

AN ON LINE PRODUCTION PLANNING SYSTEM

by

WILLIAM CHARLES MICHELS

S.B., Massachusetts Institute of Technology  
(1970)

SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February, 1972

Signature of Author \_\_\_\_\_  
Alfred P. Sloan School of Management, January 21, 1972

Certified by \_\_\_\_\_  
Thesis Supervisor, Alfred P. Sloan School of Management

Accepted by \_\_\_\_\_  
Chairman, Departmental Committee on Graduate Students

Certified by \_\_\_\_\_  
Thesis Supervisor, Department of Electrical Engineering

Accepted by \_\_\_\_\_  
Chairman, Departmental Committee on Graduate Students

Archives



AN ON LINE PRODUCTION PLANNING SYSTEM  
by  
WILLIAM CHARLES MICHELS

Submitted to the Alfred P. Sloan School of Management and the Department of Electrical Engineering on January 21, 1972 in partial fulfillment of the requirements for the degrees of Master of Science.

ABSTRACT

Current production planning research emphasizes particular techniques for solving specific problems within large companies. Less emphasis has been placed on integrating these tools economically into small companies.

This thesis describes an integrated production planning system designed to help small custom job shops on an interactive, time-shared basis. The heart of the system is an on-line data base which coordinates the engineering, marketing and production areas of the firm. This complex network type data base consists of 11 multi-chained files. A specialized data management system maintains these files and provides access to them. It permits the user to operate on the data using only knowledge of its logical structure thereby freeing him from any need to understand its internal physical representation.

The primary functions of this On-Line Production Planning (OLPP) system are bill of materials processing, order entry, time phased requirements generation, due date scheduling, finite capacity loading, interactive load leveling, manufacturing order generation, and report generation. Production is scheduled from actual customer orders and not from forecasts of demand. This reflects the environment of many small companies and requires the system to be very flexible. It must be capable of undoing old schedules and making new ones quickly and economically. Loading and scheduling are implemented in a heuristic manner in order to provide low cost efficiency within a volatile environment. When weekly work center capacities are reached in the loading or scheduling process, the system notifies the user of the problem and of possible solutions to it. It then works with him to find an acceptable solution.

THESIS SUPERVISOR: David N. Ness  
TITLE: Associate Professor of Management

THESIS SUPERVISOR: John J. Donovan  
TITLE: Associate Professor of Electrical Engineering

CONTENTS

CHAPTER ONE - A PRODUCTION PLANNING SYSTEM	
1.1 - Introduction to the OLPP System	4
1.2 - Goals of a Master Scheduling System	8
1.3 - Elements of a Production Planning System	11
1.4 - Current Developments in Production Planning	20
1.5 - OLPP System Design Strategy	24
1.6 - User Command Language	27
 CHAPTER TWO - THE OLPP DATA MANAGEMENT SYSTEM	
2.1 - A Centralized Information System	30
2.2 - The OLPP Data Management System	32
2.3 - Preliminary Design of the OLPP Data Structures	35
2.4 - Final Design of the OLPP Data Structures	44
2.5 - Accessing Considerations	55
2.6 - File System Primitives and Utilities	58
2.7 - User File Commands	61
 CHAPTER THREE - LOADING	
3.1 - Introduction to Loading	64
3.2 - Order Entry	68
3.3 - Feasibility Checking	71
3.4 - Load Leveling	76
 CHAPTER FOUR - SCHEDULING	
4.1 - Scheduling Strategy	83
4.2 - Scheduling Algorithm	86
4.3 - Generation of Manufacturing Orders	92
4.4 - Updating	95
 CHAPTER FIVE - SUMMARY	
5.1 - OLPP Capabilities	100
5.2 - Necessary Implementation Considerations	103
5.3 - Potential Additions to the OLPP System	105
 BIBLIOGRAPHY	108

## CHAPTER ONE - A PRODUCTION PLANNING SYSTEM

### 1.1 Introduction to the OLPP System

Production Planning varies in complexity according to the type and size of the company involved. This thesis deals only with the production planning problems of small custom job shops. I chose to concentrate on small companies because their problems have been largely ignored by current research. Larger companies have more 'interesting' problems and have the ability to support their own systems analysis and research efforts. This has quite naturally led to the present state where little modern computer technology has been adapted for use by small companies. Custom job shops have a class of production planning problems which make computerized information systems very attractive to them. They fabricate a large number of possibly complex parts in small lots to meet a demand which varies in terms of design, style and technological requirements. Such variety creates paperwork jungles out of their production planning departments. Continuous manufacturing firms may handle a higher volume of production, but their products are generally standardized and used to maintain inventory levels rather than to meet specific orders. Their larger volume also tends to be more capable of

supporting sophisticated planning systems.

The basic OLPP system is a production planning system which is practical and economical for most small job shops to use. This target user population must be interpreted as a general rather than a specific classification since firms exist in a continuum of sizes and types and not in a small number of discrete and easily distinguishable states. This system can be contracted or extended to increase its suitability for companies which may not exactly fit the 'typical small job shop' mold as defined in this thesis. This flexibility broadens the spectrum of potential system users. A more accurate definition of the potential user population would be all those for whom it would be more economical to use the OLPP system as is, or with slight modifications, than it would be to design, implement and maintain their own system thereby gaining the benefits of more precise tailoring.

Before we can define the capabilities which a production planning system should provide for this type of firm, we must examine the general area of production planning and control. Planning is the process of preparing for the future by estimating the resources that will be needed to fulfill a company's goals. Control is the process of monitoring activities as they occur, comparing them to the pertinent plans and taking any corrective action deemed necessary. The OLPP system is a planning system, but by generating plans it also becomes a subset of a company's entire production

planning and control system.

The area of production planning can be divided into three parts by considering the time horizon and level of detail being reviewed:

- 1) Long Range Planning - yearly plant production
- 2) Medium Range Scheduling- weekly work center production
- 3) Short Term Scheduling - daily man/machine production

These three areas involve a progressively shorter period of time and an increasing level of detail. In various parts of the literature the exact definitions of each may vary slightly and they may be known by a different set of names such as Strategic Planning, Master or Aggregate Scheduling and Job Shop Scheduling.

Medium Range Scheduling has a level of complexity that makes it difficult for small companies to handle and yet is well within the capabilities of an inexpensive computerized system. A master schedule must meet a firm's sales commitments while providing efficient utilization of its resources. The necessary information from which these schedules are developed already exists in most companies and is economically transferable to computer files. Long Range Planning in small companies cannot be formalized as easily since such companies are often new ventures that have little or no history on which to project the future. Even if they are more established, custom demand is potentially unstable and any existing sales forecasts are usually subjective estimates by key personnel. Being small it is often more

economical for them to rely on other sources such as the government or research organizations for views of the future. Short Term Scheduling relies upon the Master Schedule for a global view of the commitments which it must meet. It uses a much more detailed level of data to consider the minute by minute status of every manufacturing order, machine and man. Keeping such schedules accurate and useful requires the rapid discovery and correction of all deviations from the plan. This implies a large amount of data processing and storage. It also may imply a real time or nearly real time data collection system. Both of these may make Job Shop Scheduling a prohibitively expensive venture for small firms.

This thesis will be concerned only with Master Scheduling, since it is an important problem for many small custom manufacturing companies and is a relatively inexpensive first step towards computerized planning and control. I will use the terms master scheduling, aggregate scheduling and production planning to describe the functions of the OLPP system as I have developed them in this section. I will apologize ahead of time for the inevitable conflict this may cause for those who rely on other definitions of these terms.

## 1.2 Goals of a Master Scheduling System

Before we can design a master scheduling system we must examine the functions it should be capable of performing. To help define these we will first examine the various problems that currently arise in manual production planning systems. These trouble spots affect many different areas of the company and all parts of the production planning data base. The following list of problems is sorted by the areas which they primarily affect and should help us define the scope of the OLPP system.

### Marketing

- late deliveries of customer orders
- stockouts of finished products
- inconsistent use of engineering changes
- lack of competitive due dates or prices
- no timely feedback of order status
- no idea of possible or probable delivery dates

### Production Planning

- sales commitments beyond plant capacity
- poor estimates of future workloads
- no timely feedback on order progress
- lack of clear criteria for make/buy decisions

### Production Control

- frantic search for order status
- material stockout or unavailability
- schedules in permanent crisis
- too much expediting necessary
- supervisors too busy firefighting to supervise
- decisions made without proper guidance
- thrashing of setup and move operations
- simultaneous over and under loading
- uneven production rate

### Financial

- high inventory costs
- high production costs
- poor cost breakdown
- poor cash management

In analysing these problems we can discover that the gathering and dissemination of information is in itself a



critical problem for manufacturing companies. Data is often voluminous, scattered and hard to obtain. This creates a problem area which should be analysed in its own right to help rectify the problems of all the other areas.

Information

- high storage costs
- conflicting indexing policies
- erroneous data
- data not available until after it is needed
- different interpretations of same data
- lost information
- redundant or useless data
- no formalized information policies

The goals of a production planning system can be stated in terms of solving these problems. Costs of men, machines, materials and information must be controlled in the face of increasingly complex and voluminous operations. Bills of material must be updated to reflect the latest engineering changes. Competitive quality products must be provided at competitive prices within the time period demanded by the customer. Schedules must be generated which will satisfy varying demand with efficient utilization of the firm's resources. Management must coordinate the individual departments which recognize only their own limited objectives. For example the sales department concentrates primarily on customer service and high levels of sales, the production department about utilization of its own resources and the finance department about costs.

An integrated production planning system must solve all these problems in a manner beneficial to the company as a

whole. Too often companies give priority to only one or two of these areas and must suffer from the neglect of the others. For example marketing oriented firms may promise anything to their customers at the cost of having a chaotic and inefficient production department. Engineering or financially conscious companies may stress their own strengths so much as to lose the ability to compete in the market place. An integrated system will coordinate these various activities and produce a more profitable enterprise.

### 1.3 Elements of a Production Planning System

Companies may schedule production from forecasts of future sales, from customer orders or from a combination of both. In large continuous manufacturing firms, production is often based on sales forecasts. This is the case either when there exists a well defined history of a product's sales or when inventory is required to buffer the production rate from variations in the sales or distribution rate. Job shops are organized on a functional rather than a product basis because they often produce a large number of low volume or custom products. The same potentially random demand factors which predicate a generalized physical layout of a plant may also dictate that production be based on sales rather than forecasts. In small companies, the low volume of sales and the smallness of the market often permit effective forecasts to be made subjectively and informally by top management. Therefore I do not consider forecasting to be an integral part of the production planning process for small custom manufacturing firms. Instead the OLPP system will plan production from actual demand as it occurs.

This section will introduce the common elements of modern production planning systems. It will also discuss the suitability of these tools and techniques for solving the problems of small job shops. Although this will comprise a further definition of the OLPP system, the complete description will not occur until later chapters.

The initial step in producing a product is defining its bill of materials. Products can be either standard, custom or standard with some number of options. When there is a pre-defined bill of materials for a product, it can be designed once and produced innumerable times from the same blueprint. However, each sale of a custom product must include a separate product design. A product must always have a detailed bill of material before any price can be quoted or sale accepted. These bills of material describe the structure of each part used in a product and the specifications for each step in its production. The engineering department creates these records and the marketing, finance and production departments use them for pricing, costing and production. Consequently a bill of materials data base must be the heart of any production planning system. Although the volume and complexity of products varies between companies, this is a potentially large data base.

Inventory control is an area that is closely related to both forecasting and the production rate. Reorder points, levels of service, safety stock and other methods of inventory control of finished products are based on predicted rates of sales during the restocking period. The inherent uncertainty of demand rates for our type of firm rules out the inclusion of formal forecasting models in the OLPP system. It also reduces the usefulness of the various types of inventory

control which are dependent on them. The alternate concept of time phased requirements planning is far more suited for the OLPP environment. This method ties production not to inventory levels, but rather to actual sales. The exact quantity of parts and subassemblies that will be needed in future periods to meet sales commitments is determined through the process of due date scheduling of detailed requirements. End product requirements specify only the quantities and due dates of items which have been ordered by a customer. Detail part and subassembly requirements are calculated by a level by level explosion of an end product's bill of materials. These requirements are assigned due dates, offset backwards from the product's due date, according to the level an item occurs within the product structure and the time necessary to produce the final product from this point. This process is most appropriate where products have multiple levels of assembly, high cost components, long lead times and or multiply used assemblies and parts.

This does not preclude the use of various other inventory control schemes for parts whose use can be more easily predicted and whose inventory costs are low. In fact, the OLPP system recognizes three separate classes of parts. Parts are assigned to these classes by what is commonly called an ABC analysis of part usage versus cost. Class A items, such as airplanes, are low in volume and high in cost, while class C items, such as bolts, are high in volume and low in cost.

Class B items are in the middle of both ranges. Usually most of inventory costs are centered in class A items and most of inventory volume in class C items. The OLPP system concentrates primarily on the critical high cost class A items by generating time phased requirements and weekly schedules for these items. For class B items it keeps track of time phased requirements but does not put them the scheduling process. Class C parts are recognized by the system, but do not rate the high level attention of either time phased requirements planning nor weekly scheduling. In this way the system leaves the inventory control of class B & C parts up to the user. This is desirable since class C items are best suited for the low cost bin tag and safety stock method of control. Other methods of inventory control have been well researched (see ref. #5) and could be easily included either in the OLPP system or in the manual procedures of stock control used by the company for these items.

The system also recognizes the distinction between manufactured and purchased parts. The former is what we have been discussing up to this point and the later fits into the ABC classification scheme but does not require scheduling. Instead purchase requirements are output to the purchasing department early enough to ensure the ordering and delivery of the part by the date the requirement was needed. Most of this thesis is concerned with class A manufactured items, but appropriate mention of the other classes will be made when

necessary.

With time phased requirements planning, the production planning step that follows product specification is order entry. This forms an indirect interface between the production planning system and the customer through the marketing department. It entails entering orders or modifications to them into the central data base and accepting them if they are feasible. Small firms are often so hungry for orders that they do not bother to check if they have the capacity and resources that are required to fill an order. They also avoid feasibility checking because it is often a complicated and time consuming process. Consequently, many delivery dates are missed, customer relations suffer and so do the relations between the production and marketing departments. A centralized system simplifies this task and provides benefits to the whole company.

The next step in the planning process is requirements generation. This is the conversion of the order elements into gross product requirements. It involves the recursive explosion of an end product's requirements through each level of its product structure. Levels correspond to simultaneous assembly of one part from its components. The point at which the final product is completely assembled is referred to as the zero level of the product structure. The components that make up this stage are called level 1 items. The level numbers keep rising until the most elementary component is

described. The total number of levels is dependent on the complexity of the end product. The higher the level number, the earlier the component is required in the manufacturing process of the end product. Explosion is the level by level procedure of multiplying each component usage within an assembly by the requirement for that assembly. These component requirements are then offset from the assembly due date by the lead time by which a component's due date must precede that of its assembly. This lead time represents the average setup, assembly and move times for an average size lot of that particular part. The resulting plan consists of the exact quantities of each detail part, subassembly and end product which are needed to meet sales commitments.

The next step is the reserving of the resources necessary to achieve these specific objectives. Each work center is 'loaded' with the claims on its labor and machine capacity for each time period. Time standards for the manufacturing of each part are kept in the bill of materials data base. Time phased requirements are multiplied by these standards to obtain the load on each work center in each weekly period. This loading can be done in either of two ways. In the infinite capacity loading scheme, work centers are loaded without any consideration of their total capacity. Potential under or over loads are ignored. This is simple to implement and may be later used to focus attention on bottlenecks. Production scheduled in this fashion often has a



low variation in lead time if the shop itself is proficient at leveling the load as it occurs.

In contrast, finite capacity loading monitors the relationship between the load being scheduled and each work center's ability to handle that load. Overloads may influence the user to modify capacity through a commitment to overtime, or to change the due date or size of the order. This decision of what to change is difficult to implement since there are many possible solutions to each overload condition. The OLPP system uses finite capacity loading because it forms the basis for a more realistic and useful production plan. Defining the criteria for alleviating overloads forces a firm to formalize this decision process. This is preferable to allowing overloads to be handled in an ad hoc basis by an overworked and under-informed supervisor.

The final step in production planning is the scheduling of net requirements in a manner that considers the most economic batch quantities for each part. This involves the setting of start and finish dates, and of quantities for all manufacturing orders. Time phased requirements may be split, combined or handled separately, when converted into manufacturing orders. Since all requirements within the same manufacturing order must have the same completion dates, the actual load placed on the work centers may differ slightly from the load estimated from initial requirements. For this reason, the OLPP system uses two different load estimates.

The first is based on initial time phased requirements and is used to determine the general feasibility of a sales order. The second is based on actual manufacturing orders and is used to generate a more realistic load estimate on which to base production schedules. Two must be used because requirements cannot practically be converted into manufacturing orders until all sales have been made for a given period. In other words the scheduling of manufacturing orders is postponed until there is a high probability that the schedule will not have to be revised by a future sales order. The final schedule reflects a balanced and efficient use of the firm's resources, which will meet all promised delivery dates.

More detailed scheduling, such as the assignment of orders to particular men and machines, falls into the realm of job shop scheduling. This will be left up to the supervisors of the individual work centers. Their dispatching should become more efficient when it can rely on a production plan which has considered average lead times, work center capacity and plant load. The value of any schedule depends on how closely it mirrors reality. A weekly production plan has enough leeway in it to absorb some day to day problems. This may not be sufficient since a complex production facility is subject to many random occurrences such as machine breakdowns, material shortages and personnel absences. Sales orders may be cancelled, modified or entered at the last minute. Consequently, any planning algorithm must be flexible enough

to react swiftly and efficiently to a dynamic environment. It must include some feedback mechanism that monitors the performance of its schedules, so that future schedules will continue to be realistic.

#### 1.4 Current Developments in Production Planning

Computers have opened many new opportunities in the area of information processing by stimulating research into techniques that were previously economically unfeasible. In the business world, initial computer applications were aimed at reducing the awesome burden of maintaining voluminous financial records. Lately the impetus has shifted to more operational areas such as production planning. Early Gantt charts and other graphical methods of production planning could not handle complex factory environments. By 1971 a number of computerized integrated production planning systems were operating as effective day to day tools in progressively managed firms. These companies employ techniques which vary from sophisticated optimization models to heuristic algorithms.

These systems are found primarily in larger companies who can afford expensive research and large scale data processing systems. Although some of these systems are in job shops, few are in small ones. Small custom job shops suffer from a more variable demand than do larger ones and therefore their production planning systems must be more flexible and more capable of adjusting rapidly to short term fluctuations in orders.

The most complex scheduling techniques are those which search for a mathematically optimal solution, such as linear programming, dynamic programming or the linear decision rule.

These methods are restricted to the solution of very specific problems. The ones which are common today are based on a known or forecasted level of demand. To achieve optimality they must maximize or minimize some objective function. The validity and usefulness of these models depends on the appropriateness of both the assumptions and the objective function. Because of their complexity these models are costly to develop and to run. Either the high cost or the assumption of known demand is sufficient to make these models inappropriate for small custom job shops.

Simulation models are also widely used for scheduling. They are mathematical models which attempt to duplicate a system's response to various inputs. Simulation models are less complex than optimizing ones and consequently they cost less. They allow management to test out the implications of alternative strategies. Simulation is widely used for job shop scheduling in large companies who can afford the necessary factory data collection system. The major drawback with using regular simulation models for master scheduling is that modifications to orders require the entire model to be rerun. This is a viable technique only when changes are infrequent or where frequent model runs can be afforded.

The least complex scheduling technique is that of heuristic scheduling. A heuristic is basically a nonproven aid to decision making or, more colloquially, a rule of thumb. While heuristics do not necessarily lead to optimal solutions,

experience over time has proven their general usefulness in rapidly finding good solutions to recurring problems with a minimum of effort. This may be preferred to finding an optimal one after a long and costly search. The heuristic approach also attempts to improve a recurring human decision process by formalizing it and applying it consistently. Heuristics are logical processes but not necessarily rigorous mathematical ones. They can even be simple and relatively inexpensive. Their value depends on the premise that the benefits of a more precise solution do not merit the extra costs of finding it. This is generally true in the production planning area of small companies who can not absorb the overhead of more complex solutions. This simplicity is doubly important in 'make to order' firms because of its implied relative reversibility and flexibility.

The heuristic approach is therefore the most appropriate scheduling technique for companies who have been previously unable to afford computerized production planning. The order entry and requirements generation phases in most systems today are pretty standard. Several firms have implemented heuristic loading or scheduling phases within integrated production planning systems, but the OLPP system is probably unique in that it is also interactive and time-shared. The interactive capability is a means of creating a man-machine decision system where the experience of the human compliments the high speed data processing capability of the machine. Availability

on a commercial time-sharing system can free a small company from footing the total software development and hardware maintainance costs of the system. Interactive production planning systems are rare and primarily exist within companies who are big enough to afford their own multiprogramming system. I do not know of any time-shared production planning systems, however some commercial time-sharing companies do offer interactive bill of materials processors.

### 1.5 OLPP System Design Strategy

The primary design goal of the OLPP system is to solve the production planning problems of small custom job shops. The system is an application of current computer technology which aids this particular class of small firms who have not previously had the resources necessary to obtain the benefits of this technology. These missing resources are both human and financial. The OLPP system reduces the resource limitations of computerized production planning in two ways. First, by being an externally developed package, it permits a company, which does not have or can not afford technical specialists, to utilize sophisticated software. Second, by being available on a time-shared basis, it permits firms to pay only for the hardware resources which they use. This may be a level of expenditure which would not support the purchase or leasing of high performance hardware.

The system's first order of responsibility is to solving the problems of the ultimate user in a manner which is optimal for him. Business users are concerned with the following criteria:

- 1) Is OLPP more efficient than a manual system?
- 2) Is it based on standard production procedures?
- 3) Does it cost less in time and money than a self developed package?
- 4) Does it have flexible I/O, files and programs?

As has been mentioned previously, efficiency is obtained by integrating a company's information system and by drawing on the vast data processing power of the computer. The standard



procedures which the system provides are information retrieval and master scheduling. The information retrieval facility is geared to providing the data desired by a user in the form which he desires it. The master scheduling algorithm is heuristic in nature and is based on the accepted techniques of time phased requirements planning, finite capacity loading and due date scheduling. Cost involves not only the hardware and software economies of scale which were brought out in the previous paragraph, but also every level of the system design and implementation process. This includes the definition of the functions to be provided, the designing of flexible and efficient programs and files, and the use of goal oriented heuristics. A particularly important consideration is the tradeoff between the use of CPU time and the use of on-line storage. At stake are various additions to the file system which would take up space but reduce processing time. The relatively high cost of CPU time when compared to the relatively low cost of on-line storage in time shared environments has caused the OLPP system to freely substitute storage space for program complexity.

In order to facilitate both its initial development and its future evolution, the OLPP system was designed in relatively autonomous modules. In particular, three modular interfaces were set up:

- 1) User/System interface - (User Command Language)
- 2) Processing programs/File system interface -  
(File System Utilities)

### 3) File system/Operating system interface - (File System Primitives)

These interfaces allow easy adaptation of the system to fit different environments or to perform different functions. Each of them will be discussed later in separate sections.

The initial OLPP system was developed on the MULTICS system at MIT and not on a commercial time-sharing system. MULTICS contains several features which were very useful in system's development. Most importantly, it supported PL/1 which is a powerful and relatively machine independent higher level language with an excellent repertoire of data types and structures. MULTICS also provides valuable text editing and debugging facilities.

In addition to these features, MULTICS is a good example of the type of environment which a commercial time-sharing system would have to provide to actual OLPP users. Some of its desirable attributes are:

- 1) Users share the benefits and costs of a powerful hardware system.
- 2) Turnaround time is minimized and the user receives nearly immediate interactive service.
- 3) The user is constantly aware of the progress of his job and can terminate it whenever he desires to.
- 4) Files and programs are shareable and yet protected against accidental or malicious damage through the MULTICS back-up and protection facilities.
- 5) The system is up most of the time. This should actually approach all of the time in order to be relied on by a commercial enterprise.

## 1.6 User Command Language

The major considerations in the design of the OLPP user command language are the power of the commands and their ease of use. The rationale behind the function of each command will be explained later when the commands themselves are presented. However the ease of use of the command language is a general criterion which applies equally to all commands. The way in which users communicate with the system must be simple enough to allow non-programmers to take advantage of computer technology. There are four ways in which OLPP achieves this goal:

- 1) The physical medium is easy to use. Most users are already familiar with the conventional typewriter and thus can adjust easily to a console.
- 2) The way in which a terminal user defines his work is uncomplicated. He enters commands which resemble English words to describe the function he wishes to accomplish.
- 3) If the user is unfamiliar with the system he can type '?' or '<command name> ?', which will list either all the legal OLPP commands or all the legal formats of a specific command. In some cases he need not provide detailed parameters since the system can either provide default values or ask for the missing critical details.
- 4) The system always keeps the user aware of what is happening, so that he knows what to do next. He converses with it on a step-by-step basis. The system lets him know when it is ready to accept OLPP command level input by printing out 'INPUT'. It prints out diagnostics when errors or important events occur. He even has two modes of receiving this information. The default verbose mode uses uncomplicated language to explain events. When the messages become familiar to him, he may issue the 'set brief\_mode' command which tells the user to use an abbreviated and faster form for its messages. At any subsequent time he may issue the 'set verbose\_mode' command to restore the educational and diagnostic messages to their full form.

These system characteristics help new users become accustomed to the system, while at the same time providing a powerful and flexible environment for more experienced users.

The OLPP system is designed to be invoked by any authorized user who issues the 'olpp' command from the command level of a commercial time-sharing system . This routine opens the file system and becomes the command level environment for the OLPP system. It parses the first word of all command lines issued to it and compares this command name to a table of legal OLPP commands. The '?' command occurs on this list and serves as a means of accessing all the currently implemented OLPP commands. Other legal command names cause the proper processing routine to be invoked. These procedures parse the rest of the command line and carry out any other input or output which may be associated with the command. Each of these commands also has a table of its legal options which includes a '?'. This option causes the listing of each legal form of the command. When finished they return to OLPP which types out 'INPUT' and waits for another command line.

Most commands have several basic formats. Each of these exists to serve a specific function more efficiently than would a single multi-purpose format. Often there are single line and multiple line formats of a command. This is to permit fast and efficient input in either high volume or low. Arguments are allowed to be either implicitly defined by their position on the command line or explicitly by their pairing with a parameter name. In this way data can be entered either into an entire record or just selected portions of it. Furthermore all command names and options can be truncated to any length which preserves

their uniqueness. This allows experienced users to type less and enjoy faster response from the system.

## CHAPTER TWO - THE OLPP DATA MANAGEMENT SYSTEM

### 2.1 A Centralized Information System

Production Planning Systems are based on large volumes of data which are often kept separately by different departments. For example there are engineering blueprints, sales orders, production plans and cost reports. These are highly interdependent and often redundant and or inconsistent. The heart of the OLPP system will be a centralized information system which collects the entire production planning data base into a single internally consistent unit. It will be stored within a computerized file system and made available to the departments which must maintain or access it. Since the information will be easier to access when it is stored in one place, it will be also be easier to group it into reports of any desired level of detail.

Consolidation of files from several departments into a common data base is not a trivial process, however it is a necessary and profitable one. Data which is redundant or out of date may be discovered and and discarded. Inconsistencies and inaccuracies may also be found and corrected. After consolidation only one set of archives must be updated. This greatly reduces both the cost of maintainance and the

opportunity for error.

A centralized production planning data base coordinates the various functions within a manufacturing company. This integration may be more beneficial to the company, than the efficiency added to the other parts of the production planning system. New information is made available to all departments simultaneously whenever it becomes known to any one of them. In addition the large potential storage capacity of the computer allows intuitive approximations of factors such as available capacity to be replaced with more detailed and accurate estimates. This availability of more timely and accurate information improves the quality of decision making at all levels of the firm. The increased capacity for information also allows higher volumes of business to be handled more efficiently than is presently possible. These computer based benefits must not overshadow the fact of life that the value of information to any system depends on its initial accuracy. Although the chance of error is lowered with the reduction of manual transcription of data, it is not eliminated. This reliance of a system on its inputs is commonly referred to as the 'Garbage In-Garbage Out' (GIGO) phenomenon. If work center capacity or product lead time can not be estimated correctly, then the schedules based on them are not likely to be any more accurate or useful.

## 2.2 The OLPP Data Management System

The OLPP system data base exists within a specialized data management system, which consists of three parts:

1) File definitions and contents - The OLPP file system contains a company's entire production planning data base. This includes both information which the user inputs and that which the system creates from these external inputs. Since this is a dedicated data management system, the file definitions are an integral part of the system and not an externally variable one. The user does not have to go through the complicated process of designing his own record formats. Instead OLPP includes a comprehensive collection of files which meets the needs of most small custom job shops. Any system tailoring can be done either by the OLPP system programmers or by a user employed computer specialist. The system is designed so that the user does not have to incur this overhead unless he chooses to do so.

2) Accessing routines - In order to facilitate the evolution of the OLPP system, two interfaces were constructed within the file system. The first of these is the interface between the file system and the operating system. These primitive file commands localize machine dependence to the inner core of the file system. The second is the interface between the file system and the processing programs. These file system utility routines centralize the code for common file system functions and consequently localize the reliance



on certain file system conventions. These utilities perform basic operations such as chain searching or chain maintainance. Generalized commands and standard file formats were chosen to reduce the complexity of the file system. This additional clarity helps both system designers and users who want to learn the system. Since these file system interfaces are the most heavily used routines in the system, speed and efficiency of execution were critical design factors.

3) User file commands - This portion of the user command language forms the interface between the user and the file system. It allows him to manipulate the files without regard to their internal physical structure. All he must know is the logical interrelationships among the data. The commands give the user the ability to create, modify, delete or output any record of any file. They are designed for simplicity, efficiency of operation and usefulness to the user.

This file system will be on-line to the production, marketing and engineering areas of the company. Each has instantaneous access to any files which they have the right to maintain or query. This does not imply that the data base must be maintained on a real time basis. Each department may enjoy immediate access to the data but it is not as critical for the data to be updated in real time. Master scheduling deals with orders in future weeks. As we will see in later chapters, the usefulness of this generalized level of detail is rarely degraded by updates which occur as frequently

as hourly or several times a day. In a more detailed arena such as short term scheduling this is not necessarily true.

## 2.3 Preliminary Design of the OLPP Data Structures

In designing a file system, four basic levels of detail must be considered:

1) FIELD - elementary data item such as part number, cost or lead time.

2) RECORD - set of potentially dissimilar fields (attributes) which pertain to some single object such as part number, cost of part and lead time necessary to produce part.

3) FILE - collection of similar records. Files may represent either basic units such a part file with one record per part or relationships such as a Product Structure file with one record per component/assembly pair.

4) FILE SYSTEM - collection of all files within a particular data base.

To properly design a file system, we must examine both the natural structure of the data and the environment within which it will exist. The structure of a file system is the grouping of fields into records and records into files. This structure is most effective when it mirrors the natural relationships within a company's data base. The environment of a file system consists of the type and frequency of the queries to which it will be subjected. These characteristics determine the accessing methods which will be most efficient for each file. This section will develop the preliminary design of the OLPP data structures. Later sections will describe the final design of the file system and the accessing methods which will be used with it.

Our starting point for the design of the OLPP data structures will be a global view of the production planning data base:

Per Manufacturing Order:

- manufacturing order number
- quantity, part number
- start date, finish date

Per Work Center:

- work center number, description
- capacity (regular and maximum) per period
- load (estimated and scheduled) per period

Per Customer:

- customer number
- name, address, phone number
- orders from this customer

Per Sales Order:

- sales order number
- customer information
- due date
- parts, quantities and costs

Per Part:

- part number, description
- cost, lead time etc.
- class A/B/C
- type buy/make
- order strategy
- time phased requirements
- components (for assemblies)
- routing (operations, work centers, time standards)

We will now examine these groups more carefully in order to determine their natural inter-relationships. This will help us find the best representation for them.

The most simple of these groups is the manufacturing order one. Its basic unit is the manufacturing order, each of which has a similar set of attributes. So we will create a manufacturing order file with records in a one-to-one correspondence with manufacturing orders. This is a single level file of fixed length records. It will be initially implemented as a sequential list of orders. Except for the first and last records, each one will have exactly one predecessor and one successor. The following diagram shows

the format of a record in the Manufacturing Order file.

Field Names	MANUFACTURING ORDER #	PART #	QUANTITY	START DATE	FINISH DATE
----------------	--------------------------	--------	----------	---------------	----------------

The work center data has very similar characteristics. This data is a set of descriptions of each of the plant's work centers. Although the load per period and capacity per period attributes involve many data items, they can be represented by a standard number of fixed length fields. Since this is another very simple collection of data, we will again use our most simple representation for it. We will define a sequential Work Center file, each of whose fixed length records will have the following format:

Field Names	WORK CENTER #	DESCRIPTION	CAPACITY (26)	LOAD (26)
----------------	------------------	-------------	---------------	-----------

The next group which we will examine is the sales data. Some sample orders will help us illustrate the structure of the data in this area:

Cust Name: ACME MFG.		
Address: - - - - -		
ORDER # 101		
ORDER DATE: 4/4		
DUE DATE: 7/10		
QUANTITY	PART #	COST
23	XYZ	50.00
10	HA63	100.00
		<u>150.00</u>

Cust Name: ACME MFG.		
Address: - - - - -		
ORDER # 102		
ORDER DATE: 11/12		
DUE DATE: 12/19		
QUANTITY	PART #	COST
18	243	60.00
		<u>60.00</u>

Cust Name: SIMPLEX		
Address: - - - - -		
ORDER # 810		
ORDER DATE: 1/21		
DUE DATE: 2/14		
QUANTITY	PART #	COST
2	1847	48.00
1	1475	13.00
3	H643	63.00
		<u>124.00</u>

Here we have three separate types of data, each with their own attributes:

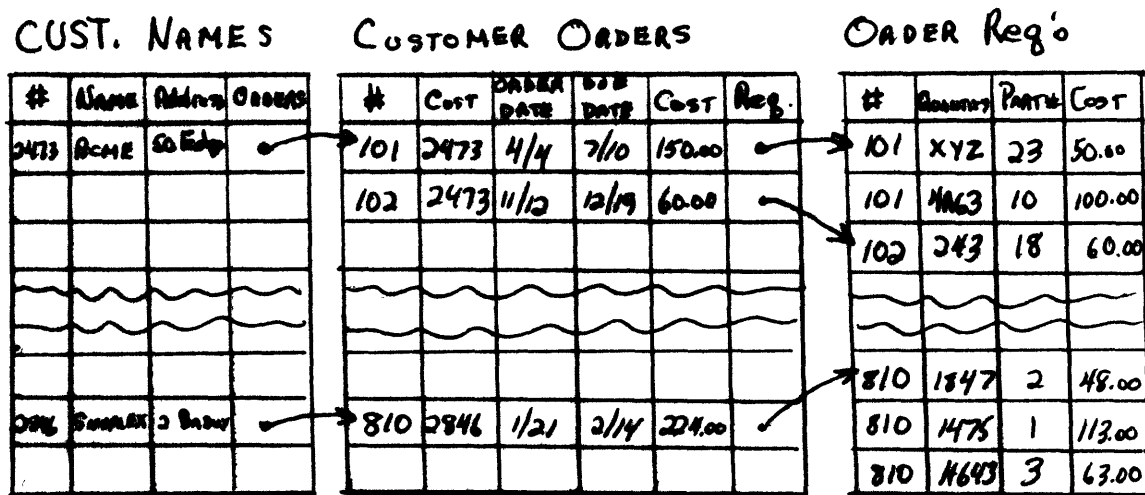
- 1) A variable number of customers
- 2) A variable number of orders per customer
- 3) A variable number of items per order

By examining order# 101, we can see that it is a partner in the following relationships:

- 1) It is a child in the parent-child relationship between the Acme Mfg Co. and its orders.
- 2) It is a brother in the brother-sister relationship between order 101 and order 102.
- 3) It is the parent in the parent-child relationship between the order and its component items.

Consequently, this is a three level tree where orders have one parent, a variable number of brothers and a variable number of children. We can either represent this tree with a complex system of variable length records or by three types of fixed length records, one for each level of the tree. Variable length records tend to save space at the expense of program complexity and processing time. Since time-sharing systems usually have large amounts of relatively inexpensive on-line storage, OLPP users would probably have to pay more for processing variable length records than for storing fixed length ones. Given that OLPP uses only fixed length records, we still have two alternative methods for representing this data. We can either insert the child records in between the parental ones in header-trailer fashion or use pointers from contiguous parental records to contiguous lists within separate children files. Once again the second method is

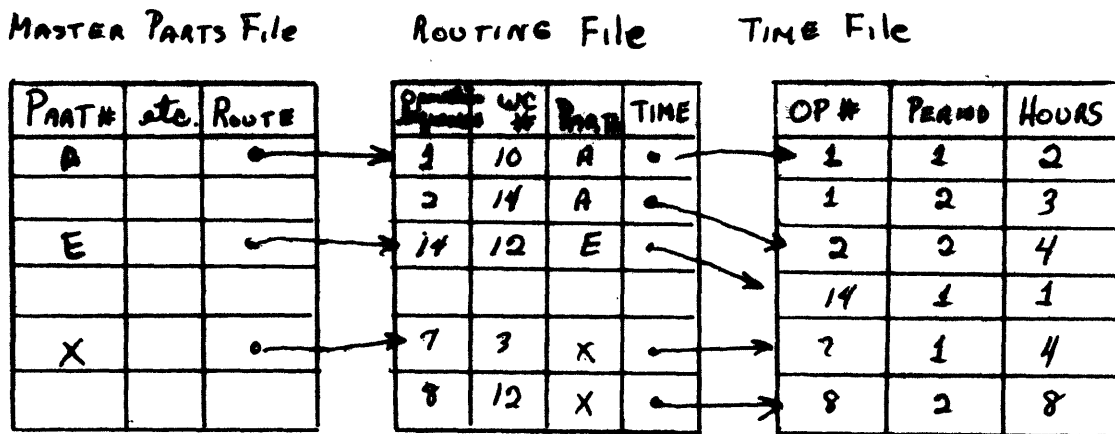
simplier and less costly in terms of processing time, so OLPP has separate customer name, customer order and order requirements files. Brothers and sisters are physically contiguous, while parents contain pointers to their first child. Each of these files represents one level of the tree. The next diagram shows how the records in these three files represent the three sample orders.



The final and most complex portion of the data base is that pertaining to all the parts used in the manufacturing process. It is clear that we need at least a Master Parts file to hold the description of each part, its total lead time, its class, its cost etc. However, the related time phased requirements, routing and product structure information adds a great deal of complexity to this one area. The time phased requirements are the gross product requirements for each time period in the planning horizon. Since this number of periods is constant these requirements can be contained in

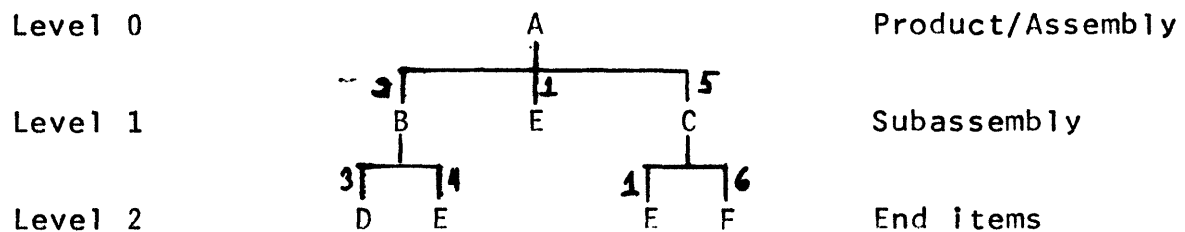
a fixed number of fixed length fields within the Master Parts record.

The routing data is the sequence of operations which a part must go through while being manufactured. Each operation has associated with it a work center and the total number of hours per period which this operation will take in the manufacturing of this part. This data has a structure similar to that of the sales data, in that it forms a three level multi-branched tree. Therefore OLPP will handle both in a similar fashion. The Master Parts record for each manufactured part will have a pointer to the list of its operations in the Routing file. Each Routing record will, in turn, contain a pointer to the time phased time requirements of this operation within the Time file. The following diagram shows the relationships between the records in the Master Parts, Routing and Time files.

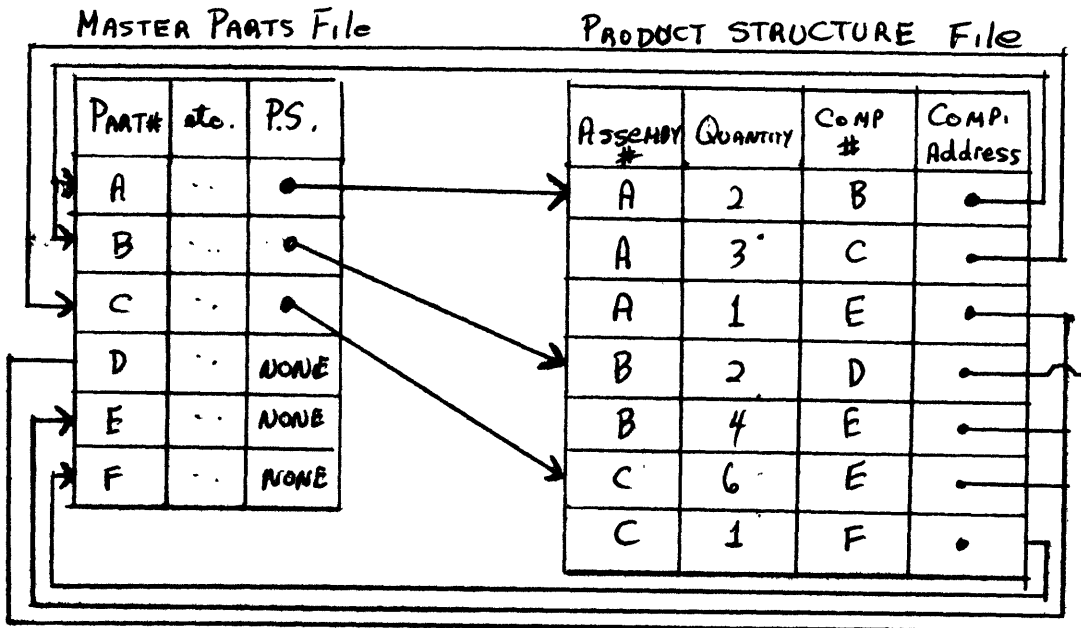




The product structure data describes all the components of each level of assembly of every part. This data is more complicated than any we have handled so far. Our previously most complex structures were the three level sales and routing hierarchies. The next diagram represents a typical bill of materials.



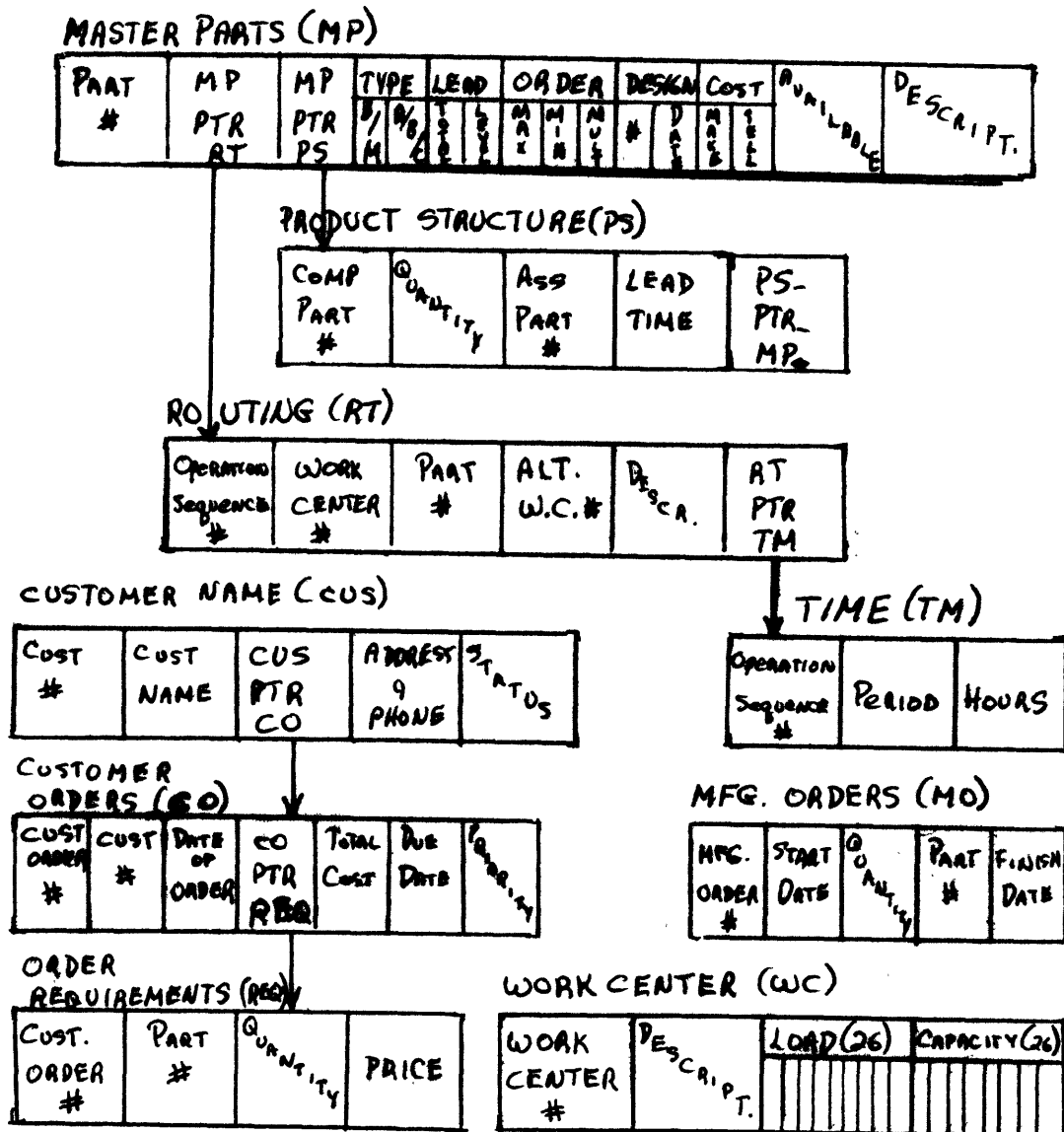
From this diagram we can see that a part can either have multiple parents (E) or multiple children (A,B,C). This makes the bill of materials data a network structure. It can be represented by a minimum of two highly inter-dependent files. The first is the basic description file (Master Parts file) and the second is a relationship file (Product Structure file). Each Product Structure record contains one component/assembly pair. Each Master Part record contains a pointer to its first component. The generality of a network structure is made possible by pointers within the Product Structure records which point back to the Master Parts record for the component part. The diagram on the next page shows the Master Parts and Product Structure records which would be needed to represent the previous sample product structure.



In order to make it impossible for an infinite loop to occur within a product structure, a part (and its components) may never include itself or one of its parents as a component. To help the file system avoid this problem, each master parts record will contain a low level code, which indicates the lowest level at which this part is ever used as a component. Whenever it generates a Product Structure, the file system checks that all assemblies have a larger low level code than any of their components. It also raises the low level code of a part whenever it is used at a lower level than it previously was. When it can set low level codes in a consistent manner within a product structure, because of a loop, it rejects it. The previous diagram includes the proper low level codes for the sample structure.

The preliminary file system which we have developed up to

this point is summarized in the next diagram. In the next section we will develop a more detailed version of it by examining the various uses of the data. In the rest of this thesis all files, records and fields will be referred to by the abbreviations listed in this diagram and the one following section 2.5. Field names will rely on PL/I structure, subscript and underscore notation.



## 2.4 Final Design of the OLPP Data Structure

Up to this point the sole criterion for structuring the OLLP files has been the 'mirroring of reality'. This is not sufficient since it does not take into account the potential uses of the data. In particular, we have ignored three basic problems:

1) How are individual records located within a file?

Most computer files are sorted according to the value of a particular field, known as the primary key. Sorting gains us two important performance characteristics:

A) In a sorted file a record can usually be retrieved or deemed absent more rapidly than in an unsorted file. In unsorted files, records are found by an exhaustive linear search, whose performance deteriorates directly with file size. The processing speed and efficiency we can gain by sorting is critical to the OLPP system because it lowers cost and improves the interactive performance of the system.

B) Any reports generated by the system will be used by people and people usually require information to be sorted before they can make effective use of it.

The OLPP system is designed to operate in a dynamic environment. Therefore its performance should not be degraded by changes to the file contents. If the files are to remain sequentially organized and sorted, they would have to be resorted after the addition or deletion of any record. Resorting is a time consuming and costly operation.

2) Although our preliminary design includes most of the information which OLPP will require, it does not take into consideration the possibility of wanting to access a record by more than one key. In particular a complete system

should include:

A) Accessing the Product Structure file by component part number within a list of components of an assembly and by assembly part number within a list of assemblies using this component.

B) Accessing the Routing file by an operation sequence number within the routing list for a particular part and by part number within a list of parts manufactured in a particular work center.

C) Accessing the Customer Order or Manufacturing Order file by customer order number or manufacturing order number to respond to inquiries, and accessing the Customer Order file by due date within a list of a customer's orders, and accessing the manufacturing order file by due date within a list of manufacturing orders for a part.

D) Accessing the Customer Name file by customer name or by customer number depending on which is known at a given time.

The first two of these are called where-used lists and are used to respond to actual or possible changes of work center or part status. All four cases involve organizing a file by the value of more than one key. Sequential organization has only one dimension and thus can handle multi-dimensional files only by making a separate copy of the file or resorting it by each potential key.

3) Since speed is critical to OLPP, inter-file references by key values may not be acceptable. For example, the use of the customer number field in the Customer Order file, would require the searching of the Customer Name file before the information within the desired record could be used. This puts a relatively high cost on inter-file references.

4) The OLPP system is destined for a dynamic environment, so any decisions which it makes must be

completely reversible. In particular, work center loads, manufacturing orders, and pegged requirements must be capable of being disassembled and reconfigured in the light of new information. This is not possible in the preliminary design since there is no record kept of the components of these various gross figures.

We will now improve the design of the OLPP file system by correcting these problems. For reference, a complete color coded diagram of the final file system is located on the last page of this section.


The first change is the abandonment of sequential file organization. Although simple, it is not suitable for a volatile environment. Instead, OLPP employs list processing techniques on chained files. Chaining is a method of using pointers to create a logical list or 'chain' of records by having all list elements point to their successors. This allows the logical list to be freed from the physical constraints of the storage medium. The problem of resorting an entire file is eliminated by the ability to patch records into or out of chains without having to move any records. Heads of chains are indicated by anchor pointers and ends of chains are indicated by a special (NIL) pointer in the successor field of the last record.

To further improve processing efficiency OLPP uses bi-directional chaining. This requires each record in a chain to contain two pointers, one (PRV) pointing to its predecessor

and another (NXT) pointing to its successor. From this extra pointer we gain the ability to traverse the chain in either a forward or a backward direction. In addition all chain maintenance can be controlled from the contents of the record currently being processed and broken chains can be more easily reconstructed. Bi-directional chaining is a tradeoff between the storage cost of one field per chain and the time necessary to process uni-directional chains. Assuming once again that the marginal cost of storage is less than the marginal cost of CPU time, OLPP has adopted bi-directional chaining.

In the file system diagrams, chains are represented by a three field subrecord containing:

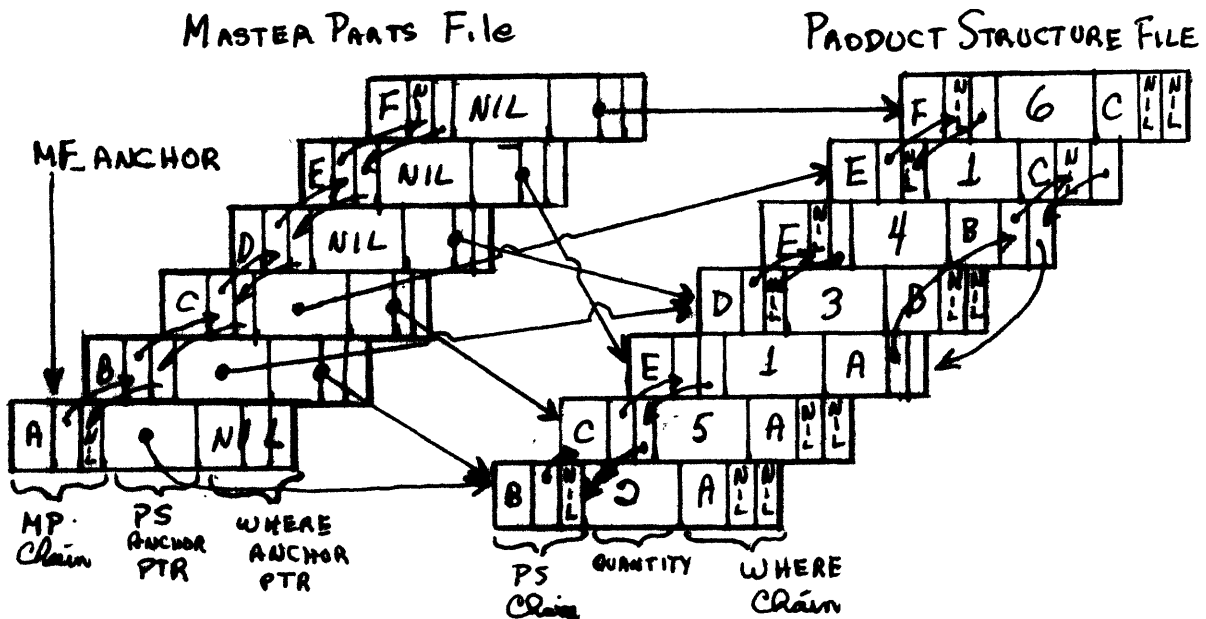
Name Of Chain		
Key	NXT	PRV
Name		



In colored diagrams the entire subrecord and the arrows indicating pointers are in blue. This subrecord forms a standard header on every record in every file.

The use of chaining also removes the one dimensional restriction from the file structure because it separates physical contiguosity from logical connectivity. Any number of chains can be threaded through a record as long as each chain has its forward and backward pointers and a key to determine its position within the chain. Each chain allows the record to have a unique position in a sorted list which is independent of its position in any other chained list. This

would not be possible in a sequential organization unless the file was duplicated and sorted for each key. Multi-chaining allows complex relationships to be represented in a deceptively simple fashion. The following diagram will show the product structure from section 2.3 in a multi-chained format.



One feature of this diagram which is not in our previous version of the file system is the inclusion of anchor pointers to the heads of the various chains. Some anchor pointers such as MP.PS\_anchor\_ptr and MP.WHERE\_anchor\_ptr fit directly into one of the existing files. However others such as MP\_anchor\_ptr must be directly accessible by the file system and not embedded within a record of some other file. To handle this problem, OLPP has a single record control (CTL)



file which holds the various anchor pointers required by the system. Anchor pointers are named <source file>.<chain name>\_anchor\_pointer and are green in the colored diagrams.

The following list reviews all the OLPP chains, keys and anchor pointers:

<u>FILE</u>	<u>CHAIN</u>	<u>KEY</u>	<u>ANCHOR</u>
PS	PS	Component_Part_#	MP.PS_anchor_ptr
	WHERE	Assembly_Part_#	MP.WHERE_anchor_ptr
RT	RT	Operation_Seq_#	MP.RT_anchor_ptr
	WHERE	Part_#	WC.WHERE_anchor_ptr
CUS	CUS#	Customer_#	CTL.CUS#_anchor_ptr
	CNAME	Customer_Name	CTL.CNAME_anchor_ptr
CO	CO#	Customer_Order_#	CTL.CO#_anchor_ptr
	CO	Due_Date	CUS.CO_anchor_ptr
REQ	OR	Part_#	CO.OR_anchor_ptr
MO	MO#	Mfg_Order_#	CTL.MO#_anchor_ptr
	MO	Due_Date	MP.MO_anchor_ptr
MP	MP	Part_#	CTL.MP_anchor_ptr
TM	TM	Period_#	RT.TIME(n).TM_anchor_ptr

These chains are maintained in a sorted order by the file system. Deleted records are not returned to the operating system but rather added to a chain of free records for the particular file. All free chains are anchored in the CTL file. This method of keeping unused records available minimizes the number of exchanges (ALLOCATE and FREE statements) between the OLPP system and the operating system, thus saving overhead time and expense.

Inter-file references are speeded up by providing pointers in place of key values within records. These so called home\_pointers are the fastest way of accessing another record. Key values are maintained only for output, when the cost of an extra field in a record is less than the cost of

accessing another record just to get the value of a single field. These pointers are referred to as <source file>.<qualifier>\_home\_<parent file>\_ptr and are pink in the colored diagram. The qualifier is an optional description which is used to make the name more meaningful. The following home pointers will be added to the file system:

<u>FILE</u>	<u>HOME POINTER</u>	<u>REASON</u>
PS	PS.Comp_home_MP_ptr	Speeds explosion
	PS.Ass_home_MP_ptr	Speeds WC to MP traverses
RT	RT.home_WC_ptr	Speeds MP to WC traverses
CO	CO.home_CUS_ptr	Speeds CO to CUS traverses
REQ	REQ.home_CO_ptr	Speeds REQ to CO traverses
MO	MO.home_MP_ptr	Speeds MO to MP traverses

The problem of reversibility of decisions is tied intimately to the REQ file because this file is the nucleus of the entire OLPP decision process. When orders enter the system they cause a CO record and a variable number of REQ records to be created. They also cause the immediate generation of time phased requirements and estimated work center loads, and the eventual generation of manufacturing orders, actual work center loads and weekly production schedules. These time phased requirements, loads and manufacturing orders are for parts in every level of the product structure of the items which have been ordered. Reversibility implies connecting loads to the manufacturing orders which caused them, manufacturing orders to the specific requirements which caused them and every time phased requirement to the order requirement which caused it.

The solution of this complex problem involves the introduction of the new concept of a pegged requirement. A time phased requirement is a part#/quantity pair in a particular time period. A pegged requirement is a part#/quantity pair which is tied both to a particular time period and to a particular order requirement. A time phased requirement can be thought of as an aggregation of pegged requirements. To make the system completely flexible we must switch to the more powerful and detailed method of recording pegged requirements. This removes the obstacle of irreversibility which is inherent in an aggregated representation, by establishing a vehicle for various linkages.

A possible way of representing pegged requirements would be to replace MP.Time\_Phased\_REQ(n) with MP.PR\_anchor\_ptr(n), where PR would be a chain of records in a separate Pegged Requirements(PR) file. An alternative scheme would be to combine the REQ and the PR files into one file with two chains running through it. These can be combined because the current REQ file is essentially a subset of the new PR file. We might also desire to expand the OR chain to contain all the requirements caused by an order, so that we can efficiently cancel or modify an order. This combined scheme centralizes the requirements information and eliminates the need for both two separate files and any references between them. Consequently the OLPP system has a single REQ file with PR and

OR chains running through it. This structure still does not allow a pegged requirement to be traced back to its order requirement, or even to the pegged requirement immediately superior to it in the product structure. To rectify this omission end use (REQ.End\_home\_REQ\_ptr) and Immediate use (REQ.Imed\_home\_REQ\_ptr) pointers are included in the REQ record.

This still leaves us with the problem of reversing the loading of work centers and the scheduling of manufacturing orders. To accomplish this the work center and manufacturing data bases must include references back to specific REQ records, which contain home\_CO\_ptr. Manufacturing orders can easily be linked back to the proper requirement records by the creation of a third chain through the REQ file. All requirements contributing to a single manufacturing order are connected by this Manufacturing Requirements (MR) chain which is ordered by RFQ.Imed\_home\_REQ\_ptr and anchored by MO.MR\_anchor\_ptr. The key for this last chain was chosen more for its uniqueness within a single manufacturing order than for any significant meaning.

Linking individual load components back to the requirements which caused them is a slightly more complex problem, since a given requirement may effect several work centers in several periods. This can not be handled just by threading another chain through the REQ file. Instead, what is needed is a separate file called the LOAD (LD) file, with

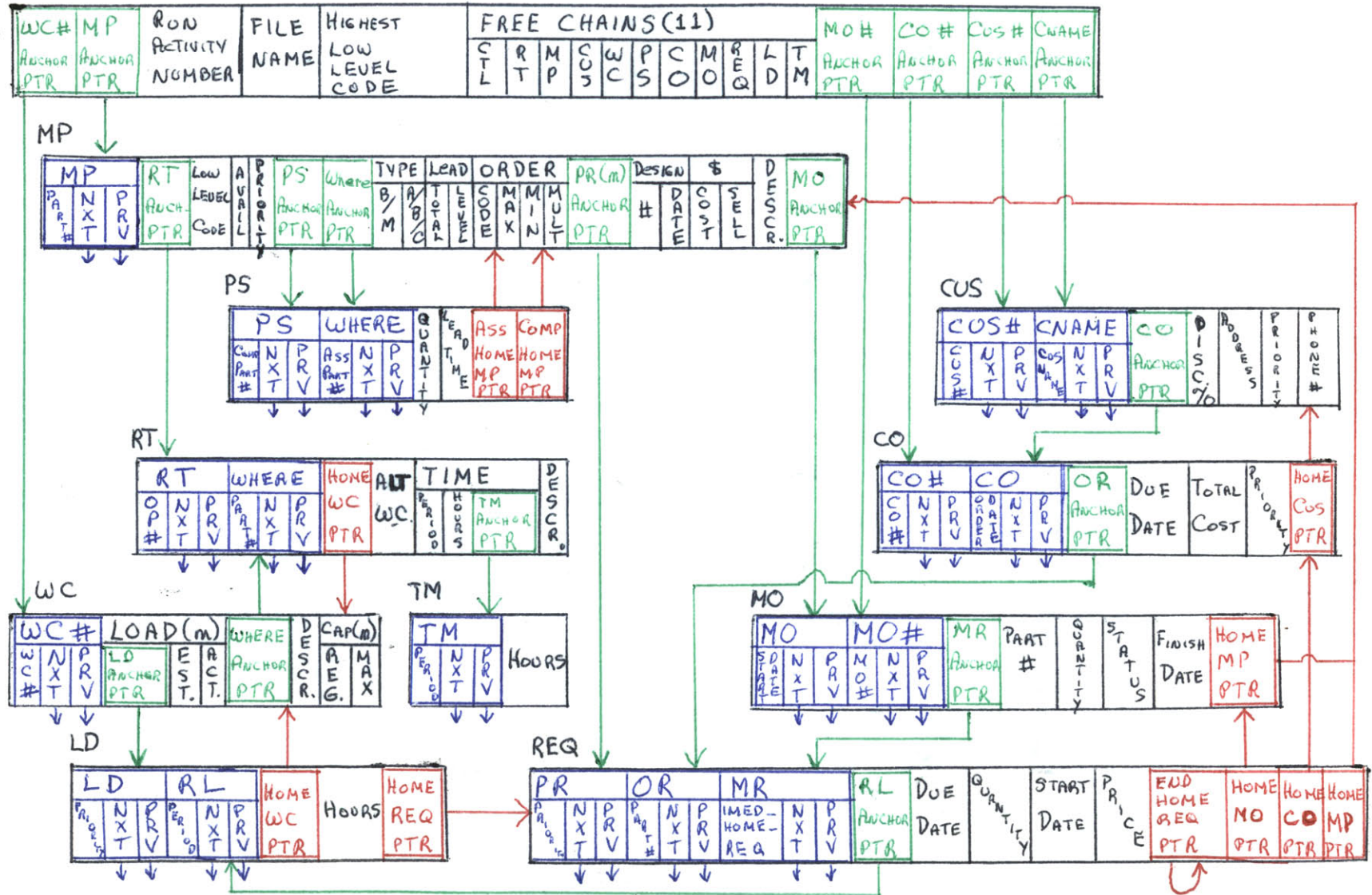
one record for each load item placed on each work center period by each requirement. This new file will have the following two chains in it:

<u>CHAIN</u>	<u>KEY</u>	<u>ANCHOR</u>
LD	Priority	WC.LOAD(n).LD_anchor_ptr
RL	Work Center_#	REQ.RL_anchor_ptr

The LD chain connects all the load elements on a work center period, thereby allowing load elements to be analysed and removed if necessary. The RL chain links all the load elements caused by a single requirement, thus allowing a requirement's load to be removed from work center totals. Each LD record also includes both a home\_WC\_ptr and a home\_REQ\_ptr to speed traverses between the WC and REQ files. Its primary fields are the number of hours needed, the priority of the requirement, and the work center number and period which are being loaded.

These additions make the REQ file the heart of the OLPP file system. It forms the dynamic interconnection among the product, sales, work center and scheduling data bases. REQ records are directly accessible from the CO, MP, WC, LD and MO files. When a REQ record is examined it is usually for the purpose of going from one of these files to the other. Complete communication through the REQ file is not possible without the addition of REQ.home\_MP\_ptr and REQ.home\_MO\_ptr fields. The diagram on the next page represents the the final design of the OLPP file system:

CTL



## 2.5 Accessing Considerations

The final OLPP file system has no provision for rapid direct access to particular records. Since all the files are chained they must be searched sequentially in order to determine the contents or existence of a specific record. This is an efficient scheme for processing entire lists but not for finding one record in a long list. The OLPP environment requires both sequential and direct access to its files. This combination is often achieved by providing some type of index into the lists. Three potential methods of indexing a file are:

- 1) A sorted index, such as one containing every tenth key value and the address of their corresponding records.

- 2) A multi-level index with the highest level containing every hundredth key value and the address of the subindex for this portion of the file. The subindex would then contain every tenth key value and the address of these specific records within the range covered by this subindex.

- 3) A hash table containing the address of each record at a location within the table which is determined by applying the hashing function to the key values.

The first of these schemes speeds up the search process by reducing the maximum number of keys which must be examined to find a specific record. Its only drawback is that the index must

be modified when records are added to the file system. This is necessary so that the key values in the index are evenly spread out over the file, so that all parts of the file will enjoy approximately the same accessing times. The second reduces the maximum access time even farther but increases the complexity of the index which must be maintained.

The hashing method provides the fastest accessing time of all by reducing the maximum number of records which must be accessed to find a record to two. It also provides uniformly fast access for all key values while the indexing methods provide fast access only to those whose keys are in the index. An additional benefit of the hashing scheme is that it does not require an index to be maintained in a sorted manner. A hash table requires modification to only one location when a record is added to or deleted from the file. These advantages come at the cost of three potential drawbacks. The first, the time necessary to hash the key value, becomes insignificant when compared with the time used by the other schemes to access unnecessary records. The impact of the second cost, the need for a large hash table, depends on the environment of the particular OLPP system. In most time sharing systems, this extra space is well worth the speed it gains for interactive users. Finally, this method does entail either the creation of a hashing function and table which results in unique locations for each key value or the inclusion of an overflow which handles non-unique mappings without seriously degrading system performance.



Particular hashing functions, table sizes and overflow methods depend heavily on the range of keys being considered and the amount of space available. In fact, for the limited requirements of the experimental OLPP system, hashing was totally unnecessary. Consequently, specific hashing algorithms will be omitted from this thesis and from the current version of the OLPP system. This is a reasonably well explored area with many such algorithms in use (such as in MULTICS directories). The definition and implementation of specific hashing methods will be postponed until the OLPP system is tailored for particular users.

## 2.6 File System Primitives and Utilities

As was mentioned earlier in this chapter, the file system contains two modular interfaces, one between it and the operating system (Primitives) and another between it and the processing programs (Utilities). These interfaces exist to localize machine dependence on one hand and file system conventions on the other. They also serve together as a tool for freeing processing routines from the need to get deeply involved either in the file system or with the operating system. This makes any modifications to the system, easier to implement.

The design of these subroutines is critical to the performance of the entire OLPP system because all parts of the system access the file very often. Efficiency thereby becomes the primary design criterion. However other considerations must also be taken into account before these routines become practical aids for system development. Should there be a few general functions or many specialized ones? The simplicity and development advantages of the first must be weighed against the operating efficiency of the second. What is desirable is a small number of efficient routines which are general enough to operate on many or all files rather than only one. This in turn influences a general design for files which makes them look the same to both primitives and utilities. This can be done by dividing records into a standardized header containing information important to the file system and a non-standardized trailer containing the rest of the data. In the OLPP environment

this led to the adoption of a standardized format for chains and the placement of all chains in the beginning of each record.

The primitive functions of the OLPP file system are:

- 1) f\_read - reads a record from a given file.
- 2) f\_state - tests if a user has a previously saved version of the file system.
- 3) f\_write - writes a record into a given file.
- 4) f\_close - closes a file.

The function of these primitive commands should be self explanatory. These are the building blocks on which the utilities are based. The precise construction of these routines will not be described any further, since their structure depends solely on MULTICS. They are the only part of the system (ignoring potential differences in PL/I compilers) which would have to be re-programmed if and when the OLPP system is ever moved to a different machine and operating system environment. They are all implemented as entries in the f\_read routine.

The functions performed by the OLPP file system utilities are:

- 1) opn\_fs - open file system.
- 2) cls\_fs - close file system.
- 3) gen - generate a record in a given file and chain(s).
- 4) get\_a - get a record given a pointer to it.
- 5) fnd - find a record given a key value and chain.
- 6) nxt - get the next record on a given chain.
- 7) put\_a - put a record given a key value and chain.
- 8) del\_entry - delete a record given a key value & chain.
- 9) del\_entry2 - delete an entry from a given chain.

The functions of these commands should also be pretty self evident. Basically they perform the mundane standard functions necessary for list processing. They were chosen because they were the largest set of utilities which were general enough to be

useful to almost any routine which desired to use the file system. All file accesses by the processing routines go through this interface. These are the only routines in the system which are dependent on the chain structure conventions of the file system. They are all implemented as entries in the open file system (opn\_fs) routine.

## 2.7 User File Commands

The direct interface between the user and the file system is the file command portion of the user command language. The formats and functions of these commands are designed to match the abilities and needs of the OLPP users. These commands exist in two groups, one for input and one for output.

The input commands are designed to provide efficient input of all, or part of, one, or more, records in a given file. They are the routines responsible for creating and maintaining the chains and pointers in the static portion of a company's data base (MP, PS, RT, TM, WC and CUS files). The exact formats of these commands can be determined by typing out the command name desired followed by the single '?' argument. The input commands are:

- 1) build - This command accepts detailed descriptions of parts and creates MP records from them. It then asks the user to supply the bill of materials for each assembly so that it can construct the necessary PS records. The build routine is the one which checks and maintains low level codes automatically. Finally, it asks the user to supply the information necessary for it to build the RT and TM records for each manufactured part.

- 2) new - This command creates new WC and CUS records.

- 3) entinfo - This command enters new or modified information into any user accessible field of any record.

- 4) order - This command enters customer orders into the file system. This involves the generation of one CO and as many REQ records as are necessary. It may also cause a new CUS record to be created. This command invokes the load module and will be discussed more fully in chapter 3.

- 5) schedule - This command invokes the schedule module, which creates MO records. It will be discussed more fully in chapter 4.

- 6) slide - This command invokes the slide module, which requests new regular and maximum capacities for the initialization of the last (26th) period. It will be discussed more fully in chapter 4.

7) remove - This command removes any single record from any file.

In an actual implementation of the OLPP system these commands would probably have to be modified to also accept data from devices which are more capable of high volume input.

The output commands are designed to output the information contained within the file system in formats which are directly related to specific user needs. Each of these commands has various alternative forms, which either change the amount of detail in a report or change the way in which it is arranged. Exact formats can be obtained once again by typing the command name followed by a '?'. The output commands are:

1) explode - Explodes a gross part requirement into gross requirements for all its components. The possible options are single level, indented, cross sectional and summarized explosions.

2) where - Implodes a part showing how many times and at what levels it appears as a component in any other assemblies. The possible options are single level, indented, cross sectional and summarized.

3) mlist - lists all parts which belong to a specific subset of the MP file. The possible groups are end products, assemblies, subassemblies and detail parts.

4) disinfo - prints out the contents of any record in any file.

5) colist - prints out the contents of the CUS, CO and REQ files associated with a given customer order. Can provide detailed information or a condensation of it.

6) wcprofile - prints out the current load status of one or more work centers in one or more weekly periods. Can also provide regular and maximum capacities, estimated and actual loads, and just those work centers which are loaded beyond a specified point.

In an actual OLPP implementation, these commands would probably have to be extended to provide for output to other devices which are more capable of high volume output. Consoles are primarily

useful for their interactive capability. They are the only device supported by the current OLPP implementation, because they are the main I/O device of the MULTICS environment and because they are excellent tools for system development.

## CHAPTER THREE - LOADING

### 3.1 Introduction to Loading

The initial step in creating an OLPP system is the generation of a company's master, product structure, routing, time, work center and customer name files. At this point the system is primed to begin the three part loading, scheduling and updating cycle. As customer orders enter the system, they are translated into pegged requirements, which are in turn used to build up an estimated load on the work centers in specific time periods. Originally an order's pegged requirements are generated in a critical path sequence in reverse chronological order from its due date. This requirements generation relies on each product structure record to contain not only the number of each component which is required for an assembly, but also the lead time by which the component's completion must precede the completion of the assembly. This lead time may differ for each component and is the amount the component's due date is offset from that of its assembly. Requirements may be moved ahead of their original critical path sequence if they cause an overload in one period, but can either fit into a preceding period of the same work center or into an alternate work center (specified in the



RT record) in the same period. The estimated loads represent the standard capacity necessary to manufacture sales commitments. They give a timely approximation of the growth of actual work center loads.

Detailed scheduling of manufacturing work center loads is a complex process which is delayed until there is a high probability that no new orders will alter the combination of economic order quantities and start dates which best satisfies the pegged requirements of any part. Although the actual load figures, which are based on manufacturing orders, may differ slightly from the estimated load, the total load on each work center will be the same. The only difference will be a minor shift of the load into earlier periods as requirement quantities are padded to reach minimum or multiple order sizes. These scheduled manufacturing orders can form the basis for weekly production only if they are updated with the status of previous schedules. This feedback is necessary because schedules are at best approximations of reality. They contain enough slack to absorb the effects of minor deviations from what was predicted on the basis of past performances. However, some random occurrences may have effects which cannot be absorbed in a weekly schedule. If these errors are not taken into account by the system, all future schedules will become progressively useless. So after each weeks production is completed, the system must be informed of which orders were completed and which weren't. Incomplete orders must be fit

into succeeding schedules, presumably as soon as possible.

The OLPP order entry routine can be used for three purposes:

- 1) Entering a customer order
- 2) Costing an order
- 3) Testing the feasibility of producing an order
- 4) Finding the earliest possible order delivery date

The primary objective of this phase of the OLPP system is the entering of a customer order. This process causes automatic validation of the customer name or number if they exist, or the generation of a new CUS record if they don't. It also calculates the cost of each order item and total cost of the order. After this preliminary bookkeeping is accomplished, the feasibility of producing the order before the due date is determined. When the manufacturing of this order in its critical path sequence causes the overload of one or more work centers, the user is notified of the scope, cause and potential solution to the problem. The marketing and or production departments can then work interactively with the system to obtain a mutually agreeable solution. Once an order has been accepted by the system, the loading routine generates PR chains, updates the estimated work center loads, and generates LD records and chains them into the proper LD and RL chains.

The second and third versions of the order command exist so that the user can obtain information generated by one part of the order entry process without incurring the delay and overhead of the functions which he does not want. Feasibility

checking illuminates potential bottlenecks at the earliest possible time. This gives everyone concerned the greatest possible chance for revising their plans and for making the most of the opportunities presented to them.

The fourth option allows the user to rely on the system to calculate the earliest possible delivery date for an order. It considers both the total lead times in the MP records of the order items and the feasibility of starting an order in the next period. When it finds that the theoretically possible delivery date is infeasible due to the current load on the plant, it keeps recursively trying the next later period until a feasible date is found.

### 3.2 Order Entry

The order command is designed to be used by the marketing department for both customer negotiation and actual order entry. It has the following three formats:

- 1) order <customer name or number>
- 2) order cost
- 3) order -test
- 4) order -date

The -cost option can be combined with either the -test or -date options by placing it on the same command line as them. In all four cases the command name is recognized by the OLPP routine, which calls the order routine and passes the argument to it. If an actual order is being entered, the CUS file is examined to verify that the specified customer exists. If no such customer can be found, a new CUS record is generated. The user is asked to supply all the missing information, such as address and phone number. After the CUS record has been settled, a new CO record is chained into this user's CO chain. The system asks for the following order header information and fills it into this record:

- date of order
- due date
- priority
- discount percentage

The last two of these are optional. The discount percentage is used by the marketing department to give a specific discount to an entire order. This discount overrides the

customer discount found in the CUS record, but can, in turn, be overridden by the item discount. The priority (1-5) is added to the priority in the part's MP record (1-5) and that in the CUS record (1-5) to rank orders and requirements relative to each other. Any one or all of these may be zero if the company does not want to distinguish orders on that basis. Priority ratings are only used by the load leveling routines when they are instructed to bump low priority requirements from overloaded work centers.

In options two, three and four the order routine omits the generation of CUS and CO records and starts immediately to accept the order items. The body of the order is entered after the system asks the user to input the following item information in the columns under the headings printed out on the console:

Part#	Quantity	Item Discount
-------	----------	---------------

A blank line in the input stream causes the system to ask for a new order#. If this is NULL, no more data is read in. Otherwise multiple orders can be read in at the same time. The item discount is needed only in options one and two. This information is immediately translated into a chain of 'dummy' REQ records. This chain is the temporary internal representation of the order and is not inserted into the REQ file nor linked to any other files. Instead, it is maintained as an independent working copy of the order. It is used primarily by the feasibility checking module as a mechanism

for the efficient summarized explosion of the order items and as a means of determining the loads which will be placed on the work centers. In options one and two, it is also used to help calculate the cost of each item and of the entire order. The item costs are retrieved from the MP records and the discounts applied to them. This cost information is then output to the user and stored in either the CO or REQ records. Once an actual order has been accepted by the system, these dummy records are moved into the REQ file and patched into the proper OR and PR chains.

In options one and three, control passes from the order entry module to the feasibility checking module. In option two, the cost information is shown to the user and control returns to the OLPP command level. In option four, the feasibility checking module is called once for each due date that is tried.

### 3.3 Feasibility Checking

Feasibility checking is the process of comparing the standard capacity necessary to produce an order with the regular work center capacities available in the periods preceding the order due date. The system will not accept an order until it can guarantee that the proper work center capacity and raw materials will be available in the proper periods. The first step in this process is the comparison of the total lead time necessary to purchase raw materials or detail parts with the amount of time between the current date and the order due date. This total lead time figure is found in the MP record of every product and represents the minimum time necessary for the purchasing and manufacturing cycles of each part.

Once this preliminary check is completed, the order requirements must be exploded into all of their pegged requirements. This complex recursive process can be simplified by noting that a subassembly may occur several times within a given product and several more times within other products in the same order. If all these occurrences were known the first time a subassembly was exploded, the system could use the information it learns in the explosion process for each one and save itself the trouble of multiple re-explosions of the exact same product structure. Therefore rather than exploding parts randomly in the order they are discovered in the explosion of each order item, the OLPP

system explodes assemblies in the order in which they occur on the previously mentioned dummy REQ chain.

This chain is ordered first by low level code and then by part number within groups having the same low level code. Initially only product requirements are on the chain. Then as each assembly is exploded, the necessary quantities and offsets of each component are determined and a new dummy REQ record generated and patched into the chain for each one. The fact that each component must have a larger low level code than its parent assembly, guarantees that each new record will be inserted farther down the chain and will eventually be exploded itself. Grouping by part number insures that all requirements for the same part can be handled by only one explosion.

The overall purpose of this exercise is to compare order loads to available capacity, so we must also use this explosion process to determine the total load placed on the plant by this order. If we knew ahead of time that this order would fit or if we were committed to this due date, we could simply load the actual work centers and look for overloads. However the order is being evaluated and not necessarily loaded onto the system, so a separate unloaded copy of the work centers is created on which to aggregate the load from this order. The system also realizes that at a later time it may either have to generate LD chains or modify this order in some way to make it more feasible. Consequently, it also



generates dummy LD chains off the dummy work centers during the explosion process. The existence of this separate subset of the file system is necessary to provide an efficient working representation of an order and its loads. This information can either be discarded, modified or integrated into the file system, according to the user's evaluation of the desirability of the order.

The explosion routine examines each part in the order in which it appears on the dummy REQ chain by traversing first its RT-TM chains and then its PS chain. Each record on these three chains is used once for each requirement with this part number and then the next record on the particular chain is fetched. The RT-TM chains are used to claim capacity for each requirement by providing the information necessary to update the estimated load and the LD chains in the proper work center periods. Each RT record represents one operation to be done in one work center while assembling a part. It also contains a pointer to the TM records which list the hours of capacity which this operation will require in periods relative to the due date of this assembly. This file is used both to list the manufacturing operations which go into a part and to indicate the work center capacity hours necessary to manufacture this part. To speed up the loading process, all TM records are aggregated into the first occurrence of a work center in a RT chain. Later occurrences of this same WC in in this RT chain will have neither a WC.WHERE chain nor TM information. This

is only a heuristic to speed up processing and has no conceptual significance.

The PS chain is used to determine both the quantity and time period in which a component part will be required. It is the potential uniqueness of these quantity/date pairs which cause a separate REQ record to be generated for each separate use of a component part. Class A, items which are also assemblies, are the only parts for which both a RT and a PS chain are examined. Detail parts and class C items do not cause any new REQ records to be generated and class B & C items do not add any load to the dummy work centers. Therefore, depending on the class of the part, either none, one or two chains are searched from the MP record.

Once the entire dummy REQ chain has been exploded to its lowest level requirements, the dummy work center records will be loaded with the estimated factory load of this order. These loads can be easily compared with the currently available capacities to determine the feasibility of manufacturing this order by its due date. When order is feasible and the user requests the system to load it, the dummy REQ records can be easily chained into the proper OR and PR chains in the permanent REQ file. Additional fields such as home pointers must be filled in at this point. These requirements will be used by the system for scheduling or for possible user retrieval. The dummy LD records can be easily inserted into the existing LD chains in the LD file and the

estimated loads from the dummy WC records added to the estimated loads in the WC file. The RL chains in the dummy LD and REQ records can be moved directly as is into the actual file system.

As will be explained more fully in the next chapter, the load routine must also build a list of all requirements which must be scheduled immediately. If there are any such requirements, the load routine must invoke the scheduling procedure and pass this list to it.

### 3.4 Load Leveling

The system outputs a detailed report to the user whenever current plant loads or insufficient lead times make it infeasible for an order to be produced in its critical path sequence terminating at its due date. Multiple orders input with the same command cause one summarized report to be given to the user rather than one for each order. The first part of these reports is a problem description:

- which lead times are insufficient and by how much
- which work centers are overloaded and by how much
- what the capacities of these work centers are

To make this listing even more useful the system also shows which orders have contributed to this overload and by how much. This includes both the requirements from the order being considered and all other orders already accepted by the system.

When the system is asked to calculate the earliest feasible due date, it prints out both the date which it found and a list of those dates which were tried and discarded. For each unfeasible date it lists the number of work centers which were over loaded and the average percentage each was overloaded. This allows the user to estimate how worthwhile it would be to try to level the loads enough to make an earlier date feasible.

The final part of the problem report is an ordered list of potential solutions:

- 1) Raise the regular capacity of the overloaded work

center(s) by a specified amount. This represents a commitment either to overtime or to additional manpower or mechanical resources.

2) Try moving requirements to the alternate work centers listed in the RT records. This is a simple solution which depends on the existence, suitability and availability of capacity in another work center in the proper period.

3) Move certain requirements into earlier periods. This is a good solution because it allows the order due date to be met at slightly increased inventory carrying costs. This type of solution is offered only if there is some available capacity in this work center in a previous period. Movement of a requirement is not a simple process since both its loads and all requirements within its product structure must be moved with it. In addition this movement may cause overloads in any one of the work centers affected by it. So it is hard to predict whether or not this 'nice' solution will work.

4) Shift the order due date back. This entails sliding all the loads in the dummy work centers over one or more periods and trying again.

5) Decrease the order quantity of one or more items. This requires re-explosion to modify some or all of the dummy REQ, WC and LD records.

6) Bump other orders which use an overloaded work center to a later period. Changes to these orders require co-ordination with the customers involved and some complex

unloading and reloading. It may also involve the even more troublesome process of unscheduling and rescheduling.

This report should give the user the opportunity to get a feel for the problem and its potential solutions.

At this point the user has several alternative ways of working out the problem. He can either execute any of the regular user file commands which would give him a better picture of the situation or he could try one or more of the above solutions. For example he could invoke one of the user file commands to examine the work center load profile or the details of some other orders. A user may also invoke one of the special load leveling commands which are worked out in the dummy files. Then this trial solution is tested for feasibility and a regular overload problem report is printed out. On the basis of this report the user is given the opportunity to discard the solution, save its results as part of an eventual solution or finally to load the order and the previously saved modifications, by inserting them into the permanent portion of the file system. In this way the user can simulate various solutions on a trial and error basis. He can also test the aggregated effect of the most effective trial solutions by saving them in a second set of dummy files. This is all made possible because the load module has three separate copies of the WC, REQ and LD files within which it can work. These three distinct environments are necessary to:

- work out the net effects of some trial

modification

to an order or requirement.

- save the combined effects of some number of user approved solutions.

- preserve the original state of the file system until a final solution can be found.

The new set of load leveling commands which the user has at his disposal to manipulate these environments are:

1) exec <any OLPP command> - This is an entry to the OLPP routine which bypasses the initialization or shutdown of the file system and which returns to the procedure which called it. This is the mechanism by which regular OLPP commands can be invoked from within the environment of the load routine without losing any of the trial solutions.

2) newcap <WC#> <period#> <new regular capacity> - This command replaces the regular capacity of a work center period with a new value and then reviews the new feasibility of the order under consideration. Any new value greater than the maximum capacity of the work center is rejected.

3) modify <optional order#> -date <new date> - This command modifies the due date of either the specified order or the order being reviewed (default condition). Modifying the due date of the order being reviewed merely entails incrementing the due date of all its requirements and sliding its loads forward. Other orders must first be exploded, and their loads subtracted from the periods they now occupy and added to the new period (all within dummy work centers).

4) modify <optional order#> -item <item list> - This

command provides new quantities for all parts on the item list (part# quantity pairs separated by commas). Once again the order being tested is the default order if no order number is given. Each item must be re-exploded and its requirement quantities and loads modified to reflect the change.

5) modify -any - This version of the modify command requests the system to start bumping low priority orders until all overloads are alleviated. As was mentioned earlier this is a complex process because the REQ records may be in several chains and may involve both estimated and actual loads. The system starts with the lowest priority order in the LD chain of the most overloaded period and moves it forward until the first feasible due date which does not use this work center period. This process includes movement of all the order's loads and REQ and LD records. This may also involve the rescheduling of any parts whose previously scheduled requirements are moved by a special call to the schedule routine.

6) move -any - This version of the command instructs the load routine to attempt to remove overloads either by moving some of the trial order's requirements to earlier time periods or by moving one of its requirements to an alternate work center. Any movement of a requirement to an earlier period may also include the movement of requirements which provide components either directly or indirectly to the original requirement. This is because the requirements have



originally been scheduled in the critical path sequence necessary to manufacture them. Changes to the due date of any requirement within a product structure can never be allowed to violate this sequence by finishing a component requirement after it is needed by an assembly. However, it can always allow a component to be finished earlier than it is needed. Since the movement of a requirement may be a complex process, the system employs the heuristic of never trying to relieve an overload on a work center by this fashion unless there is at least enough capacity available in earlier periods of this work center to accept the needs of the requirement which is causing the overload. This does not guarantee that all attempted movements will succeed but it does eliminate those which are obviously infeasible. The system first goes through the dummy LD chain for an overloaded work center period and finds all single requirements whose movement can remove the overload. These are then ordered according to cost. This list is searched in order until a feasible movement is found. If no single requirements are either feasible or large enough to remove the overload, the system searches for any combination of requirements including the first element of the LD chain whose movement is both feasible and will remove the overload. If this doesn't work it ignores the first and tries with the second etc. The initial part of this strategy is in some sense optimal but the second is just looking for the first solution that works. This reflects the overall heuristic nature of the

OLPP system. If a simple optimizing solution exists it is used , otherwise satisfising heuristics are used.

7) move <requirement#> - This command tries to move a specific requirement into an earlier period. This may cause the movement of the requirements which supply components to this requirement. The requirement# refers to a reference number which is printed out in the load diagnostic report and which has meaning only within the load environment. The movement is first made and then the feasibility of this new environment is reviewed and reported to the user.

8) move -alt <requirement#> - This command tries to move all operations in the RT chain of this requirement's home MP record to alternate work centers. All feasible movements are made within the dummy WC and LD files and a new status report is output to the user.

9) save - This command adds the WC.LOAD totals and the REQ and LD records from the temporary single command environment into the temporary 'save' environment.

10) load - This commands integrates the WC.LOAD totals and the REQ and LD records from the temporary save environment into the permanent file system.

## CHAPTER FOUR - SCHEDULING

### 4.1 Scheduling Strategy

The job of the scheduling module is to convert the class A pegged requirements which were generated by the load module into a schedule of manufacturing orders. The load module considers only the feasible manufacturing sequence of a single part which is closest to its critical path sequence. The scheduling module operates on a more global strategy and provides the user with four potential levels of flexibility:

- 1) A separate 'order strategy' for each part. This defines the most economical lot size for each of its manufacturing orders.

- 2) A separate 'scheduling window' for each part. This is the number of periods whose requirements can influence the size of the last manufacturing order in the primary scheduling period.

- 3) A separate 'product lead time' for each part. This reflects how long the purchasing and manufacturing cycle is for this part and all of its components and consequently which period is the critical period.

- 4) A standard 'planning horizon' for a given schedule. This describes how many periods the primary planning period will precede the critical period which is the last period in which an order can be scheduled and still produced on time. These levels of flexibility enable the OLPP system to produce a schedule which is attuned to the different characteristics of each part and yet is an efficient means of producing their requirements.

The order strategy parameters in each MP record determine the proper number, sizes and dates of manufacturing orders which

minimize set-up and inventory carrying costs for each part. They allow each part to be scheduled in a distinct fashion which is optimal for it and it alone. Generally speaking the scheduling algorithm tries to find the smallest number of orders with the maximum economical lot size which fulfill all the requirements within the primary scheduling period. Whenever the remaining requirements are insufficient to warrant a maximum size order, it builds one last order of a size which includes as many other requirements in the scheduling window as possible and as little extra inventory as possible. The scheduling window is a figure which represents the trade-off between inventory carrying costs and set-up costs. It reflects the relative desirability of generating two small manufacturing orders for the same part in adjoining periods versus that of generating one larger one in the earlier period.

As we have mentioned previously, the OLPP system postpones scheduling until there is a very low probability that any new order will make the schedule obsolete. However, we must generate the schedule of actual manufacturing orders early enough for all detail parts and raw materials to be purchased and delivered before they are needed. So the proper time to schedule a part is at least the period before the first of its components must be ordered or started into production. If we waited any longer, the component's due date and consequently the order's due date couldn't be met. This

last fact makes the critical period both the last one in which an order could be accepted for this part and the first one in which we should know the final profile of this part's requirements one lead time into the future. The OLPP scheduler determines which period it should be scheduling for each part by examining the total lead time field in each MP record. This is a very flexible scheme since the constraints of one part are not forced onto any others by the restriction to a common lead time.

The scheduler adds a standard planning horizon to each part's total lead time in order to permit the company to generate its schedule as far ahead of the critical period as they would like. This may be appropriate where customers order relatively early or where this information is needed more than one period ahead.

#### 4.2 Scheduling Algorithm

The scheduling module is invoked by the following command:

```
'schedule <standard planning horizon>'
```

```
'schedule <list of part#'s and periods>'
```

The first of these is the one used for regular weekly scheduling by the production planning department. It is recognized by the OLPP routine which calls the scheduling module and passes the argument to it. The second command is called directly by the load and update modules to reschedule parts which have been moved into or out of a previously scheduled work center period.

Scheduling is a process which is done one part at a time, so it is logical to assume that the weekly scheduler can do its job in one pass over the MP file. However there is one problem which makes this infeasible. The scheduler has the ability to move requirements from any period within the scheduling window into the primary scheduling period. This causes the related movement of all requirements which supply components either directly or indirectly to this requirement. If any of these requirements have a part number less than that of the assembly which is moved, the scheduler will have already past them. It must either scan the MP file in low level code sequence or make repeated passes over the entire file until there is no more shuffling of requirements. The OLPP system has chosen the low level code sequence because it

avoids having to reschedule any parts. If a company used a part numbering scheme which was proportional to a part's low level code this could be done in one pass over the MP file and any lists sorted by low level code could rely on part number instead. The list of parts and periods passed to the scheduling routine by the other modules is already in this order. In the regular version of the command the scheduler decides which of a part's PR periods is the primary period by adding its total lead time to the standard planning horizon. The lead time increment is necessary because the pegged requirements are stored by due date and not by start date. In the other version, there is no primary period. Instead the scheduler reviews the entire manufacturing order profile of the parts on its list by comparing their M0 records with the current status of their pegged requirements.

The scheduling module uses five order parameters which are: order code, scrap percentage, and maximum, minimum, and multiple order size. The order code field determines how the other fields will be interpreted. Currently, there is only one order strategy implemented but the inclusion of this field allows the smooth insertion of any other strategies which may be desired at a later date. The scrap percentage field defines the percentage that must be added to orders for this part to insure the completion of the proper number of items. This field is not examined until scheduling time because it is more efficient for the scheduling routine to add a single

increment to manufacturing orders than it is for the load routine to add it to every requirement. This delay also simplifies the explosion and load leveling routines since they do not have to worry about adding in or subtracting away scrap factors. Since this percentage is not reflected in every level of an assembly, they must be roughly in proportion to each parts low level code. This will insure the completion of enough components to at least start the proper number of assemblies.

The remaining parameters are used as their names imply. The maximum and minimum fields are the respective limits of economic lot sizes for this part. The multiple field contains a number by which the lot size must be evenly divisible. The scheduling algorithm totals up the pegged requirements for a part in its primary scheduling period and checks of there is sufficient inventory available from the previous period. If there is any it is claimed and the total of unfilled requirements decremented. If there are still requirements to be filled, it creates as many maximum size orders as are necessary to fill them. When the unfilled total becomes less than the size of a maximum order, the rest of the scheduling window is examined. As many of these requirements are moved into the primary period as are necessary to bring the last order in the primary period up to maximum size. These moves are made by the invoking of the 'move -t <part#> <period> <immediate use pointer> <distance>' command. This command



specifies that a requirement and all of its component requirements are to be moved a certain number of periods ahead. The '-t' argument tells the move routine to return a boolean value which indicates whether the move was feasible and completed or whether it was infeasible and aborted. The move module determines feasibility by checking both for available capacity and for violation of a part's total lead time. The scheduler does not move requirements except into primary periods, which, by definition, must be the length of the length of the planning horizon beyond the critical period. This critical period is calculated from the total lead time necessary to produce this part and all of its components. Therefore the movement of a requirement for an assembly into a primary period can never cause any of its component requirements to be moved into any period preceding their own primary period. If enough requirements can not be moved into the primary period to create one last maximum size order, the multiple and minimum fields are examined to determine the smallest possible lot size which fills all the unfilled requirements now in the primary period. Any production necessary to pad the lot size up to a minimum or multiple size is made available as inventory in the next period. In all cases the only requirements manufactured early are those for whom there is available capacity and which are necessary to round up the remaining requirements which were originally in the primary period.

When called by the user the scheduler makes one pass over the MP file for every low level code value. It is finished when it reaches the maximum low level code which is stored in the CTL file. When called by the load and update modules it considers only parts on its argument list. Each pair in this list defines a part number and a previously scheduled period in which the PR chain has been modified. Since periods are scheduled sequentially, without any reference to past periods, the scheduler knows that any changes which will have to be made to manufacturing orders will occur only in the specified period or in the ones following it. The removal of a requirement will entail the reduction of one or more manufacturing orders in the original requirement period and possibly the restructuring of orders in following periods. Inserting a new requirement into a period may affect the last manufacturing order in that period and the ones in subsequent periods. This allows the scheduler to ignore any manufacturing orders which occur before the change since they can not be affected by it. The scheduler examines the PR chain containing the change and the first M0 record on this part's M0 chain which could possibly be affected by the change. It then proceeds down the M0 chain comparing the existing orders with the new state of the pertinent PR chains. If anything deviates from what ordinarily is the best order policy for this part, the M0 chain is rebuilt from the point of the deviation. This entails creation of the proper M0

records and updating of the REQ,LD and WC files. When the scheduler reaches the end of its list it is finished. Rescheduling in this manner will update the schedule to reflect whatever changes have occurred, but it cannot undo loading or purchasing decisions which were made on the assumption that certain cancelled requirements needed capacity and raw materials.

#### 4.3 Generation of Manufacturing Orders

Once the scheduling algorithm has decided on the precise number, sizes and dates of manufacturing orders which are needed for a particular part, it proceeds to generate the actual MO records and make other related changes to the data base. Whenever a new manufacturing order is required, a new MO record with a unique MO# is generated and chained into the MO# chain and onto the end of the proper MO chain. This record is then supplied with the proper part#, order quantity and home\_MP\_ptr. Its finish date is the due date of its earliest requirement and its start date, this date minus the single level lead time. Its status represents that it has been scheduled but not started or completed.

The most complex part of the MO information is the MR chain which includes each REQ record which is being fulfilled by this MO. When the individual pegged requirement sizes are less than or equal to the size of the MO, this is a straight forward process of chaining the REQ records onto the proper MR chain. The presence of the MR chain pointers within a record serves as an indication that this requirement has been scheduled. This process becomes more complex when the MO has been padded with inventory to meet a specific order size or when a requirement must be split between two or more MO's. Inventory is handled by generating a separate REQ record and chaining this new record into the head of the PR chain in the next period and into a MR chain. This record will be examined

by the scheduler in future periods and eventually replaced by an actual requirement. It can be identified as an inventory record by its null lmed\_home\_REQ\_ptr and home\_CO\_ptr fields.

Split requirements are more difficult to handle because a single REQ record can not be chained into more than one MR chain. The need for split requirements will arise either when a requirement quantity is larger than the maximum lot size or when only part of it is needed to complete a maximum lot size order. This situation is handled by the splitting of the original pegged requirement into as many parts as is necessary. Each split requirement will have a unique quantity and will be on a separate MR chain. However, each will have the same part#, due date, lmed\_home\_REQ\_ptr and End\_home\_REQ\_ptr. They will also occur in adjoining records on their OR and PR chains, so they can easily be recognized as part of the same original requirement.

At the same time the MO records are being generated, the WC and LD files must be reviewed and updated. All estimated loads for the requirements in this MO must be changed to actual loads. The home\_WC\_ptr, hour and period fields necessary to do this are found in the LD records on the RL chain of each requirement. LD records must be created for each inventory REQ record which has been created and the proper actual load totals updated. The LD records on the RL chain of any requirement which has been split must also be split and spliced into separate RL chains for each of the split

requirements. Also the load hours for each of the original LD records must be moved into the actual load totals and decremented from the estimated ones.

#### 4.4 Updating

When the weekly scheduling run is finished, the user may invoke any of the user file commands to get such reports as:

- a profile of each work center capacity and the current actual and estimated load on it.
- a summary of all manufacturing orders which are scheduled for future periods.
- a detailed report of the load facing each work center in any period.
- detailed manufacturing orders, with appropriate routing and material requisitions for use as factory paper.

These type of reports form the feedback from the system to the user. The production planning department either can accept these as the scheduler has prepared them or can invoke the 'level' command, which gives access to the interactive load leveling commands discussed in section 3.4. Different users may wish to level loads to accomplish goals which are not recognized by the scheduler such as leveling of under capacity manpower requirements.

No matter whether the schedules are done solely by the system or interactively with the user, they are approximations of what should happen under ideal circumstances. In reality, purchase orders will be delivered late, workers will get sick and machines will break down. A weekly schedule may absorb some but not all of these random fluctuations. What the OLPP system still needs is a means of updating its knowledge of manufacturing order status to conform with events as they have actually transpired. Unless this is done the system's knowledge of the world will not conform to reality nor will

its schedules.

The vehicle by which the user communicates this information to the user is the 'update' command which causes the following column headings to be printed out to aid user input:

MO#	Status
-----	--------

The system then accepts lines of input until it finds a blank line. Since the update module assumes a majority of orders will be finished on schedule, it has adopted 'completed' as its default status. This allows the user to list all the MO's which have been completed without specifying a status for each one. To ease the load of system input, completed MO's can be reported to the system daily as they are finished. At the end of a weekly period, or at the beginning of the next one, the system must also be notified of the scheduled MO's which weren't finished on time. The user must state the precise status of these orders so that the system can calculate the total impact these delays will have on other orders. The possible status options are:

- completed
- not\_started
- quantity <Quantity completed>
- operation <Last operation sequence# to be completed>

The completed status causes the status field of the MO record to be updated accordingly. The completed status causes each REQ record on the MO's MR chain to be deleted. This deletion will be the sign that the requirement has been met and



consequently is no longer needed by the system in either the PR, OR or MR chains. The not\_started status causes the entire MO to be shifted to the next period. This movement is accomplished by a 'move -now <part#> <lmed\_home\_REQ\_ptr>' call. This tells the move module that a specific requirement, together with all requirements which rely on it either directly or indirectly as a component, and all of their loads must be moved from period# one to period# two. The arguments are necessary to uniquely identify a pegged requirement.

The other two status options are alternative ways of representing the degree of completion of a manufacturing order. In case one, the existing MO is marked completed but its quantity is reduced to the amount that was actually finished. A new MO is generated for the next period with the remaining quantity and with a RL chain containing LD records for the remaining load. In case two, the existing MO is marked partially completed and a new MO with the same quantity, but only the remaining load, is created. In both cases, the requirements must be moved back to period number two and given the new RL chains. This is again accomplished with a 'move -now <part#> <lmed\_home\_REQ\_ptr>' call. As in the not\_started case, this implies the shifting of any superior requirements and all their loads.

When weekly updating is finished, the user automatically gets a report of all work centers which have been overloaded

by the shifting of requirements. He can then use this report as a basis for invoking load leveling commands, testing various combinations of solutions and implementing the most effective ones. After this additional load leveling, the system's schedules will be in tune with the actual plant situation and reasonably leveled. However the file system will include one period of past history and only twenty-five periods of future plans. To maintain pace with the passing of time, the system must be capable of purging itself of old periods and initializing new ones.

This movement of time is done through the invocation of the 'slide' command, which summarizes past history, purges dead information, slides existing periods and initializes new periods. Its first task is to check the status of all MO's with finish dates this period. If they are completed (ie. have NULL MR\_anchor\_ptr fields) they are included in an end of week manufacturing summary and deleted from the system. If they have not been completed, the user is asked to supply their status and it is acted on as in the update module. When the current period's MO's are all deleted, the system checks all the CO's which have due dates in this period. If their OR chains are also null, they are included in a completed order summary and deleted from the system. Incomplete orders are included in an overdue order report and moved into period number two. This will only occur if the order is delayed by a snag in this weeks production. Other delays are realized by

the system whenever a manufacturing order is reported late. The marketing department is notified immediately whenever requirements, which are moved back to compensate for late MO's, cause an order due date to become unrealistic. The system will update the due date by itself and remind the marketing department to warn the customer. After the CO and MO files have been purged, the PR(1) chains are checked to see if they contain any requirements. Any record remaining on these chains is in error and is reported to the user for advice as to how the requirement should be disposed of. Finally, all the records in the WC.LOAD(1).LD chains are deleted, since they are no longer needed.

At this point all dead information has been purged, so the system can slide periods two through twenty-six in the LOAD and PR fields into periods one through twenty-five. It then initializes the anchor pointers to the LD and PR chains in the twenty-sixth period to NULL and the LOAD totals to zero. Then the user is asked to specify how he would like the regular and maximum work center capacities to be initialized in the last period. He has the options of leaving them as they were in the previous week, expressing new capacities in terms of a percentage of the last ones or of specifying each explicitly. When this entire slide and updating process has been completed, the system is in tune with the latest production results and has successfully re-initialized itself for another week.

## CHAPTER FIVE - SUMMARY

### 5.1 OLPP Capabilities

The OLPP system is a production planning tool which is suitable for small custom job shops. It performs eight major functions for this class of user:

1) Data Management - The OLPP system integrates the engineering, marketing and production areas of a company's data base. It maintains this data in a co-ordinated fashion for the user and can provide on-line real time access to any portion of it.

2) Bill of Materials Processor - The heart of the OLPP system is the portion of its file system which contains descriptions of all parts used by a company, their bills of material and the operations necessary to produce them. The system also provides various routines to explode or implode these product structures and to trace all the parts using a common work center.

3) Flexible Inventory Control - The OLPP system uses an A/B/C classification system which permits parts to be controlled in a manner which reflects their cost and volume. It also recognizes the distinction between parts which must be purchased and those which must be manufactured. This allows

each to be handled separately. Finally, each part may have a distinct ordering strategy which reflects its unique cost structure.

4) Requirements Generation - The OLPP system has the facility for exploding order requirements into detailed requirements for every single component which will be required to manufacture a product. This permits the system make detailed plans which insure that all necessary component parts will be manufactured or purchased.

5) Due Date Scheduling - The OLPP system assigns due dates for all requirements which are components of some order item by using average lead times as offsets from the item's due date. This critical path sequence co-ordinates the timing of all manufacturing and purchasing so that requirements arrive neither too early nor too late. This minimizes both both work-in-process inventory and carrying costs on one hand and missed due dates, confusion and expediting on the other.

6) Load Leveling - By monitoring both sales and the load placed on all work centers by these commitments, the system gives management a reasonable estimation of what plant loads will be like in future periods. The system also compares these loads with the stated capacities of the work centers and does not allow work centers to be loaded beyond their capabilities. Through a large repertoire of load leveling commands the manager can try a wide variety of load leveling techniques and implement only the most effective ones. The

system calculates all the implications of the various load leveling solutions, thereby saving the user laborious paperwork and allowing him to test alternatives which would otherwise be too complex to try.

7) Generation of Manufacturing Orders - The flexible ordering strategy which exists for each part allows its manufacturing orders to be generated in lot sizes which are economical for it and it alone. The system can also output these manufacturing orders in a factory paper form which includes routing and material requisition information.

8) Report Generation - The availability of a comprehensive centralized data base makes it possible for the system to provide reports of any level of detail for any level of management. The system is designed modularly so that the user may define commands which generate reports in the format which will be most useful to him.

## 5.2 Necessary Implementation Considerations

The current version of the OLPP system does not yet include all of the design features described in this thesis. This reflects the size of the project, the resources which were available for it, and the priority I placed on design rather than implementation. Even when this design is completed, the system will not necessarily be a commercially viable tool. Its primary limitations stem from the fact that it was developed in an academic rather than a commercial environment. A commercial version of OLPP would have to be transferred to a commercial time-sharing system and subjected to various shakedown tests.

Moving a large software system from one environment to another is a complex problem. Although PL/1 is relatively machine independent, no two machines or compilers operate in exactly the same fashion. The system was started in an OS/360 environment and soon moved to MULTICS. Conversion routines were written for the transfer of these programs between the IBM 360/65 and the HIS 645 one year ago but there is no guarantee that either one of the environments or the OLPP system itself haven't changed enough to require modification to these routines. Since the system will probably never be commercially available on a MULTICS system, it made little sense to try the various cost and performance tests which should precede a marketing effort. In addition, a generalized software package should be developed with the active

participation of several test users. This allows algorithms to be tested extensively for consistency, sensitivity and performance. Since this thesis was an academic exercise within an academic environment, the more commercially viable, but time consuming and costly, user tests were not feasible.

The OLPP system was designed as a part time academic project, rather than as a full time commercial one. I believe that it represents a sound foundation for becoming a viable production planning tool for small custom job shops.



### 5.3 Future Extensions

The OLPP system was designed to be an integrated production planning system. The functions which it performs were chosen for their importance to a low cost production planning system which would constitute the first computer experience for small custom job shops. This criterion was chosen so that the OLPP system would be a well integrated and efficient production planning system, which would constitute a realistic alternative to the manual methods of planning now used in many small companies. Being a computerized system based on standard manufacturing practices, it could clearly outperform a manual system, but this alone is not sufficient criteria for a small company to adopt it. Instead, such companies weigh far heavier the questions 'What will it cost me?' or 'How will it increase my profits?'. Unless these can be answered positively with concrete facts, a small low margin enterprise could not risk using the OLPP system. So OLPP choose to implement only the smallest and most economical subset of production planning functions.

Once the basic system is operating is operating, there are several potentially worthwhile options which could be added to it. The first of these is a wider variety of inventory control techniques, such as order point or safety stock maintainance. This will make system more flexible and capable of more internal inventory control. A second addition might be the addition of report generator which could accept

inputs , which would control the forming of new reports. Although this is not critical to individual companies, it would make the adaption of the OLPP system to different companies easier. Another straightforward addition could be the generation and control of purchase orders. This would require the addition of purchase order and vendor files but would integrate very cleanly with the present control of manufacturing orders. The system could have the capability to choose vendors and lot sizes which would take optimum advantage of various price structures. It could also handle the job of preparation of the actual purchase orders and might even automatically remind vendors of future due dates and of overdue shipments. Finally, it could also have the capability of monitoring vendor performance.

Two additional features which would drastically enlarge the scope of the system would be financial accounting and job shop scheduling. cost reports and even accounts receivable or accounts payable information. The OLPP system could even be integrated or co-ordinated with a computerized accounting system, which might include such functions as payroll, or generation of income statements and balance sheets. Job shop scheduling involves large amounts of data handling but it is a logical extension to any master scheduling system. These last two possibilities would drastically change the role of the OLPP system and yet they demonstrate that OLPP might be the first step towards an interactive computer utility geared

toward small manufacturing companies. The potential of computerized systems such as these is vast and in future years our understanding of them should increase, while technological costs should decrease and make them a more widely used tool.

BIBLIOGRAPHY

1. Baker, C. T., "Fabrication and Assembly Operations"  
IBM Systems Journal, 4, April 1965, pp 86-136
2. Beard, "Practical Approach for Job Shop Planning and Control", Automation, 15, March 1968, pp 80-88
3. Bill of Materials Processor, White Plains, N.Y., IBM  
GE20-0114-1, 1969
4. Booz, Allen and Hamilton, "The Computer's Role in the Manufacturing Industry", Computers and Automation, 15, Dec. 1966
5. Buffa, Elwood, "Production - Inventory Systems - Planning and Control", Homewood, Illinois, Richard Irwin & Co., 1968
6. Byrd, J., and Siemans, N., "Production Control Based on the Bill of Materials Processor", Industrial Engineering, 3, July 1971, pp 22-33
7. Canning, Richard, "Application Packages Coming into their Own", EDP Analyser, 6, July 1967
8. \_\_\_\_\_, "Creating the Corporate Data Base",  
EDP Analyser, 8, Feb. 1970
9. \_\_\_\_\_, "Data Management: File Organization",  
EDP Analyser, 5, Dec. 1967
10. \_\_\_\_\_, "Fast Response System at Rohr",  
EDP Analyser, 6, Sept. 1968
11. \_\_\_\_\_, "New Management Reporting System",  
EDP Analyser, 5, Jan. 1967
12. \_\_\_\_\_, "New Dimensions in Management Control",  
EDP Analyser, 6, Feb. 1968
13. \_\_\_\_\_, "Organizing the Corporate Data Base",  
EDP Analyser, 8, March 1970
14. \_\_\_\_\_, "Trends in Data Management", EDP Analyser  
9, May 1971
15. \_\_\_\_\_, "Using Shared Data Files", EDP Analyser,  
8, Nov. 1970

16. Church, F. L., "Requirements Generation, Explosions and Bills of Material", IBM Systems Journal, 2, Oct. 1963 pp 268-287
17. Clements, D., "Information Flow, Control Systems and Management", Management International Review, 1, Aug. 1968, pp 19-31
18. Dicky, "EUREKA, A Unique Approach to Scheduling", Industrial Engineering, 2, May 1970, pp 59-61
19. Dodd, G., "Elements of Data Management Systems", Computing Surveys, June 1969
20. "EDP Hustles the Huffies out of Hoffman", Business Automation, May 1970
21. Johanson, H., and Gargiulo, G., "Maximizing Production from Existing Capacity", Management Services, 7, Jan. 1970 pp 23-30
22. Gere, W., "Heuristics in Job Shop Scheduling", Management Science, 12, Nov. 1966
23. Grossman, "Using a Computer to Control Production", Automation, 15, Oct. 1968, pp 60-65
24. Hammerton, "Cost Determinents in Designing Production Control", Automation, 14, Nov. 1967, pp 92-97
25. Holstein, W., "Production Planning and Control Integrated" Harvard Business Review, 46, May 1968, pp 121-140
26. Hurst, E. G., and Mc Namara A., "Heuristic Scheduling in a Wollen Mill", Management Science, 14, Dec. 1967, pp B182-B203
27. Jordan, H., "Check List for Production & Inventory Control", Management Review, 59, Aug. 1970, pp 38-43
28. Korn, K., and Lamb J., "Production & Inventory Control by Computer - A Universal Model", Computers and Automation, 16, May 1967, p 23
29. Malloy, J. P., "Computerizing the Management Cost System in a Small Production Shop", Computers and Automation, 16, July 1967, pp 18-21
30. Modie and Novotny, "Computer Scheduling & Control Systems for Discrete Part Production", Journal of Industrial Engineering, 19, July 1968,

pp 336-341

31. Nathan, R. V., "Due Date Scheduling by Computer", Data and Control Systems, 4, Oct. 1966, pp 34-40
32. Nicholson, T., and Pullen, R., "The Design of the SPUR Production Scheduling Program", The Computer Bulletin, Nov. 1970, pp 381-386
33. Opler, A., "Information Retrieval", Business Management, 28, June 1965, pp 36-40
34. Pelham, R., "Putting Information Systems into the Company Control Structure", Data Processing Magazine, July 1970, p 23
35. "Planning and Scheduling Steelworks", Journal of the Iron and Steel Institute, Aug. 1967
36. "Planning for Production Control", Journal of Accountancy, 128, Aug. 1969, pp 82-84
37. "Production and Inventory Control Outlook", Systems, 8, Sept. 1967, pp 12-45
38. The Production and Inventory Control System, White Plains, N.Y., IBM, GE20-0280-2, 1969
39. "Totality and Particularity in Production Planning", Journal of Accountancy, 130, Aug. 1970, pp 82-83
40. Umstead, D., "Are You Overlooking a Cool Million in Your Factory?", Business Management, 39, Dec. 1970, pp 17-19
41. Van De Mark, R. L., "Controlling Production & Inventory in Small Plants", Foundry, May 24, 1967, p195
42. Wiest, J., "Heuristic Programs for Decision Making", Harvard Business Review, 44, Sept. - Oct. 1966
43. Willis, C., "Computers in Production Control", The Tool and Manufacturing Engineer, Aug. 1968