3 Membership Concept

The usage membership concept can be implemented in this system to provide built customer loyalty. Besides, the member has some benefits such as discount and new release information.

Conclusion

Inventory investment can be the single most distinguishing characteristic of a successful distributorship or it can be the death knell for a less disciplined company. Given the potential the future offers, it is reasonable to expect that the businessman who master the inventory question will survive and thrive. Those who don't will not have much time to worry about the problem; they will simply disappear.

One would get no argument to the statement that inventory is the single largest asset of a distributorship. Yet, left unmanaged, it quickly becomes the biggest liability. When asked to identify the most common inventory problem, many respond that it is having too many units of the slow moving (dead) inventory. Yet, as we will see in our conversations with a distribution executive and a solution provider, the answer is not always what it appears at first blush. Managing an asset of such significance can be a daunting task if a methodical approach is not employed with structured discipline.

There is likely no more contentious issue facing than inventory. The philosophies and recommended methodologies could well fill a warehouse themselves. Running through it all however is the fundamental premise that advances in technology can effectively be applied to the inventory issue. Whether it is bar coding to capture the movements of stock or the use of three dimensional algorithms that can define the most efficient storage configurations for incoming freight, it is the application of computing

power that enables higher degrees of accuracy, faster information capture and finer levels of analysis.

Technology alone will do little to remedy a critical inventory situation. In the end, inventory is best managed through a combination of appropriate technology, sound business sense, attention to detail, smart buying and total awareness of customer needs. With these skills and resources in place, the distributor buys only what the customer wants, under the most favorable terms, and delivers at the lowest achievable cost.

In working with wholesalers, distributors and business owners for many years, IBM's Pat McNabb has gained a unique perspective of the importance of inventory issues. "Face it, inventory is the distributor's lifeblood. When it flows properly, the entire business organism functions at peak efficiency. When it doesn't, the anemia can kill a business."

References:

1) Achrol, R.S., Reve, T., & Stern, L.W. (1983). The environment of marketing channel dyads: A Framework for Comparative Analysis. Journal of Marketing, 4 7 (4) 55-67.

2) Anderson, J.C. & Narus, J.A. (1984). A model of the distributor's perspective of distributor-manufacturer working relationships. Journal of Marketing, 48 (4), 62-74.

3) Arsham, H. & Shao, S.P., Jr. (1985). Seasonal and cyclic forecasting for the small firm. American Journal of Small Business, 9(4), 46-56.

4) Banks, J. & Heikes, R.G. (1983). A technical aid for EOQ determination. American Journal of Small Business, 7(4), 27-30 .

5) Bhasin, A. & Stern, L.W. (1982). Vertical integration: considerations of efficiency, risk, and strategy. Marketing Channels: Domestic and International Perspectives, 1-5.

6) Bowersox, D.J. (1978). Logistical Management, 2nd edition, New York: McMillan Company.

7) Dollinger, M.i. & Kolchinr M.G. (1986). Purchasing and the small firm. American Journal of Small Business, 11(1), 33-45.

8) Dun & Bradstreet (1981). The Dun & Bradstreet Business Failure Record. New York: Author.

9) Duncan, D.J., Hollander, S.C. & Savitt, R. (1983). Modern Retailing Management, Homewood, Illinois: Richard D. Irwin, Inc.

10) Gaski, J.F. (1984). The theory of power and conflict in channels of distribution. Journal of Marketing, 48 (3) 9-29.

11) Gattorna, J. (1978). Channels of distribution conceptualizations: a state-of-the-art review. European Journal of Marketing, 12(7), 471-512.

12) House, R.J. & Mitchell, T.R. (1974). Path-goal theory of leadership. Journal of Contemporary Business, 3(4), 81-97.

13) House, R.J. & Rizzo, J.R. (1972). Toward the measurement of organizational practices: scale development and validation. Journal of Applied Psychology, 56(5), 388-396.

14) Justis, R. (1981). Managing Your Small Business. Englewood Cliffs, NJ: Prentice-Hall.

15) Khan, M.R. & Rocha, J.R. (1982). Recurring managerial problems in small business. American Journal of Small Business, 7 (1), 50-58.

16) Lambert, D.M. & Bennion, M.L. (1982). New channel strategies for the 1980s. Marketing Channels: Domestic and International Perspectives, 124-128.

17) Lynagh, P.M. & Poist, R.F. (1984). Logistics management: a frontier area for small business, American Journal of Small Business, 8(3), 9-16.

18) Pearson, M. & Monoky, J.F. (1976). The role of conflict and cooperation in channel performance. Proceedings of American Marketing Association, 42, 240-244 .

19) Robinson, R.B., Jr., Logan, J. E., & Salem, M. Y. (1986). Strategic versus optional planning in small retail firms. American Journal of Small Business, 10(1), 7-16.

20) Robinson, R.B., Jr., Salem, M.Y., Logan, J. E., & Pearce, J.A., II (1986). Planning activities related to independent retail firm performance. American Journal of Small Business, 10(3), 19-26.

21) Schul, P.L., Pride, W. M. & Little, T. E. (1983). The impact of channel leadership behavior of intrachannel conflict. Journal of Marketing, 47(3), 21-34.

22) Sheth, J. N. (1983). Emerging trends for the retailing industry. Journal of Retailing, 59(3), 6-18.

23) Wichmann, H. (1983). Accounting and Marketing--key small business problems. American Journal of Small Business, 7(4), 19-26.

24) Hilton, Ronald W., Managerial Accounting. McGraw-Hill,Inc (1994). p.13-14, p. 407,p.218

25) Galloway R, Principles of Operations Management, Routledge

26) Hill T, Production/Operations Management, Prentice Hall, 1991

27) Saunders, Strategic Purchasing and Supply Management, Pitman

28) Slack N, Chambers S, Harland C, Harrison A and Johnston R, Operations Management, Pitman, 1995

29) Anonymous. Successful Small Business Management. Los Angeles Times, January 20 1980

30) Boris Beizer, *Software Testing Techniques*, 2nd ed., New York: Van Nostrand-Reinhold, 1990.

31) Robert V. Binder, "The FREE-flow Graph: Implementation-based Testing of Objects Using State-determined Flows," *Proceedings, 8th Annual Software Quality Week.* Software Research, Inc. San Francisco.

32) Robert V. Binder, "State-based Testing," *Object,* v 5, n 6. July-August 1995.

33) Robert V. Binder, "State-based Testing: Sneak paths and Conditional Transitions," *Object,* v 5, n 8, October 1995.

34) Robert V. Binder, "The FREE Approach for System Testing: use-cases, Threads, and Relations," *Object,* v 6, n 2, February 1996.

35) Grady Booch, *Object-Oriented Design With Applications.* Redwood City, Calif. Benjamin/Cummings Publishing Co, 1991.

36) Tsun S. Chow, "Testing Software Design Modeled by Finite State Machines," *IEEE Transactions on Software Engineering,* v. SE-4, n. 3, 178-86, Jan 1978.

URL:

1) http://www.aednet.com

2) http://www.uow.edu.au/admin/personnel/conditions/management.html

3) http://www.tomax.com/solutions/InvManag.htm

4) http://www.effectiveinventory.com/article3.html

5) http://www.sumtotal.com/nortoncase.htm

6) http://www.collectoronline.com/newsletter/issue5.html

7) http://www.msu.edu/course/prr/473/oldstuff/Inventory%20Management.htm

8) http://www.packagingdigest.com/articles/199805/2.html

9) http://www.collectoronline.com/manage.htm

10) http://www.integratedsolutionsmag.com/articles/tech/index.htm

11) http://www.snapsystems.com/features/mesa%20case%20study.htm#overview

12) http://www.inventorymanagement.com/ccrecac1.htm

13) http://www.zdnet.com/pccomp/stories/all/0,6605,2386421,00.htm

14) http://www.proxim.com/wireless/index.htm

15) http://www.able-consulting.com/system.htm

16) http://nsu-cc.northen.edu/eickhofb/phases.htm#Planning

17) http://nsu-cc.northen.edu/eickhofb/phases.htm#Analysis

18) http://nsu-cc.northen.edu/eickhofb/phases.htm#Design

19) http://nsu-cc.northen.edu/eickhofb/phases.htm#Implementation

20) http://nsu-cc.northen.edu/eickhofb/phases.htm#Support

21) http://nsu-cc.northen.edu/eickhofb/waterfall.htm

22) http://www.softwareatest.com.qatfaq1.index.html

23) http://hissa.nist.gov/HHRFdata/Artifacts/ITLdoc/234/val-proc.html

24) http://www.keylabs.com/

25) http://www.rbsc.com/pages/