GH20-1260-0

Program Product

Information Management System Virtual Storage (IMS/VS) General Information Manual

IBM

GH20-1260-0

Program Product

Information Management System Virtual Storage (IMS/VS) General Information Manual

Program Number 5740-XX2

IMS/VS is a control system that has been developed to improve the computer user's ability to implement teleprocessing and/or batch-type data processing applications. Its development is within the framework of Operating System/Virtual Storage.

This manual includes a general description of the system and its various facilities and programs, listings of typical and minimum configurations, and sample applications.

IBM

First Edition (January 1973)

This edition applies to Version 1, Modification Level 0, of the program product Information Management System/Virtual Storage (IMS/VS) (5740-XX2) and to all subsequent versions and modifications until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein. Therefore, before using this publication, consult the latest System/370 SRL Newsletter (GN20-0360) for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments has been provided at the back of this publication. If this form has been removed, address comments to: IBM Corporation, Department J11, 1501 California Avenue, Palo Alto, California 94304. Comments become the property of IBM.

CONTENTS

CHAPTER 1. INTRODUCTION

IMS/VS is a control system that has been developed to improve the computer user's ability to implement teleprocessing and/or batch-type data processing applications. Its development is within the framework of Operating System/Virtual Storage.

Today, companies are evaluating computer systems, not only with regard to programming systems and hardware, but in relation to the information needs of the total corporate environment. There are increasing demands for applications that interrogate and maintain large centralized information files. IMS/VS provides a number of features that facilitate implementation, change, and expansion of such applications and information files.

The use of IMS/VS can be considered pertinent to the needs of most corporations and institutions. Applications that might lend themselves to IMS/VS include payroll and personnel, manufacturing bill of material, inventory control, accounts receivable, hospital records, student records, petroleum well records, and demand deposit accounts systems. Using IMS/VS, a company can design its applications to interface with the information files from remote terminals, in the more conventional batch mode, or in combination.

These features, coupled with the ability to respond to frequent and anticipated high-volume information requests, make IMS/VS a powerful new tool for the data processing user.

CHAPTER 2. ENVIRONMENT

Prior to discussing IMS/VS, it is appropriate to describe the environment within which IMS/VS operates and to define terms and concepts that are used later in this manual.

DATA BASE

Traditionally, data files were designed to serve individual applications, such as inventory control, payroll, engineering drawing release, manufacturing planning, etc. Each data file was specifically designed with its own storage space within the computer, on tape or direct access devices. In many instances these data files included duplicate or redundant information. This information overlap would often result in one file being kept current while the other would remain static and fall out of date.

When the same data resided in different application files, it normally existed in different formats. This variance in the format of common data meant that application programs were tailored to specific data organizations and even specific physical devices. When new data management techniques and devices were introduced, the application programs normally had to be changed. Therefore, application programs could be in an almost perpetual state of change, adding appreciably to the overall cost of data processing.

These undesirable attributes of data files have been eliminated by the advent of the "data base". A data base is defined as a nonredundant collection of interrelated data items processable by one or more applications.

The data base provides for the integration or sharing of common data. As an example, a manufacturing company having an application for release of engineering part data may first integrate its data with an application dealing with a manufacturing part release (Figure 1). Subsequently, application data for assembly installation accounting may be integrated. Note that the data and the programs of the first two applications need not change when the data of the third application is integrated.

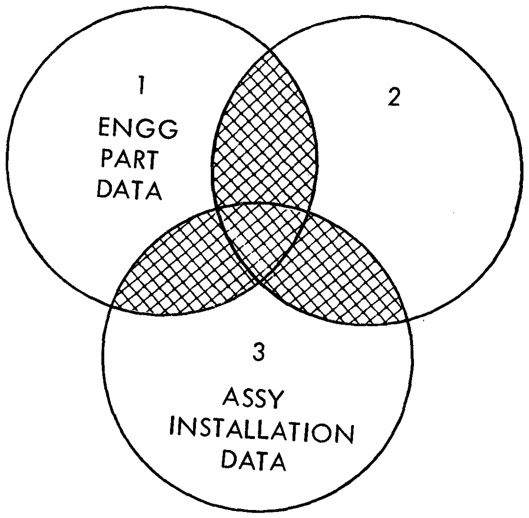


Figure 1. Application data integration--data base concepts

The data base provides flexibility of data organization. It facilitates the addition of data to an existing data base without modification of existing application programs. In Figure 1, the assembly installation accounting data may be added, when it is ready to be integrated, to the engineering and manufacturing data base. This independence is achieved by removing the direct association between the application program and the physical storage of data.

The advantages of the data base are:

- Elimination of redundant data and implied redundant maintenance
- Consistency through the use of the same data by all parts of the company
- Application program independence from physical storage and sequence of data
- Reduction in application costs, storage costs, and processing costs

TELEPROCESSING AND BATCH PROCESSING

In batch processing, single transactions are accumulated and processed periodically against the data base. The significant characteristic of batch processing is that of elapsed time. The use of batch processing should depend on how current the user's information needs to be and on the costs of alternative forms of processing.

Because the computer data base is not continuously available to the batch processing user, the information in the data base may not be up to date. However, the user of the information in a data base maintained by batch processing may not require current information at his disposal.

The user of information may acquire more nearly current data through teleprocessing. Remote terminals provide the user with the ability to enter transactions as "messages", allowing both inquiry and update capability to the data base. Data bases used for teleprocessing can also be used by batch processing programs to produce reports or to answer complex inquiries. Remote terminals may operate in a conversational or interactive manner where a complete terminal operation is represented by several interrelated transactions.

CHAPTER 3. GENERAL DESCRIPTION OF IMS/VS

IMS/VS extends the capabilities of the Operating System for Virtual Storage (OS/VS) in the data base and data communication environment. It is implemented as an OS/VS-authorized program and has four major objectives:

1. To provide data organization methods that are conducive to the creation, interrelation, and maintenance of large common data bases and the multiapplication use of these data bases

2. To provide the means to develop and maintain a data base system in the batch processing environment

3. To provide the ability to easily extend data base processing to the teleprocessing or data communication environment

4. To provide an efficient telecommunication subsystem to support the development of a high volume/rapid response online application system.

IMS/VS comprises two major components: (1) the data base facility and (2) the data communication facility.

DATA BASE FACILITY

The data base processing capabilities of IMS/VS are provided by a facility called Data Language/I. The data base functions supported are definition, creation, access, and maintenance. The full data base capabilities of Data Language/I can be used in the IMS/VS batch processing or teleprocessing environment.

DATA COMMUNICATION FACILITY

Data communication capabilities are characterized by the use of input/output terminals in remote and local environments, connected to the computer, which provide the user with access to the data base. The remote communication network consists of IBM 2780 Data Transmission Terminals, 1050 and 2770 Data Communication Systems, 2740 Model 1/Model 2 and 2741 Model 1 Communication Terminals, 2980 General Banking Terminal Systems, 3270 Information Display Systems, 33/35 Teletypewriters (ASR), 7770 Model 3 Audio Response Units, Card Reader/SYSOUT devices, System/3, and System/7. The local communication network consists of 3270 Information Display Systems. Card Reader/SYSOUT devices are supported by BSAM. All other terminals are supported by OS/VS - BTAM. This terminal and device support enables the system to receive and transmit a variety of message types for multiple applications. Terminals need not be dedicated to specific applications. A single terminal may be used by multiple applications.

System Definition

A system generation function called system definition allows the user to specify communication lines and terminals, message types, message classes, programs, and data bases particular to the user's environment. This information enables IMS/VS to tailor a system for efficient execution of message processing and message switching. Message processing may result in both data base inquiry and update activity. Conversational terminal operation is possible. A user-provided library of application programs for message and batch processing and a description of their data base requirements must be provided. These application programs may be written in any of the following programming languages: Assembler Language, COBOL, or PL/I.

HIGHLIGHTS OF IMS/VS

In order to provide insight into IMS/VS and its capabilities, the following list of highlights is provided:

1. IMS/VS is a general purpose system applicable to the diverse data processing requirements of many companies. It has been designed as an open-ended system, thus providing the ability to extend functions.

2. A data base capability provides improved access to, and maintenance of, data and provides an effective method for handling variable-length application data.

3. A means of cross-referencing or interrelating the data within two or more data bases facilitates nonredundant data storage.

4. Application programs are independent of the physical storage organization of data. Data may be accessed by IMS/VS for application programs through sequential, index, or direct access techniques.

5. A means is provided to restructure and expand an established data base without modifying existing application programs that use it.

6. Batch programs and teleprocessing or message processing programs may operate concurrently in the system.

7. Security capabilities are provided in the message processing environment to assist the user in ensuring that information is available only to those entitled to it and that only eligible persons may update the data base.

8. Checkpoint, restart, and data recovery capabilities are facilitated in both the batch and message processing environments through the use of the system log.

9. Statistical information is provided by the system to assist the user in evaluation of performance and of changing communication requirements.

10. The user of IMS/VS has the capability of structuring data bases, defining various applications, and tailoring the input/output terminal and data storage environment.

11. IMS/VS permits the evolutionary expansion of data processing applications from the batch environment to the teleprocessing environment. An application can initially utilize Data Language/I for data base batch-only processing. Once experience

is gained in data base batch-only processing and as the needs of the application dictate, the same data base and application program design may be used in a teleprocessing environment.

12. Data interchange between IMS/VS and existing application programs is facilitated by use of the OS/VS VSAM access method.

13. IMS/VS can process DL/I DOS/VS data bases.

14. DL/I DOS/VS can process IMS/VS data bases constrained to the DL/I DOS/VS functional subset.

SYSTEM DESCRIPTION

IMS/VS includes the following functional facilities:

- Data base facility

  Data Language/I

- Data communication facility

  Telecommunications

  Message scheduling

  Checkpoint

  Restart

- Utility programs

Data Base Facility

Data Language/I. Data Language/I is the data management facility through which users adapt IMS/VS to the data requirements of their own applications. An application program has two distinct interfaces with Data Language/I: (1) a data base description, the logical data structure of the data base given as a definition external to the application program; and (2) a common symbolic program linkage, which allows Data Language/I to process input/output requests during execution of the application program.

Data Language/I can be used to:

- Assist in the creation and maintenance of data bases

- Promote integration of applications

- Reduce application program maintenance caused by changes in the data requirements of the application user

- Accomplish data storage and access through sequential, index sequential, or direct storage organizations.

Data Communication Facility

In the teleprocessing environment of IMS/VS, Data Language/I also provides the interface for input and output terminal messages. This terminal message interface is the same as that used for all data base requests.

Telecommunications. IMS/VS supports IBM 1050, 2740 Model 1 and 2, 2741 Model 1*, 2770, 2780, and 2980 communication terminals, 7770 Model 3* Audio Response Units (in conjunction with a Touch-Tone** Telephone (or equivalent) and the IBM 2721 Portable Audio Terminal), the 3270 Information Display System, the 33/35 Teletypewriter (ASR)*, Local Card Reader/SYSOUT device support, System/3***, and System/7*** to be used for message input and output. Depending upon the application requirements specified by the IMS/VS user, input messages may cause a program to be scheduled to process the message or may be switched to another terminal. One of the 1050 or 2740 station-controlled terminals operates as the master terminal of the system and provides the user with a control center. The master terminal controls checkpoint/restart initiation, user terminal operation, and input message processing. Conversational operation between terminals and one or more message programs is possible. A "scratchpad" area, either main storage or direct access storage, is provided with IMS/VS for retention of information during a conversation. A video terminal paging capability is also provided.

Message Scheduling. IMS/VS initiates execution of message processing programs based upon message types received within class. All messages acceptable to the system are predefined and verified through a one- to eight-character code in the first segment of a message. When a valid message is completely received and queued, its presence is made known to message scheduling. Messages are selected from the queue by priority within message class. When the required resources for message scheduling are available, processing is initiated.

The scheduling of two or more message programs that might concurrently update a common data base is permitted, even though it (they) may update the same segment type. Segments are described in Chapter 4 of this manual.

Checkpoint. Periodic checkpoints of IMS/VS are required in order to provide the ability to restart after loss of main storage, direct access storage message queues, or data base information. There are many conditions under which IMS/VS may require that a checkpoint be taken. These conditions, in general, can be grouped into four classifications:

- System-scheduled checkpoints based upon message volume
- Master terminal request to checkpoint the system

---

*Because IBM does not provide error-detection capability for the 7770-3, 2741, or the 33/35 Teletypewriter (ASR), these devices are supported for INQUIRY-only transactions, even though the non-INQUIRY (update) capability exists.

**Registered Trademark of American Telephone & Telegraph Co.

***Since IMS/VS provides no code resident in the System/3 or the System/7, a significant amount of programming must be done by the user to attach to the IMS/VS System/3 or System/7 support.

- Master terminal request to orderly terminate the system

- Master terminal request to produce a current copy of the data from a data base

Checkpoint and restart capabilities are provided for both batch and online data processing.

Restart. IMS/VS can be stopped and restarted daily or at explicit intervals. Restart provides for system reconstruction after a controlled stop, an emergency stop, or a data base destruction.

The online checkpoint and restart functions are dependent upon message queuing on direct access storage and the recording on the system log of all messages and data base modifications.

A means is also provided for batch programs to coordinate their own checkpoints with the IMS/VS log tape, thus allowing batch jobs to take advantage of any checkpoint/restart facility available to other OS/VS jobs.

Through program isolation, a means is provided to ensure data base integrity after IMS/VS system failure with restart and message or batch message program failure.

Utility Programs

The following utility programs are provided:

- System definition - Structures control blocks used by IMS/VS to define the particular user's data processing environment and selects executable code based upon user's IMS/VS processing requirements.

- Security maintenance - Creates control blocks used by IMS/VS that describe a particular user's data processing security requirements.

- System log analysis - Produces statistical reports concerning message type and terminal operation.

- Program specification block generation - Creates control blocks that identify the characteristics of the terminals and data bases to be used by a particular application program.

- Data base description generation - Creates the control blocks required to describe each data base.

- Application control blocks maintenance - Uses the output of program specification and data base description generations to create and maintain the control blocks in a form directly usable by the IMS/VS system.

- Data base load and reorganization - Provides a generalized program for assistance in the creation and reorganization of a data base.

- Data base dump/restore - Provides an efficient dump/restore program for data bases. This utility is to be used typically for assistance in data base reconstruction.

- Data base recovery - Provides the means for collection and application of data base modifications that were previously recorded on the system log and are to be used in data base reconstruction.

- Format language - Permits the user to define message formats and their associated display formats for 3270 Terminals. The format language will provide the application programmer with the capability to cause the transmission of a message between a logical terminal and an application program without being concerned about 3270 device characteristics.

CHAPTER 4. IMS/VS SYSTEM CONCEPTS

While a general description of IMS/VS has been given, there are numerous technical considerations that require additional discussion. This chapter discusses those considerations that are of particular interest to personnel responsible for planning the use of IMS/VS.

IMS/VS operates as an authorized program under the operating system. The teleprocessing capabilities require Operating System/Virtual Storage 1 (OS/VS1) or Operating System/Virtual Storage 2 (OS/VS2). The data base processing capabilities of IMS/VS, which are provided by Data Language/I, can operate independently for batch processing under OS/VS1 or OS/VS2, or as part of the IMS/VS teleprocessing environment.

CONTROL FACILITY

The initiation and control of the various IMS/VS facilities are provided by the IMS/VS control facility. The control facility is executed in part as SVC (Supervisor Call) routines added to the operating system and in part as one or more jobs under the operating system with high scheduling priorities. Once loaded, the control facility performs the following functions:

1. Initiation and control of all facilities within the IMS/VS control program:

   a. Telecommunications

   b. Message scheduling

   c. Data Language/I

   d. Checkpoint

   e. Restart

2. Initiation and control of each operating system region utilized by application programs for message processing.

   Initiation is performed through the services of the operating system job management routines. Subsequent scheduling, loading, and execution of message processing programs are performed within an existing message processing address space. This approach allows each message processing program executing concurrently to operate in a unique address space. The use of main storage protection facilitates evolutionary development of a multiapplication environment.

3. Communication between operating system address spaces, (regions, partitions, or memories) that have been initiated for message processing and the operating system address space containing the IMS/VS control program. This interaddress space communication allows scheduling the execution of message processing programs and the handling of data base requests from message processing programs. This ability to communicate is provided through supervisor call routines and the asynchronous scheduling facility of the operating system.

4. Provision for a function designated program isolation. The intent of program isolation is to provide the ability for two

or more user programs to use a data base concurrently. When a user program updates a particular occurrence of a segment, no other user program is permitted access to that segment until the updating program completes. If a program should ABEND after updating but prior to normal completion, the updates it performed are backed out, leaving the referenced data base intact as if the user program had never been scheduled.

5. Execution of the system logging functions necessary for proper restart of the system.

6. Through implementation of the operating system STAE and SPIE functions, an attempt to minimize the impact of any failure to the IMS/VS environment. Of greatest importance is the integrity of the IMS/VS system log necessary for execution of system restart.

TELECOMMUNICATION FACILITY

The telecommunication facility is the interface between communication terminals and the remainder of the IMS/VS system, providing such functions as the initiation and control of all input/output operations on communications lines and the enqueuing of input and output messages cn a direct access storage.

Physical Terminals

Physical terminals are the hardware devices used to enter or record messages being sent or received over communications lines. Within the IMS/VS environment, physical terminals may be permanently attached to leased communications lines or may operate in an AutoAnswer mode with respect to switched communications lines. IMS/VS provides communications support for the following terminals:

```
1050     (including components) (nonswitched or switched line)
2740     Model 1 (nonswitched or switched line)
2740     Model 2 (nonswitched line)
2741     (nonswitched line)
2741     (switched line)
2770     (nonswitched line) (multipoint only)
2780     (nonswitched line)
2980     (nonswitched line)
33/35    Teletypewriter (ASR)
Touch-Tone Telephone (or equivalent)}  {in conjunction with
2721 Portable Audio Terminal        }  {7770 Model 3
Local Card Reader/SYSOUT
3270     (remote and local mode) (nonswitched line)
System/3 (nonswitched line) (multipoint only)
System/7 (nonswitched line)
```

In addition, the architectural modularity of the telecommunication facility assists the user in the design and implementation of teleprocessing support for other devices.

Local Card Reader/Sysout support is provided with BSAM. All other terminal support is provided through BTAM.

Logical Terminals

A logical terminal is a name that is related to a physical terminal. One physical terminal can have one or more logical terminals associated with it. The user of IMS/VS references the logical terminal in the construction and transmission of messages. The user is never concerned about such things as physical terminal addresses. If a physical terminal becomes inoperative, the logical terminals associated with that physical terminal can be dynamically reassigned to another physical terminal, thereby reassigning output queues of messages to another physical destination. Each logical terminal can have unique security parameters associated with it.

Master Terminal

The master terminal is a logical terminal that acts as the operational hub of IMS/VS. The physical terminal with which it is associated must be a nonswitched IBM 1050 or 2740 Model 1 with Station Control Feature. A 2740 Model 1 without the Station Control Feature, although supported by IMS/VS, should not be designated as the master terminal. The master terminal has complete control of IMS/VS communications facilities, message scheduling, and data base operations. It is used for checkpointing and restarting the system, for continuous monitoring of the system, and for dynamically altering the operation of the system. In case of master terminal failure, the operating system console can be used as an alternate master terminal. Since the master terminal is a logical terminal, its status may be dynamically reassigned to another physical terminal.

Input Messages

IMS/VS processes three different kinds of messages. The kind of message is determined by the first one to eight characters of the first segment of the message. The initial character string specifies the destination of the message text that follows.

1. If the message is a transaction, the destination is called a transaction code. It identifies the application program that is to process the message.

2. If the destination is the name of a logical terminal, terminal-to-terminal message switching is accomplished.

3. If the first character of the destination is a slash (/), the message is a command, the function of which is to dynamically interrogate or alter the operation of the IMS/VS system.

Message Queuing

All input and output messages, whether online message processing or message switching, are written to a system log. They are queued in main storage with direct access storage backup as required. These functions are performed to ensure an ability to restart after a system failure. For improved performance, long and short messages can be queued separately on multiple direct access data sets. The space in the message queue data sets is reused when it is no longer required for the previous message.

Message Editing

IMS/VS provides a standardized message editing facility and provides the capability to add user-defined editing routines. The following editing capabilities are available:

1. Standard input edit. This editing routine accomplishes such generalized functions as backspace editing and the removal of control characters from input messages.

2. User input edit. User exits are available, to be defined at system definition time, for input editing by transaction name. The user may provide a separate edit routine for each transaction code defined within the IMS/VS system.

3. User output edit. The user may provide one logical terminal edit routine for operation within the IMS/VS system. The activation of an exit to this routine for a specific logical terminal is defined at system definition time. This edit routine is invoked to edit message-switch messages if the exit has been activated with respect to the destination logical terminal.

4. User device output edit. The user may provide an output editing routine for each type of device defined within the IMS/VS system. The activation of an exit to the specific device-oriented edit is definable relative to a specific physical terminal at definition time.

Conversational Processing

Conversational processing provides the user with the ability to retain message continuity from a given terminal even though his program is not retained in main storage residence in a message address space. The initial conversational transaction code must be supplied by the terminal operator. Thereafter, only data is required from the terminal during the conversation. The message destination has been predetermined by the transaction code in the first piece of the conversation. Once the initial transaction of the conversation is scheduled and the message processing program given control, output messages may be directed to the source terminal and to a scratchpad area. The scratchpad area may be composed of main storage within the IMS/VS control program or direct access storage. The content of the scratchpad area would typically be composed of information from the terminal and from data bases to be saved for continuing the conversation. One scratchpad area exists for each terminal concurrently operating in conversational mode. Subsequent entry from a terminal already operating in conversational mode causes the scheduled message processing program to receive both the contents of the scratchpad and the input terminal data. For purposes of program isolation, each input message is considered as an individual user unit of work.

A terminal command is provided to enable the terminal operator to end a conversation prior to the normal completion of the message processing program. Commands are also provided to enable the terminal operator to temporarily suspend and to save an incomplete conversation and to resume that same conversation at a later point in time.

Video Terminal Paging Feature

A video terminal paging feature is provided to allow the application programmer to output multiple screens of information to a video device which may then be viewed by the terminal operator either in or out of sequence and as many times as he chooses. The capability is also of skipping over one or more screens, and of returning later to view them. The only restriction IMS/VS imposes is that the operator may not return to a series of images after notifying IMS/VS that he has completed viewing them and has moved on to a new message or series of screens.

Security

IMS/VS provides the capability to verify user-defined security requirements with respect to transaction and command messages. At the user's option, two types of security verification may be designated: terminal security and password security. Terminal security ensures that a secured transaction or command may be entered only from specific, designated terminals. Password security ensures that a transaction or a command message will not be processed unless a user-defined password is appended to the transaction code or to the command verb.

Non-IMS/VS operational access to data base(s) must be secured by the user through his own operational policy and procedural controls.

Master Terminal Commands

Master terminal commands are provided for the purpose of dynamically interrogating or altering the processing functions of IMS/VS. The following constitutes a summary of some of these functions:

- Starting, stopping, or otherwise modifying the system functions of message receiving, queuing, scheduling, and sending
- Allowing the IMS/VS system to purge its message queues prior to shutdown
- Allowing the temporary halt of transaction processing, message processing program scheduling and execution, and data base usage
- Starting and stopping message processing address spaces
- Assigning and modifying message processing address space classes
- Modifying message priorities and classes
- Initiating and controlling system checkpoint and restart
- Assigning logical terminals to physical terminals
- Modifying passwords
- Modifying password and terminal security provisions
- Displaying the status of various control blocks related to transaction types, programs, data bases, message queues, and communications facilities

Remote Terminal Commands

Remote terminal commands are provided to change the status or mode of operation of the user's terminal, to provide extended security facilities, and to provide extended user message entry or data output facilities. The following constitutes a summary of some of these functions:

- Associating one or more logical terminals with a switched communications line, similar to a signon function
- Locking or unlocking system resources, such as the user's physical or logical terminal
- Providing the ability to terminate, to save, or to restore a conversation
- Presetting the destination of all messages subsequently entered into the user's terminal
- Broadcasting a message to all or selected logical or physical terminals
- Canceling a partially entered input message
- Writing informational messages to the system log
- Formatting a 3270 video screen for data input
- Placing a terminal in a test mode for online diagnostic purposes
- Placing a terminal into exclusive mode so that only responses to messages entered into the terminal will be transmitted back to the terminal
- Displaying the identification of the master terminal

Test Mode

Facilities are provided to allow for the testing of a terminal in an online environment. A user may place his own terminal or another terminal into one of the test modes. The simple test mode ensures that any input message entered into the terminal under test is transmitted back to the test terminal, with error analysis procedures being bypassed. The loop test mode provides for the establishment of an output write loop, whereby a a user-entered message is continuously transmitted to the test terminal. Appropriate commands are provided to remove a terminal from the simple test mode or the loop test mode.

MESSAGE SCHEDULING FACILITY

Separate operating system address spaces (regions, partitions, memories) are used for message processing. These address spaces are initiated through the normal operating system job management routines during IMS/VS initialization or by an IMS/VS master terminal command during IMS/VS execution. Each message processing address space may have unique characteristics such as size, priority, and classes of messages to be processed in the region. These characteristics define the class of the message processing address space. Subsequently, message processing programs are loaded into and executed within these address spaces based upon input messages received. The message scheduling facilities of IMS/VS initiate message processing program load and execution through resident supervisor call routines added to the operating system nucleus.

The IMS/VS input message scheduling algorithm is controlled by the system user. The user must provide four parameters at the time he describes each message type:

1. Normal priority. The priority at which messages of this type are normally processed. This may be priority level 0 through 14.

2. Limit count. A number ranging from 1 to 65,545. When the count of messages of this type in the input queue (queue count) is equal to or greater than the limit count, the normal priority is raised to the limit priority (below).

3. Limit priority. A number ranging from 1 to 14. When the limit count is equal to the queue count, the normal priority is raised to the limit priority until the queue count returns to zero. At that point, the normal priority is restored.

4. Message Class. A number ranging from 1 to 255. The message class determines the message processing address space into which the transaction type will be scheduled.

An example of the scheduling process is as follows:

- Transaction code = MTI
- Normal priority = 4 (level 4)
- Limit count = 20
- Limit priority = 11 (level 11)
- Class= 1

As soon as a message processing address space with a class of 1 is started, this transaction will compete for scheduling.

Assume this application requires a maximum of 10 minutes turnaround on all messages. The minimum message rate is 25 per 10 minutes. During normal working hours, message type MTI may be scheduled every minute or more often, and most of the messages are processed each time. During peak period when there is high activity on messages at levels 9 through 14, for example, messages at the lower levels may only receive service every several minutes or perhaps not until the peak is over. During these peak periods, message MTI will stay at level 4 without service until the twentieth message is enqueued. When the twentieth message arrives, the priority of MTI is automatically raised to level 11 by making the current priority equal to the limit priority (that is, 11). MTI will now contend for service at a priority of 11. MTI will remain at priority 11 until the enqueued message count returns to zero. MTI is then automatically restored to priority 4.

It is possible for the user to specify a normal priority of zero (null), and no processing occurs until the limit count is reached.

Data Language/I Concepts

Data Language/I provides application program independence from access methods, from physical storage organizations, and from the characteristics of the devices on which the data of the application is stored. This independence is provided by a common symbolic program linkage and by data base descriptions external to the application program. A reduction in application program maintenance should be realized. The section entitled "Data Base User Interface" provides an extended definition of this interface.

Data Language/I provides for elimination of redundant data while assisting in the integration or sharing of common data. The majority of the data utilized by any company has many interrelationships that can cause significant redundant storage of data if conventional organizations and access methods are used. For example, Manufacturing and Engineering have many pieces of data that would be useful to Quality Control. The storage organizations and access methods employed by Data Language/I facilitate data integration with a minimum of data redundancy. The use of secondary index data base allows processing of data in more than one sequence. However, if analysis of a customer's data shows that all the data cannot be placed in a single common data base, Data Language/I allows the user the additional capability of physically structuring the data over more than one data base. Before Data Language/I, personnel responsible for application programs frequently were not able, nor did they have the time, to integrate other data with their own to eliminate redundancies without the necessity of a major rewrite of the application programs involved.

An important capability of Data Language/I that protects each application of a multiapplication data base is the concept of data sensitivity. When operating against a Data Language/I data base, only the data that is predefined as sensitive is available for use in a particular application. Each application using the data base can be sensitive to its unique subset of data. Where an application has defined "sensitivity" to a subset of the data within a data base, modification and addition of nonsensitive data do not affect the processing capability of the application. In addition, any application can be restricted as to the types of data base requests made against its sensitive data.

Environment Definitions

Within the Data Language/I environment, the following definitions apply:

- Segment. A data element of defined length, containing one or more logically related data fields. A segment is the basic data element that interfaces between the application program and Data Language/I and upon which the user defines his sensitivity. A segment may be defined as either fixed or variable in length.

- Logical data base record. A set of hierarchically related segments of one or more segment types. Each segment type may have a unique format. As viewed by the application program, the logical data base record is always a hierarchical tree structure of segments.

- Logical data base. The major unit of data storage under Data Language/I--a set of logical data base records stored in the Data Language/I organizations and accessible by any one or more of the

Data Language/I access methods. A data base is typically composed of one or more operating system data sets.

No attempt is made at this point to relate the logical data base record or logical data base to a physical storage organization or access method technique, which are presented subsequent to the section entitled "Logical Data Structures".

Data Independence

Under Data Language/I, the data base concept gives the user independence of his data and program from access methods and storage organizations. Physical storage is accomplished through the use of two unique Data Language/I storage organizations: hierarchical sequential and hierarchical direct. Four Data Language/I access methods--Hierarchical Sequential Access Method (HSAM), Hierarchical Indexed Sequential Access Method (HISAM), Hierarchical Direct Access Method (HDAM), and Hierarchical Indexed Direct Access Method (HIDAM)--are provided to allow access to these organizations. These storage organizations and access methods are discussed under "Data Base Organization and Access Methods". The application program interface with these two organization types and four access methods is totally symbolic. The application program is typically insensitive to the particular organization or access method.

To provide this independence, three definitions are required prior to the use of a data base by a program:

- The segments within a logical data base record to which a program wishes to be sensitive
- The logical data base record structure represented by one or more segments from one or more physical records
- The data base organizations and access methods

Declaration of segment sensitivity and logical data base record structure, separate from the application program, is used to define the program's "view" of a Data Language/I data base.

Logical Data Structures

Application programs written to use IMS/VS deal with logical data structures. _Logical_ refers to the manner in which the application program _sees_ the data. A logical data structure is always a hierarchical structure of segments. Programs written to process logical data structures can be independent of the physical data structure. _Physical_ refers to the manner in which the data is stored on a tape or direct access storage device. An application program user never deals directly with a physical data structure.

Most data processing information, regardless of industry, can and should be viewed in a logical data structure. Payroll/personnel is chosen here for explanatory purposes because of its commonality. Additional examples from a variety of industries are presented in Chapter 10.

Concept of Segment Interrelationships. The traditional manner of depicting data can be seen in Figure 2.

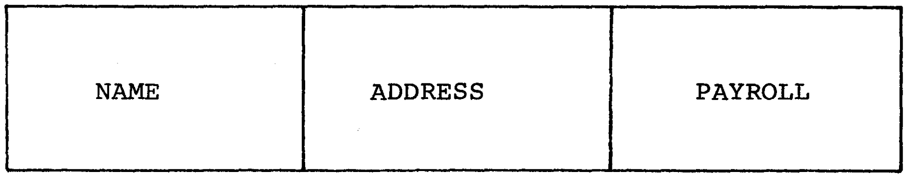


Figure 2. Traditional record layout

This picture describes the physical makeup of the record as it might appear on tape or on a direct access storage device. Each of the three divisions (name, address, and payroll) is referred to as a field. These fields usually contained more basic data elements. For example, one of the data elements in the payroll field would be rate of pay. In addition, the record might actually contain multiple address and payroll fields for a single name. This is typical if address and payroll history are desired. This same data record appears in Figure 3 as an IMS/VS logical data structure.

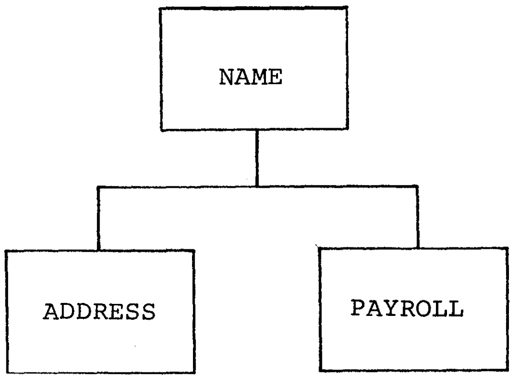


Figure 3. Hierarchical record layout

The name, address, and payroll fields are now considered segments of information. Each segment of information is considered to be made up of fields. Rate of pay is a field within the payroll segment.

The logical data structure in Figure 3 represents a hierarchical relationship. Data relationships described by this hierarchical picture have only one segment at the first level in the hierarchy but may have multiple segments at subordinate levels in the hierarchy (for example, multiple address and payroll segments for one name segment). Since each dependent segment in the hierarchy has only one parent or immediate superior segment, the logical representation is sometimes called a tree structure.

In Figure 3, the name segment with its associated address and payroll segments constitutes a logical data base record.

Through the concept of program sensitivity, IMS/VS allows a program to be written in such a manner that it sees only those segments of information that are relevant to the processing being performed. For example, a payroll program could be written to see only the name and payroll segments of the logical data base record shown in Figure 3. The program need not be aware of the existence of the address segment.

The IMS/VS data base capabilities allow for handling hierarchically related logical data structures of considerable variance. The maximum number of segment types is limited to 255 per logical data base record. A maximum of 15 segment levels can be defined in a logical data base record. Figure 4 represents an example of a logical data structure for a skills inventory.

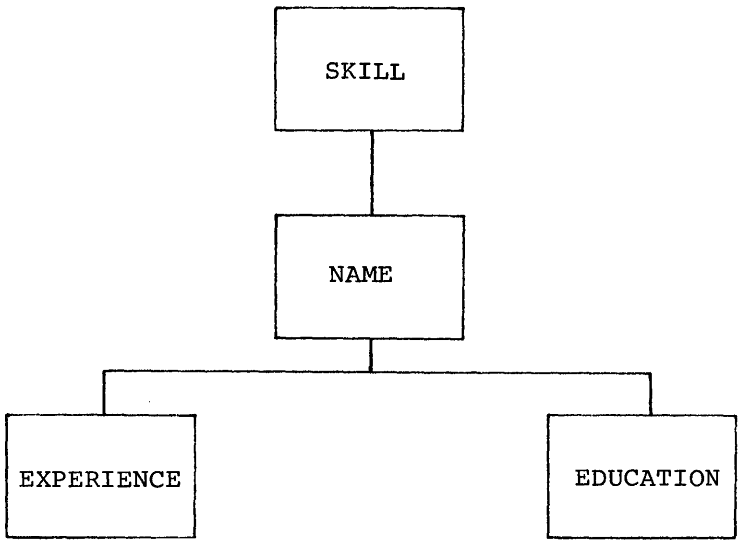


Figure 4. Logical data structure

The structure consists of four segment types: skill, name, experience, and education. The logical relationship shows that the name segment is a dependent segment of the <u>root segment</u>, skill, and that experience and education are dependents of the name segment. The root segment is the highest in the hierarchy. A <u>dependent segment</u> relies on some higher level segment for its full meaning or identification. Since the number of employees (name segments) may vary from one skill classification (skill segment) to the next, it is necessary for logical data base records to vary in size by the number of segments occurring in the hierarchy for the data structure.

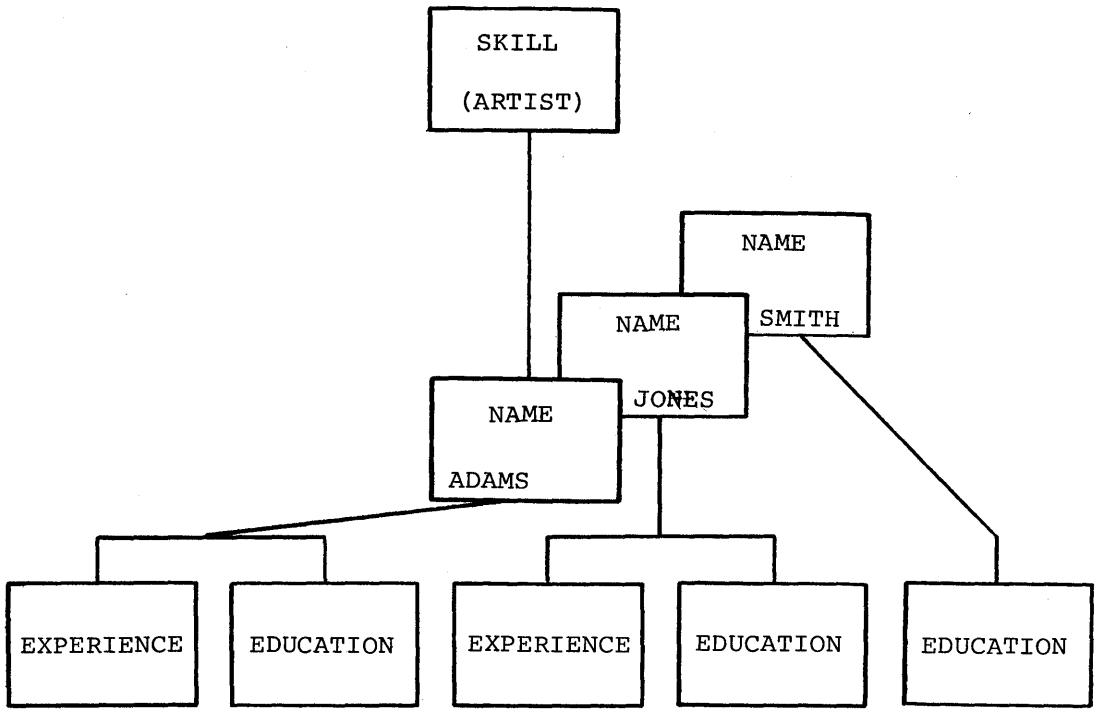


Figure 5. Logical data base record structure (derived from Figure 4)

Figure 5 shows a specific logical data base record from the logical data structure described in Figure 4. The root segment contains the specific skill of artist. There are multiple name segments. In this case, there are three employees who have some capability as artists: Adams, Jones, and Smith. Adams and Jones have both experience and education relating to this skill, while Smith has education only. Although the data base can be described as having four segment types, this particular data base record consists of nine segments because of the multiple occurrences of the name, experience, and education segments.

The segment types immediately above and below a given segment type are called parent and child segment types respectively. In Figure 5, the skill segment for artist is the parent of three name segments. Each name segment is a child segment of artist. The experience and education segments subordinate to a particular name segment are its children. All occurrences of a particular segment type within a logical data base record are called twin segments. The three name segments for Jones, Smith, and Adams are twins of each other.

Interrelated Tree Structures. A segment of information need not be a part of only one logical data structure. The physical storage techniques of IMS/VS allow a segment of information to participate in more than one logical data structure. By allowing a segment to exist only once yet participate in a number of logical data structures, IMS/VS allows elimination of redundant data and facilitates processing requirements. For example, the two logical data structures described previously could be related as shown in Figure 6. The name segment in the skills inventory data base has been replaced by a pointer, which directly addresses the name segment. The name segment is said to be a target; that is, it is pointed at by one or more other segments.

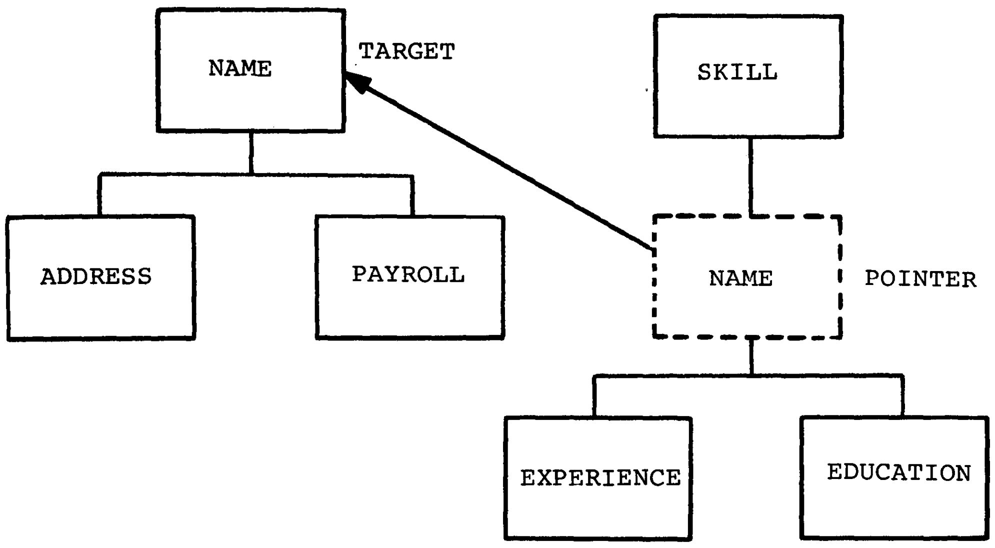


Figure 6. Interrelated tree structure (derived from Figures 4 and 5)

Figure 6 shows only the most basic form of direct address relationship capability. Direct address relationships may be bidirectional. The target, which is the name segment, can have a relationship back to all skills for a particular name. Figure 6 shows a logical data structure composed of the two interrelated trees. The interrelationship is accomplished by the <u>pointer segment</u>--<u>target segment</u> concept. We can now redraw this segment structure as a logical data structure in the form of one simple hierarchical tree, Figure 7.

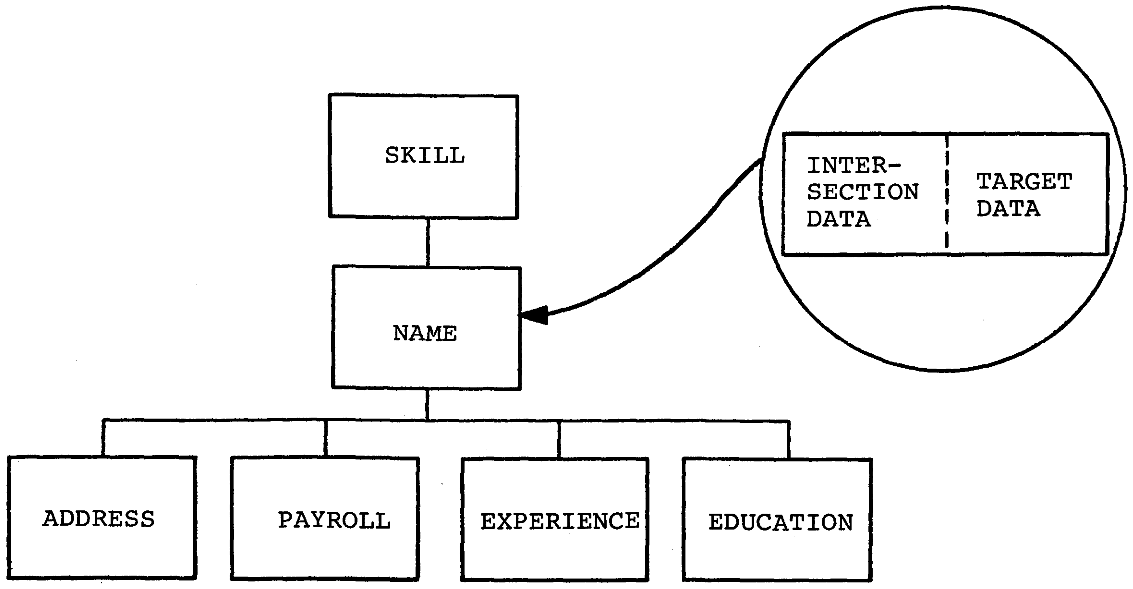


Figure 7. Simple hierarchical tree structure

An application program should always view the logical data structure as a simple hierarchical tree structure as shown in Figures 3, 4, and 7.

Intersection Data. In addition to having a pointer, a pointer segment may also contain user data. The data contained in a pointer segment is said to be intersection data. It is data unique to the relationship between a specific pointer segment and a target segment. The name segment as illustrated in Figure 7 may be thought of as the concatenation of the data in the target segment and the intersection data in the pointer segment. Alternatively, the name segment in Figure 7 may contain only the intersection data or only the target data. The data content of the pointer and target segments may be retrieved and modified independently. Alternatively, a specific pointer-target segment relationship may be retrieved and modified as one concatenated segment.

Inverted Data Structures. Inverted structures are facilitated with Data Language/I. For example, a logical data base record can be described to allow the accessing of a name segment in the payroll data base, Figure 6, and, from that, reaching all skill segments related to, or which point to, that name. This is accomplished by allowing the target segment to direct address backwards through all its associated pointer segments. (See Figure 8.) The seemingly complex data structure depicted in Figure 8 can be viewed as a simple hierarchical tree structure. Figure 9 illustrates the logical data structure in the hierarchical tree format. The intersection data in the pointer segment and the data in the skill segment may be concatenated.

The name segment--skill segment relationship can be inverted to allow accessing a skill segment and, from there, reaching all names related to that skill. (See Figure 7.)

Again, the application program should always view the logical data structure as a simple hierarchical tree structure as shown in Figures 3, 4, 7, and 9.

Secondary Index Data Bases. Additional flexibility is provided by designing secondary index data bases which may be used to present data in different sequences than the one in which it is physically stored and faster retrieval for search criteria which are indexed. For example, the NAME segment could be retrieved based on the locality of the person's address if an index were defined for that field in the ADDRESS segment.

Additional Environment Definitions

In addition to the environment definitions given previously, the following definitions apply within the Data Language/I environment:

- Logical data structure. A set of hierarchically related segments that serve as a prototype of a logical data base record. Only the segment types and not the number of occurrences of each segment type are illustrated.

- Logical data base record. A family of related segments described in the logical data structure but containing all related occurrences of each segment type. A logical data base record may exist as contiguous sets of segments within related physical blocks. In this case, the logical data base record is represented as one physical data base record. Alternatively, the logical data base record may be composed of segments from several physical data base records. The relationships between the physical data base records that represent the logical data base records are accomplished by direct address techniques or symbolic field values and indices.

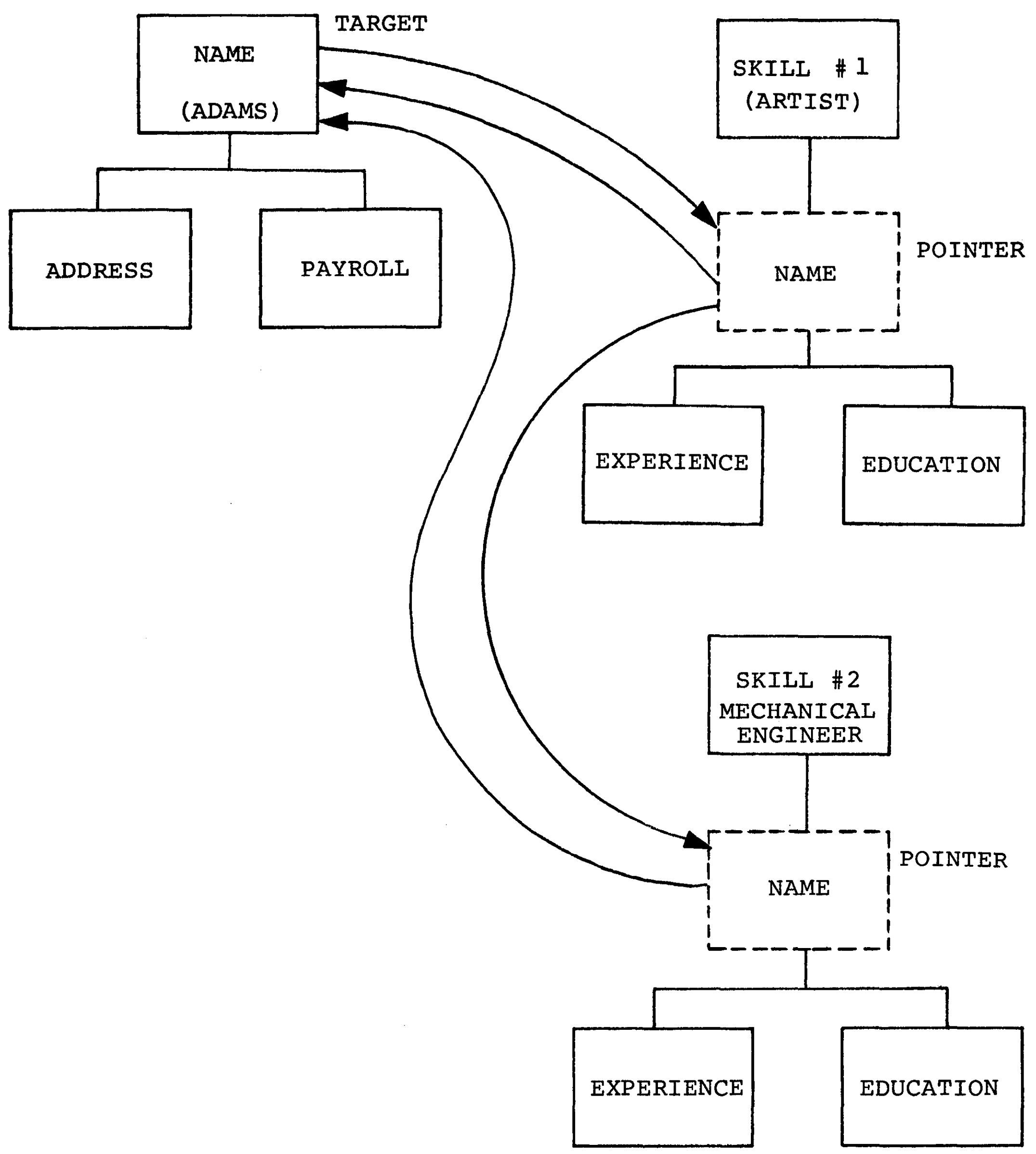


Figure 8. Complex data structure viewed as several interrelated tree structures

Data Base User Interface

A common symbolic program linkage and data base description allow the application program the ability to request Data Language/I to:

- Retrieve a unique segment (GET UNIQUE)
- Retrieve the next sequential segment (GET NEXT)

- Replace the data in an existing segment (REPLACE)
- Delete the data in an existing segment (DELETE)
- Insert a new segment (INSERT)

An input/output operation may be performed on a single segment, some segments, or all segments in a hierarchial path. Segment retrieval is based upon position in the data base and/or upon one or more comparisons on fields within a segment.

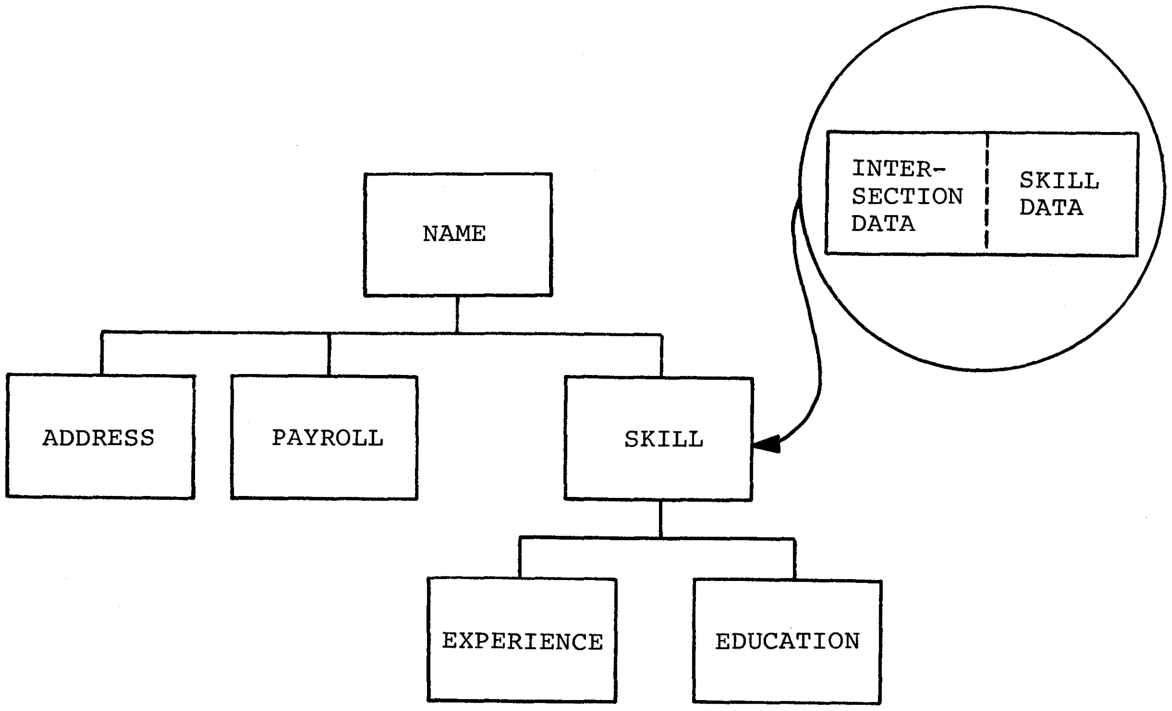


Figure 9. Logical data structure in hierarchical tree format

The common symbolic program linkage handles the following languages: COBOL, PL/I, and Assembler Language. The external data base description(s) describes the logical data structure and physical data organization of data base(s) to Data Language/I. Using these techniques, it is possible to physically reorganize established data bases in a timely manner without modification to application programs.

In the COBOL language, the common symbolic program linkage enables use of the ENTER LINKAGE and the CALL verb to perform the input/output functions above. Application programs written in PL/I or Assembler Language use similar statements to reference Data Language/I. Because of this approach to data reference, input/output operations and associated system control blocks are not compiled into the application program. This removes dependence upon currently available access methods and physical storage organizations.

Each data base description is created from user-provided statements that define the logical data structure and physical organization of each data base. These statements are input to an offline utility program of IMS/VS. The result of the utility program is the creation and storage of a data base description in the user-defined data base description library. This data base description provides Data Language/I with a "mapping" from the logical structure of the data base used in the application program to the physical organization of the data used by operating system data management. The logical data structure can be "remapped" into a different physical organization without program modification. Other application data can also be added to this data base and still not cause a change to the original application programs. The concept of the data base description reduces application program maintenance caused by changes in the data requirements of the application.

Data Base Organization and Access Methods

Data Language/I supports two basic physical storage organizations. The first organization, hierarchical sequential, provides the basis for both the Hierarchical Sequential Access Method (HSAM) and the Hierarchical Indexed Sequential Access Method (HISAM). These were implemented in Version 1 of IMS/360 and are upward-compatible with Version 2 of IMS/360 and IMS/VS.

The second organization, hierarchical direct, provides the basis for two additional accessing techniques. The hierarchical direct organization is made available through the Hierarchical Direct Access Method (HDAM) and the Hierarchical Indexed Direct Access Method (HIDAM). HDAM uses an addressing algorithm for direct access support of the hierarchical direct organization. The user may incorporate his own algorithms. HIDAM is an indexed access support of the hierarchical direct organization. These access methods were implemented in IMS/360 Version 2 and are upward compatible to IMS/VS.

The primary difference between the hierarchical direct and the hierarchical sequential organizations is the manner by which segments are related and the techniques of data access. Segments in the hierarchical sequential organization that represent one physical data base record (that is, a physical hierarchical tree structure) are related by physical juxtaposition. This requires that segments that represent one physical data base record be contained in a variable number of storage blocks unique to that physical data base record. Figure 10 presents the hierarchical sequential physical storage of the logical structure depicted in Figure 5.

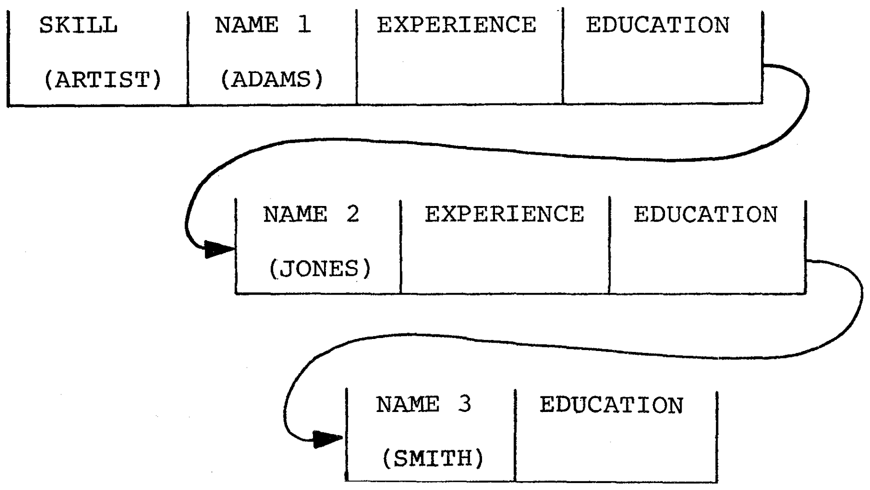


Figure 10. Hierarchical sequential physical storage of Figure 5 logical structure

Physical blocks (a variable number) required to contain a particular physical data base record are either related by physical juxtaposition (HSAM) or by direct address relationships (HISAM).

HSAM is used for sequential storage and access on tape or direct access storage. The operating system Sequential Access Method is used as a basis for HSAM.

HISAM is used for indexed access to the hierarchical sequential organization and uses either ISAM or Virtual Storage Access Method (VSAM) for the index structure. In either case, each physical data base record starts within an ISAM or VSAM logical record. If additional space is required to store dependent segments of this data base record, it is obtained from either VSAM or the Overflow Sequential Access Method (OSAM). Direct addresses relate all the physical blocks for one physical data base record.

The IMS/VS-Data Language/I Overflow Sequential Access Method combines the better characteristics of BSAM and BDAM for use in data overflow from ISAM. The data sets upon which OSAM operates are defined as standard operating system physical sequential data sets. OSAM concurrently provides:

- Sequential and direct access for retrieval
- Update in place as well as addition at the end of a data set, that is, extension of a direct data set
- A special provision for recognition of the current end of a data set
- Direct data set usage without formatting
- Secondary extent definition for data sets

Virtual Storage Access Method (VSAM) is a System Control Program (SCP) available with the operating systems (OS/VS1 and OS/VS2). It has the capability to permit multiple accessing modes (direct, sequential, keyed, and addressed) within a single access method. VSAM data sets can be divided into two types: key sequenced and entry sequenced. In key sequenced organization, the physical placement of records is ordered by the collating sequence of a key field. The entry sequenced data set contains records that are physically placed in the same order as they were created. VSAM has the following advantages:

- A new data format designed for long term stability
- Keyed access to data by means of an index, either direct or sequential
- Addressed access to data by means of a relative byte address in either direct or sequential mode

The use of VSAM as an alternative to ISAM and OSAM is a user option selected at the time the data base is defined.

When a single segment (root segment only) HISAM data base is stored using VSAM, its record format is compatible to a standard VSAM key sequenced data set.

Segments in the hierarchical direct organization that represent one physical data base record (that is, a physical hierarchical tree structure) are stored in one or more physical blocks. However, all segments in that physical data base record, rather than physical blocks containing the data base record, are related by direct addresses. Each segment in a physical data base record relates to segments of the same type as well as to adjacent segment types through direct addressing. Physical blocks that contain segments of a data base record are not related by direct addressing.

When the hierarchical direct organization is used, Figure 11 represents the segment direct address relationships in the physical storage of the logical structure from Figure 5. Within a physical data base record, occurrences of a particular segment type are related by direct addressing to other occurrences of the same segment type (physical twins) under a given parent. In addition, the segment type immediately above (physical parent) and the first and last occurrence of each segment type immediately subordinate (physical children) are related by direct addressing. (See Figure 12.)

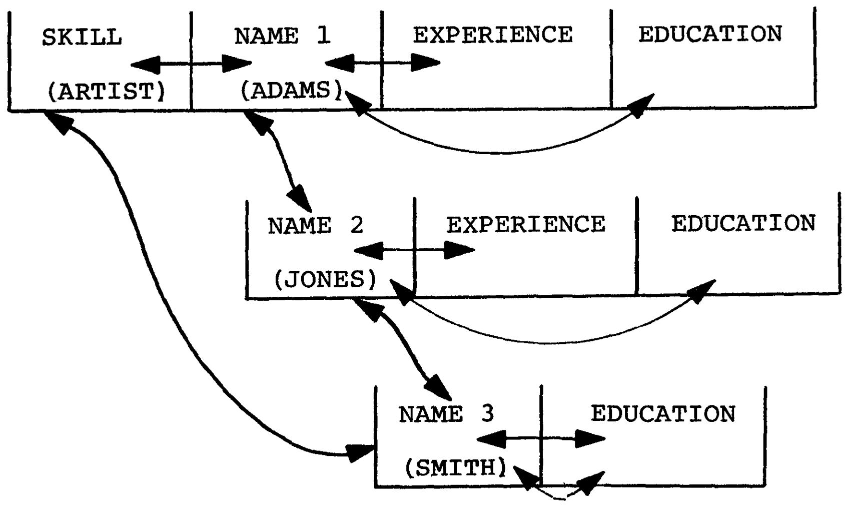


Figure 11. Hierarchical direct physical storage of Figure 5 logical structure

In the hierarchical direct organization, the space requirement for each segment is increased from that required in the hierarchical sequential organization. This space is required to accommodate direct addresses. However, the following advantages are gained:

- More rapid direct access to segments within a physical data base record
- The ability to share space in a direct access storage block across multiple data base records. One physical block may contain segments from different physical data base records. This may result in considerable data base storage space saving.
- The ability to reuse space occupied by deleted segments through the maintenance of free space addresses

HDAM is used for algorithmic addressability to records in hierarchical direct organization. A choice of either OSAM or VSAM is available as a base for HDAM.

HIDAM is used for indexed access to the hierarchical direct organization. The index for HIDAM may be constructed with either VSAM or ISAM, at user discretion. The logical record of either access method contains the key of a root segment and a direct address to the root segment. The storage of all segment data is actually accomplished in an additional data set, a VSAM entry-sequenced data set paired with a VSAM key-sequenced data set or an OSAM data set paired with an ISAM data set. Because the data set for the index storage is separated from the data set for the segment or physical data base record storage, reorganization of the index separate from the data is facilitated.

Segment Definition and Format

The foregoing discussion has introduced some of the concepts used in the physical storage of data. In both storage organizations discussed, the term physical data base record is used. The physical data base record is represented as a simple hierarchical tree structure of related segments. The hierarchical tree structure represents one sort sequence of segments. This chapter later discusses how logical data base records are constructed from one or more physical data base records in the same or different data bases. In this manner, many sort sequences of segments can be obtained. The foregoing discussion also indicated that segments in the hierarchical direct organization are related by direct addresses. Independent of whether the organization is hierarchical sequential or direct, each segment is composed of two parts--the _prefix_ and the _data_. The prefix contains:

- Segment type
- Segment deletion indicators
- Direct addresses that establish intersegment relationships

The format of the prefix for any segment type is unique. It is determined by the data base organization and segment interrelationships. The segment format, which includes both prefix and data portions, is specified in the data base description. The use of the segment prefix is controlled entirely by Data Language/I. The using application program need not be concerned about the presence or the format of the segment prefix.

The data portion of a segment is composed of one or more user-supplied data fields. One of these fields describes the physical sequence of occurrences of that segment type. Only the data portion of a segment is passed between an application program and Data Language/I.

Physical Data Base Record

If the assumption is made that Figure 5 represents one physical data base record, Figure 12 shows the _physical_ parent-child-twin intersegment relationships. The skill segment--artist--is the root segment and _physical parent_ to three name segments. Each name segment is a _physical child_ of the skill segment. Figure 12 shows two occurrences of the skill segment type--one for artist and one for mechanical engineer. Two or more occurrences of one segment type represent a relationship of twins. Two or more occurrences of one segment type within a physical data base record, such as the name segments for Adams, Smith, and Jones, represent _physical twins_.

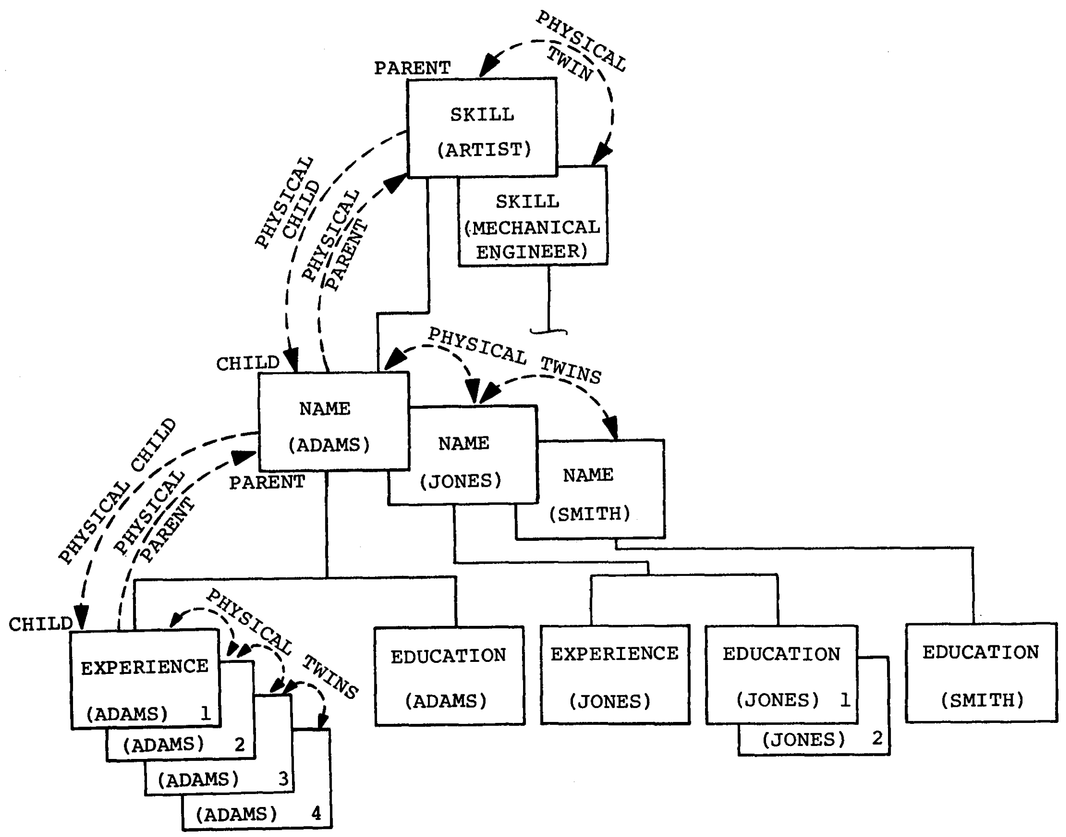


Figure 12. Physical data base record parent-child-twin intersegment relationship

Depending upon the data base organization (hierarchical sequential or hierarchical direct), the physical parent-child-twin relationships may be indicated by physical juxtaposition of segments or by direct addresses in the prefix of a segment.

Interrelated Physical Data Base Records

Even though logical data base records can be described in the form of simple hierarchical trees, their physical representation may be considerably more complex. A logical data base record might be composed of segments from only one physical data base record. Alternatively, a logical data base record might be composed of segments from several physical data base records. These physical data base records could be contained in one data base or in multiple data bases. Here the term data base means a family of physical data base records in which all have a common hierarchical segment structure. For example, Figures 3 and 4 each represent hierarchical structures that could represent physical data base records. All records of the structure in Figure 3 represent one data base, and all records of the structure in Figure 4 represent another data base.

With reference to Figure 3 and the discussions of interrelated tree and inverted data structures, Figure 13 indicates the relationships that can be established between physical data base records in two different data bases.

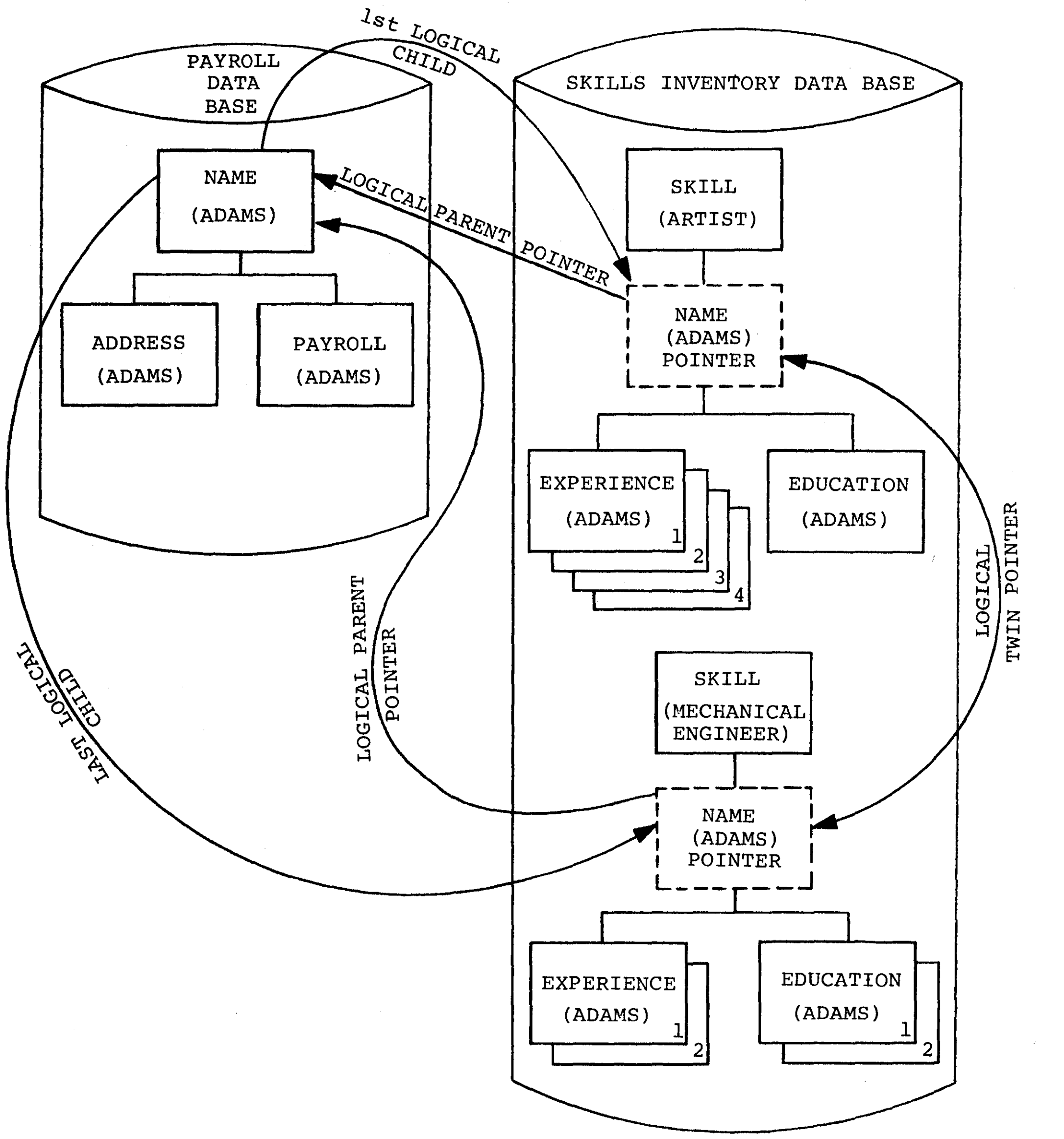


Figure 13. Relationship between physical data base records (two data bases)

These relationships are again described through the parent-child-twin terminology. However, the relationshps are considered logical rather than physical. For the name segment of Adams, the _logical child_ segments are the pointer segments under skill for artist and skill for mechanical engineer. All pointer segments under skill segments that point to the name segment for Adams are considered _logical twins_. The name segment for Adams is considered the _logical parent_ for all pointer segments that point to it.

Since logical data base records are constructed from one or more physical data base records and their intersegment relationships, the reader might consider the terms _logical data base_ and _physical data base_. A physical data base would be composed of physical data base records of common segment format. The segment format is always hierarchical in nature. The logical data base is composed of one or more physical data bases. The physical data base records are interrelated by logical parent-child-twin and physical parent-child-twin relationships.

Depending upon the data base organization, the logical parent-child-twin relationships may be indicated by symbolic key values or direct addresses in a segment's prefix. If the logical relationships are between two or more physical data base records in the hierarchical direct organization, they are accomplished through direct addresses or symbolic identifiers. This is true whether the relationships exist within one or multiple data bases. If the logical relationships are between two or more physical data base records in the hierarchical sequential organization, they are accomplished through symbolic key values.

If the logical relationships are between records in a hierarchical direct and a hierarchical sequential organization, they are accomplished through a combination of symbolic key values and direct addresses.

CHECKPOINT FACILITY

The master terminal of IMS/VS provides the operator a means of entering commands for checkpoint execution. These commands provide the ability to:

- Cause the message queue control blocks and associated pieces of information to be recorded on the IMS/VS system log

- Cause orderly termination of the IMS/VS system. Unprocessed input messages may be retained on direct access storage queues or recorded on the system log for subsequent processing.

BATCH CHECKPOINT FACILITY

User batch processing programs are provided a facility to synchronize checkpoints taken of their environment by means of the IMS/VS tape log. This support is based on the operating system checkpoint facility, but any user checkpoint/restart method may be employed.

RESTART FACILITY

IMS/VS is capable of being stopped and restarted daily or at other explicit intervals. To start the IMS/VS system, the operator performs the procedure for initial program load (IPL) of the operating system and then instructs the operating system to START IMS. Once the IMS/VS control program is operative, one or more jobs may be initiated which become IMS/VS processing address spaces. Remaining address spaces are used for batch processing. Upon initiation of the IMS/VS control program, a message is then transmitted to the master terminal requesting an indication of the type of restart for IMS/VS. The operator's response causes control to pass to the restart facility, which reads a tape of the system log. (This tape was written by CHECKPOINT at the previous system stop.) On this tape are input messages received but not processed or any output messages generated but not transmitted on the previous execution.

Any other information required to restart the system is also carried over on the tape. Messages on this tape are put back into the same queues in which they were left at the previous system stop. When the end-of-file indicator is reached, the master terminal is informed that restart is complete. The master terminal operator may now enter control messages to initiate communication line operation, message processing, and data base reference.

Restart, when performed without a previous system log, amounts to an initial start for all message transmission and processing.

In addition to normal restart, three emergency restart procedures are provided by IMS/VS:

1. A procedure that handles the condition caused by an ABEND of IMS/VS or by a machine error causing an ABEND where the data bases, system log, and the message queues are not destroyed

2. A procedure that handles the ABEND condition described above when IMS/VS message queues are destroyed

When IMS/VS is restarted after an ABEND condition occurs, the restart capabilities of IMS/VS provide the following information to the master terminal:

1. The message processing program that was executing in each message processing address space at the time of ABEND

2. The input messages that caused the message processing programs in Item 1 to be scheduled

Modifications to data bases are placed on the system log. This information can be used at restart to correct data base modifications made by programs in process at the time of system failure. The data base modifications caused by programs in process at time of failure are inverted and the original input messages are reprocessed in their entirety.

For batch programs, a utility is provided which uses the log records created during the original run to back out the data base changes to a specified user checkpoint. The user's restart facility may then be invoked to restart the job at the same checkpoint.

In addition to system restart, facilities are provided to reconstruct data bases. A utility program operative in the batch environment is provided to create a tape copy of any data base. The use of this data base copy and the data base modifications recorded on the system log can be used to reconstruct a data base. Reconstruction is accomplished by reloading the data base from the tape copy. Then the latest changes to all physical blocks in the data base are applied through another utility program from information on the system log. This second utility also operates in the batch environment.

CHAPTER 5. SYSTEM FLOW

TELEPROCESSING SYSTEM

Once the region containing the IMS/VS control program and one or more address spaces to be utilized for the message processing have been initialized by the job management facilities of the Operating System, the following system flow occurs (see Figure 14):

1. The telecommunication facility (event 1) requests restart instructions from the master terminal. After the completion of restart, the master terminal enables communication from all user terminals (event 2).

2. When an input message or message segment is received (event 2), the telecommunication facility invokes the common service facility (event 3), and the input message is logged (event 4) and queued (event 5).

3. When there are input messages pending for processing, and a message processing address space of the required class is available for scheduling, control is passed to the scheduling facility to determine the application message processing program to be scheduled. The application program is loaded into address space B and given control.

4. The application program subsequently makes requests for the input message and/or data base references (event 6). Control passes to the Data Language/I facility for either message reference (event 7) or for data base reference (event 8). The message reference is accomplished through the common service facility.

5. During application program execution, modifications can be made to the data base (event 8) and/or output messages may be queued (events 5 and 7).

6. When the application program terminates or requests another input message, all output messages queued are transmitted to the designated output terminal (events 3 and 2).

BATCH PROCESSING OF TELEPROCESSING DATA BASES

Once the IMS/VS address space associated with teleprocessing have been initiated by the Operating System, a batch region can be initiated. The application program in the batch address space is scheduled by Operating System job management. This batch region may contain an application program for processing against teleprocessing data bases. The Data Language/I facility of IMS/VS is used for data base reference and update (Figure 14). Any data reference is initiated by the batch application program (event 9).

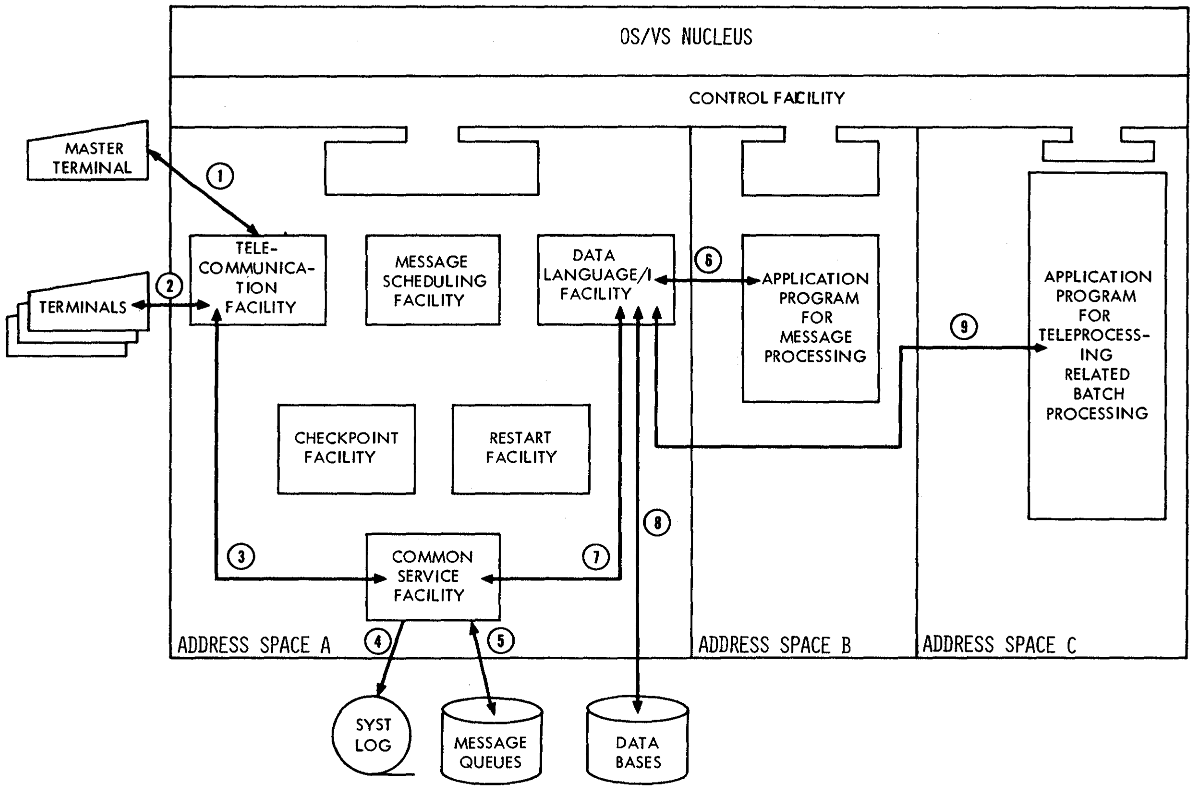


Figure 14. Teleprocessing and related batch IMS/VS system flow

DATA LANGUAGE/I DATA BASE BATCH PROCESSING

Whether or not the teleprocssing capabilities of IMS/VS exist within the jobs operating under the Operating System, the Data Language/I facility of IMS/VS can be used in a batch-only data base environment (see Figure 15).

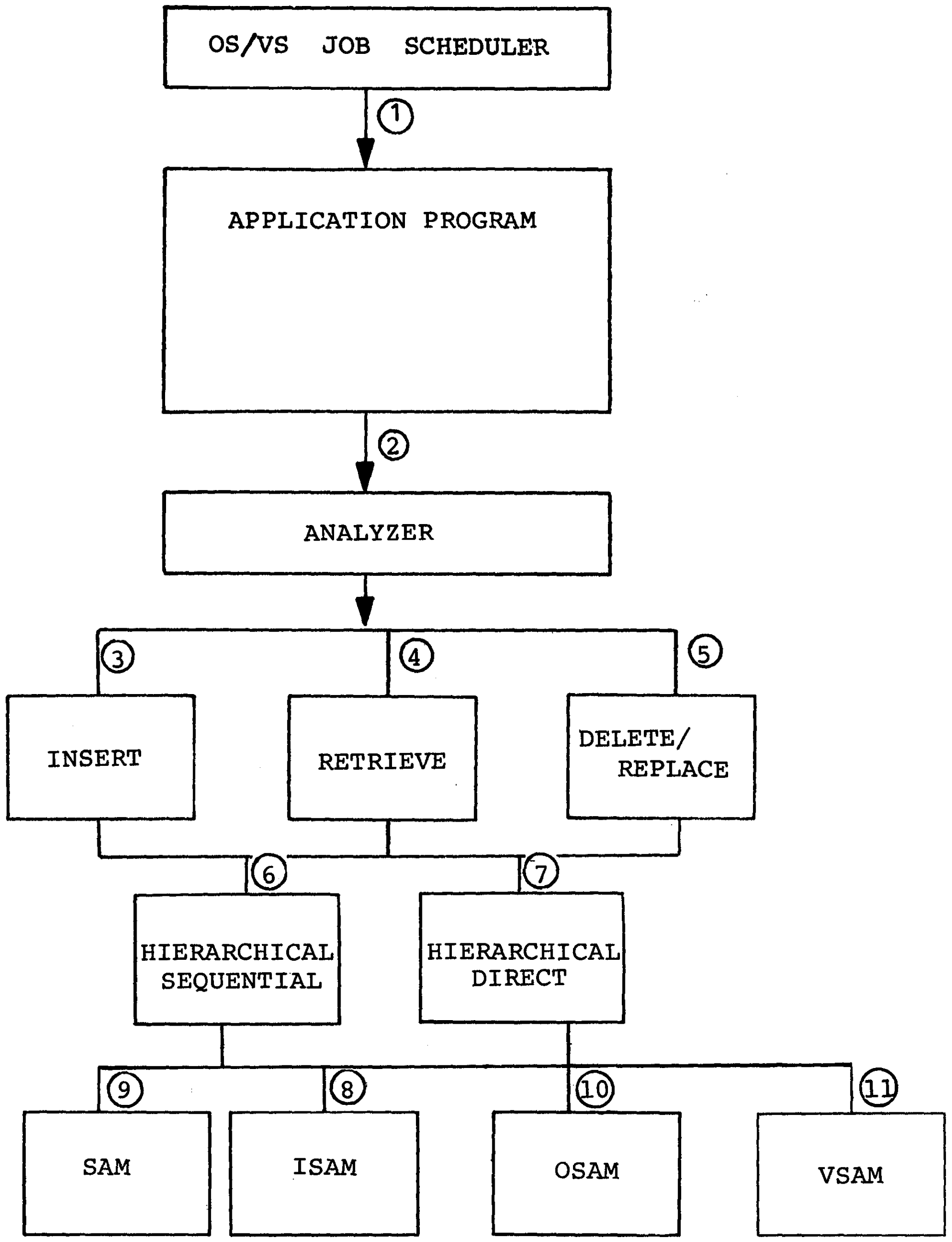


Figure 15. Data Language/I data base batch system flow

1. The application program for batch-only data base processing is initiated through the job management routine of the Operating System (event 1).

2. The Data Language/I facility is invoked by the application program (event 2). The highest level Data Language/I module analyzes the data base call request. Depending upon the I/O function requested in the call request, the insert (event 3), the retrieve (event 4), or the delete/replace function (event 5) is invoked. These functions subsequently invoke functions unique to either the hierarchical sequential (event 6) or the hierarchical direct organization (event 7). These functions subsequently invoke access method modules for ISAM (event 8), SAM (event 9), OSAM (event 10) or VSAM (event 11).

CHAPTER 6. USER RESPONSIBILITIES

The user of IMS/VS has two primary responsibilities:

1. The development of data processing applications that use the facilities of IMS/VS. This development includes all application programs for message and batch processing and for the creation of data bases through the facilities of Data Language/I.

2. The definition of his data processing environment:

   - Data bases
   - Processing programs
   - Transaction types, classes, and processing priorities
   - Communications lines and terminals
   - Security requirements

Note: If the communication network includes remote System/3 or System/7 as either an end-use terminal or a concentrator, it is also the user's responsibility to provide the necessary program in the remote CPU to communicate with IMS/VS in accordance with the Intelligent Remote Station Support (IRSS) interface. The user must also provide the necessary bit-handling routines to interface to IMS/VS and all System/3 or System/7 application programming to perform blocking, deblocking, and other message-handling functions, as required by the application. This interface is described in the IMS/VS System/Application Design Guide and the System Programming Reference Manual.

APPLICATION DEVELOPMENT AND STRUCTURING OF IMS/VS

Those parameters that define a particular application must be provided by the user. These include:

1. The definition of each Data Language/I data base in terms of its hierarchical structure and storage, and the creation of each data base in the batch processing environment

2. The definition and construction of all message and batch processing programs. For message processing programs, Data Language/I must be used exclusively for all input/output requests. For both message and batch processing programs, the following restrictions apply:

   a. If multitasking is used within an address space, all Data Language/I requests must be made from the user application program task given control by IMS/VS.

   b. If COBOL is used, no asynchronous processing is allowed against a Data Language/I data base.

3. The definition of various transaction types, classes, and their associated processing programs, scheduling priorities, and detailed security characteristics

4. The definition and the number and types of communications lines and terminals utilized by the application

The user must also structure IMS/VS by the creation of a control block for each communications line, terminal, message type, message processing program, and data base. The construction and integration of these control blocks into the resident IMS/VS control program are performed by the system definition utility program. Restructuring of the control blocks may be necessary periodically as the operating environment changes.

PROCESSING IN AN OS/VS ENVIRONMENT

All IMS/VS environments will execute in either virtual or real storage.

VIRTUAL=VIRTUAL ADDRESS SPACE (V=V)

In most cases, programs execute in virtual address spaces and are paged by the supervisor. These address spaces are allocated in segment multiples. The REGION parameter is increased to a size that is a multiple of 64K. The address spaces are allocated from the low end of the virtual dynamic area. The minimum virtual space requirement for a job is 192K: one segment for LSQA, one segment for System Work Area, and one segment for the region.

When the IMS/VS control address space is run in a virtual environment, the user is given the ability to fix specific portions of the control program. This allows the user to tune his IMS/VS system to his particular environment. The parameters indicating which portions are to be fixed are contained in members in the IMS2.PROCLIB library. The particular member to use is indicated in the 'PARM' field of the control address space 'EXEC' card.

VIRTUAL=REAL ADDRESS SPACE (V=R)

The limit of virtual address space having the same address range as the real storage, or optionally less, is specified during supervisor initialization. This area, the virtual-equal-real dynamic area (V=R), is used to execute jobs that cannot run in a paged environment. The use of this area is expected to be limited, because the real storage assigned to this area is fixed when it is allocated to a job requesting V=R.

The use of this facility is specified by a new JCL parameter, ADDRSPC=REAL. When this parameter is encountered, the system allocates an address space that is a multiple of the page size from the virtual-equal-real dynamic area and fixes the real page having the same address as each virtual page in the address space. This ensures that translation exceptions never occur in this area, and channel programs for I/O in this area need not be translated. Each V=R address space has a unique storage protection key.

CHAPTER 7. PERFORMANCE

IMS/VS provides a functional capability. It is the user's responsibility to prepare the system configuration, provide the operational environment, and design and implement the application program(s) to meet his requirements. It is therefore difficult to provide other than a range of possible performance objectives. Performance within this range is predicated on the user having reasonably standard requirements and implementing his application while adhering to good realtime programming practices.

TIMING AND THROUGHPUT INFORMATION

As user demands on the system increase, that is, through transactions per unit of time or through added functions, response time may be held relatively constant by increasing the amount of main storage, channel throughput, and CPU power.

Response time varies, depending upon:

- The type of terminal. The IBM 3270 Information Display System has a substantially faster transmission rate in characters per second than a terminal such as the IBM 2740 Communication Terminal that produces hard copy.

- The number of terminals on the system. Each terminal represents an increase in use of system resources.

- The number of terminals on a communication line and the transmission speed of the line. Multiple terminals on a single communication line increase the chance of a delay in communication with a specific terminal; transmission speed of the line also directly affects response time.

- The terminal operator entry technique and the output response format. Inquiry and order entry dictate the number of transmissions to and from a terminal and the total number of characters that are transmitted each time.

- The central processing unit, real main storage size, and related input/output devices. The resources available on the data processing system as well as those available at any one moment to process multiple transactions affect the throughput rate for each transaction.

- The scope of the user application. The demands upon the entire system increase as the application is extended from one in which inquiries are processed to one in which: (1) orders are entered, (2) data bases are updated to reflect change, and (3) work assignments are distributed.

- The number of transactions to be entered. As the volume of transactions increases, the likelihood that there will be a wait for the availability of a facility such as a data base segment also increases.

PERFORMANCE CONSIDERATIONS

The performance of IMS/VS in a virtual environment is highly dependent on the system resources available, on any programs that operate concurrently and their relative priorities, system and application data set placement, system timing, etc.

For specific online performance and response time requirements, particular attention must be given to assuring that adequate real resources (main storage, CPU computing capability, channels, disk file arms, etc.) are available.

In some cases, the program must be benchmarked using the specific user workload and configuration to verify what system resources are necessary to give adequate performance.

CHAPTER 8. MACHINE CONFIGURATIONS

MINIMUM IMS/VS CONFIGURATIONS

SYSTEM FUNCTION	UNITS PERMITTED
Processing Unit	System/370 Models 145, 155II, 158, 165II, 168 with relocate feature and main storage of:

IMS/VS	OS/VS1	REL 1 OS/VS2	REL 2 OS/VS2
DB System	384K	512K	768K
DB/DC System	512K	768K	1000K

Minimum Address Space	The practical minimum virtual address space size is:

IMS/VS	OS/VS1	REL 1 OS/VS2	REL 2 OS/VS2
DB System	90K*	90K*	90K*
DB/DC System	200K	200K	200K

*Plus the size of the user's application program

In order to determine particular virtual storage requirements, complete either the DB or the DB/DC storage estimates worksheet in the IMS/VS System/Application Design Guide. Examples of minimum configuration storage estimates are also provided in that manual.

System Console	See OS/VS1 or OS/VS2
Tape Units	DB System: at least one 2400 9-track DB/DC System: at least one 2400 9-track
Direct Access	For system libraries and working storage space, any devices supported by the operating system. Minimum space for system use and maintenance:

	2316 cyls	3336 cyls
DB System	125	83
DB/DC System	225	158

	For IMS/VS data base storage, within the capability and restrictions of the operating system support by Indexed Sequential Access Method, Sequential Access method, and Virtual Storage Access Method:
	2314/2319 Direct Access Storage Facility 2305 Fixed Head Storage 3330 Disk Storage
Telecommunications	For the DB/DC system master terminal, one of the following nonswitched station-controlled devices is required:
	2740 Model 1 Communication Terminal 1050 Data Communication System with 1052 Printer-Keyboard
	A complete description of the required and prohibited features for telecommunications control and terminal devices appears below.

Terminal Control Units

The 2701 Data Adapter Unit, 2702 Transmission Control, or 2703 Transmission Control may be used. If Audio Response support is desired, the 7770 Audio Response Unit is required. If 3277 local display stations or 3284/3286 local printers are desired, a 3272 Control Unit is required. In addition, if remote 3277 display stations or 3284/3286 remote printers are desired, a 3271 Control Unit is required. Line adapters and/or data sets must be added as required for the selected terminal control units and communication line facilities. The specifications shown here do not consider combinations of terminal types or lines attached to the same control unit. Contact your local IBM representative for a description of prerequisites and limitations applicable to each control unit. Programmable features or features which change the line or terminal control characteristics and which are not shown below are not supported. See your IBM representative for information on restrictions and dependencies upon CPU models and channels.

<u>2701</u> <u>Data Adapter Unit Model 1</u>

For use with 1050 and 2770 Data Communication Systems, 2740 Communication Terminal, 2780 Data Transmission Terminal, 3270 Information Display System, 2980 General Banking Terminal System, System/3, System/7, and 33/35 Teletypewriter (ASR)

	<u>Feature #</u>
For 1050, 2740-1, 2740-2, System/7:	
IBM Terminal Adapter Type I with appropriate speed selection	4640

	Feature #
For 2770 and 3270:	
Synchronous Data Adapter Type II	7698
Transmission Code	9060 or 9061
Features not permitted:	
Transparency	8029
For 2780:	
Synchronous Data Adapter Type II	7698
Transmission Code	9060, 9061, or 9062
Features not permitted:	
Transparency	8029
For 2980:	
Synchronous Data Adapter Type II	7698
Transmission Code	9060
Features not permitted:	
Transparency	8029
For 33/35 Teletypewriter (ASR):	
Telegraph Adapter Type II	7885
For System/3:	
Synchronous Data Adapter Type II	7698
Transmission Code	9060
Transparency	8029

2702 Transmission Control

For use with 1050 Data Communication System, 2740 and 2741 Communication Terminals, 33/35 Teletypewriter (ASR), and System/7

For 1050, 2740-1, 2740-2, System/7, 2741:

IBM Terminal Control Type I	
appropriate selective speed	4615
Terminal Base	9696

For 2741 add:

Line groups for 2741 terminals with active Interrupt feature (#8055) must be isolated from 1050/2740-1 and 2740-2 terminals and the appropriate selective special features.

For 33/35 Teletypewriter (ASR):

Terminal Control Base	9697
Terminal Control Expansion	7935

Additional features supported:

Autopoll	1319

	Feature #
Two processor switch	8110

2703 Transmission Control

For use with 1050 Data Communication System, 2740 and 2741 Communication Terminals, 2780 Data Transmission Terminal, 2980 General Banking Terminal System, 3270 Information Display System, and 33/35 Teletypewriter (ASR).

For 1050, 2740-1, 2741:

IBM Terminal Control Type I	4696
IBM Terminal Control Base	4619
Start Stop Base Type I	7505
Line Speed Option 134.5 bps	4878

For 2741:

Line groups for 2741 terminals with active Interrupt feature (#8055) must be isolated from 1050/2740-1 and 2740-2 terminals and the appropriate selective special features.

For 2740-2, System/7:

IBM Terminal Control Type I	4696
IBM Terminal Control Base	4619
Start Stop Base Type I or II	7505 or 7506
Line Speed Option 134.5 bps	4878
Line Speed Option 600 bps	4879

For 2770:

Synchronous Attachment	7702
Synchronous Base Type 1A or 2A	7703 or 7706
Base Expansion	1440
Synchronous Terminal Control	7715 or 7716

Features not permitted:

Transparency	9100

For 2780:

Synchronous Attachment	7702
Synchronous Base Type 1A, 1B, or 2A	7703, 7704, or 7706
Base Expansion	1440
Synchronous Terminal Control	7715, 7716, or 7717

Features not permitted:

Transparency	9100

For 2980:

Synchronous Attachment	7702
Synchronous Base Type 1A, 1B, or 2A	7703, 7704, or 7706
Base Expansion	1440
Synchronous Terminal Control	7715

Features not permitted:

Transparency	9100

	Feature #
For 3270:	
Synchronous Attachment	7702
Synchronous Base Type 1A or 2A	7703 or 7706
Base Expansion	1440
Synchronous Terminal Control	7715 or 7716
Features not permitted:	
Transparency	9100
For 33/35 Teletypewriter (ASR):	
Line Speed Option 110 bps	4877
Telegraph Terminal Control Base	7905
Optional features supported:	
Two processor switch	8110
2712 Attachment	8043

7770 Model 3 Audio Response Unit

For use with Touch-Tone telephone (or equivalent) or the Portable Audio Terminal communications over the ABB' code line interface.

For Touch-Tone or equivalent operation:

No special feature required

For 2721:	
ABB' Code Line Interface	1091
I/O Line Gate	4663
I/O Line Board	4660
Features not permitted:	
EOI Disable	3540

Terminal devices not supported:

IBM 1001 Data Transmission Terminal
IBM 1093 or 1092/1093 Programmed Keyboard
Rotary dial telephones
Rotary dial card dialer telephones

For System/3:	
Synchronous Attachment	7702
Synchronous Base Type 1A, 1B, or 2A	7703, 7704, or 7706
Base Expansion	1440
Synchronous Terminal Control	7715
Features required:	
Transparency	9100

Terminals

Programmable features which change the control or transmission characteristics and which are not shown are not supported. Line adapters and/or data sets must be added as required for the selected terminal control units and communication line facilities. The specifications shown here do not consider combinations of terminal types or lines attached to the same control unit. Contact your local IBM representative for a description of the prerequisites and limitations applicable to each control unit.

Terminals which are equivalent to those explicitly supported may also function satisfactorily. The customer is responsible for the impact that any changes to the IBM-supplied products or programs may have on such terminals.

The supported terminals and required features are:

	Feature #
1050 Data Communication System - Nonswitched and Switched Network:	
1051 Model 1 or Model 2 Control Unit:	
Keyboard Request (if a 1052 is attached)	4770
Automatic EOB (if a 1052 or 1056 is attached)	1313
Optional feature supported:	
First Printer attachment	4408
1052 Model 1 or 2 Printer-Keyboard:	
Automatic EOB	1313
Dual Case Printing Element	9571 or 9591
Optional feature supported:	
Forms Feed Control	4452
1053 - 1 Printer:	
Dual Case Printing Element	9571 or 9591
Optional feature supported:	
Forms Feed Control	4452
1054 - 1 Paper Tape Reader:	None
1055 - 1 Paper Tape Punch:	None
1056 Model 1 or Model 2 Card Reader:	
Optional feature supported:	
Extended Character Reading	3861
1057 - 1 Card Punch:	None
1058 - 1 Printing Card Punch:	
Optional feature supported for 1057 and 1058:	

	Feature #
Extended Character Punching	3860

2740 Model 1 - Nonswitched Network:

Record Checking	6114
Terminal to Multiplexer	9700
Dual Case Printing Element	9571 or 9591
Automatic EOB	1313

Optional features supported:

Station Control (Required for master terminal and multipoint terminals.)	7479

2740 Model 1 - Switched Network:

Record Checking	6114
Transmit Control	8028
Terminal to Multiplexer	9700
Dual Case Printing Element	9571 or 9591
Automatic EOB	1313
Dial up	3255

2740 Model 2 - Nonswitched Network:

Record Checking	6114
Buffer Receive	1499
Dual Case Printing Element	9571 or 9591

Optional features supported:

Buffer Expansion positions 121-248	1495
Buffer Expansion positions 249-440	1496
Edit	3600
Header Control	4510
Document Insertion	3401 or 3402

2741 - Nonswitched and Switched Network:

Optional feature supported:

Receive Interrupt	4708

2770 Data Communications System - Nonswitched Network:

2772 Multipurpose Control Unit:

Feature required:

Multipoint Data Link Control ("Inquiry Mode" operation of feature #5010 is supported.)	5010

Optional Features Supported: (See Sales Manual for valid combinations of these features)

Buffer Expansion	1490
Buffer Expansion, Additional	1491
Display Format Control	3250
Expanded I/O Capability	3830
Expanded Print Line	3860

	Feature #
050 Attachment -- first	3940
050 Attachment -- second	3941
545 Attachment	3950
Keyboard Correction	4690
Keylock	4695
Optical Mark Read	5450
Printer Horizontal Format Control	5890
1017 Attachment	7910
1018 Attachment	7915
1053 Model 1 Attachment	7920
1053 Ribbon Shift	7925
1053 Vertical Forms Control	7930
Transmit/Receive Monitor Print	7950
2203 Model A1 Attachment	8000
2203 Model A2 Attachment	8001
2213 Model 1 Attachment	8010
2265 Attachment	8015
2502 Model A1 Attachment	8020
2502 Model A2 Attachment	8021
2213 Model 2 Attachment	8700
15 Rows Screen (2265)	9101
12 Rows Screen (2265)	9102
1255 Attachment	9755
EBCDIC Transmission Code	9761
USASCII Transmission Code	9762

Features not supported:

Data Set Attachment	9123
Automatic Answering	1340
Conversational Mode	1910
EBCDIC Transparency	3650
Identification	4610
Security Identification	6310
Switched Network Attachment (WTC)	2981
Space Compression/Expansion	6555
5496 Data Recorder Attachment	3970

545 Output Punch Model 3 or 4:

Optional feature supported:

Keyboard, 48 Character for Arrangement A	9651
Keyboard, 64 Character for Arrangement USASCII	9671
Keyboard, 64 Character for Arrangement EL	9677
Acoustic Cover	9014
Punch 81 Indication	5550

2213 Printer Model 1 or 2:

Optional features supported:

6-lines per inch	9435
8-lines per inch	9436
Roll Paper Feed	6200
Pin Feed Platen	9509
Forms Stand Stacker	4450

	Feature #
2265 Display Station Model 2:	
Optional features supported:	
Transmission Code for EBCDIC	9761
Transmission Code for USASCII	9762
15 Rows of 64 Characters	9101
12 Rows of 80 Characters	9102
Display Format Control	8015
2502 Card Reader Model A1 or A2:	
Optional features supported:	
Interchangeable Feed, 51/80 Column Card	4650
Interchangeable Feed, 66/80 Column Card	4651
Optical Mark Read	5450
050 Magnetic Data Inscriber:	
2772 Attachment	7850
1017 Paper Tape Reader:	
Optional Features Supported:	
Model 1 or Model 2	9750
1018 Paper Tape Punch:	
Model 1 or Model 2	9750
2772 Attachment	8050
Optional Features Supported:	
Take up reel	7801
1053 Printer	
2772 Attachment	7850
Optional Features Supported:	
Accelerated Carriage Return	1006
Forms Feed Control	4452
Forms Stand/Stacker	4462
Pin Feed Platen	9509
Pin Feed Platen with Forms Control	9510
1255 Magnetic Character Reader. Models 1, 2, or 3	
2772 Adapter	7850
Optional Features Supported:	
Balance List	1470
Dash Symbol Transmission	3215
51-column Card Sorting	4380
High Order Zero and Blank Selection	4520
Self-checking Number	7060

Feature #

2203 Printer Models A1 or A2

Optional Features Supported:

Print Positions, 24 Additional	5558

2780 Data Transmission Terminal - Nonswitched Network:

Models 1, 2, 3, and 4 are supported; on a multipoint communication line, if any terminal is a Model 3, all 2780 terminals must be Model 3.

Terminal Use (Communication with System/360 - point to point)	9710
or Multi-Point Line Control (multi-point)	5020

Optional features supported:

Multiple Record Transmission	5010
Printer Horizontal Format Control	5800
Print Line 120 Characters	5820
Print Line 144 Characters	5821

Features not permitted:

EBCDIC Transparency	8030

2972/2980 General Banking Terminal System - Nonswitched Network:

2980 Models 1, 2, and 4 are supported in remote mode. The 2980-4 requires the Message Lights (RPQ 858156). The batch message input option is not supported. RVI response to selection is required. (There are no numbers associated with these features. They are field-pluggable options furnished through IBM Field Engineering.) Refer to *IBM SRL GL27-3020 Component Description: IBM 2972 Models 8 and 11 General Banking Terminal System*, to determine the appropriate RPQs for your installation (see the RPQ flowchart in Appendix A).

3270 - Information Display System - Nonswitched Network:

3271 Control Unit, Model 1 or 2 (Remote Attachment):

Optional features supported:

Copy	1550
ASCII Transmission Code	1087
Device Adapter (up to 7)	3250
Transmission Speed	7820 or 7821

Note: At least one 3277 with keyboard of an appropriate model must be attached to the 3271.

3272 Control Unit, Model 1 or 2 (Local Attachment):

Optional features supported:

Device Adapter (up to 7)	3250

	Feature #
Note: At least one 3277 with keyboard of an appropriate model must be attached to the 3272.	

3275 Display Station, Model 1 or 2:

Optional features supported:

ASCII Character Generator (A)	1085
ASCII Character Generator (B)	1086
ASCII Transmission Code	1087
Audible Alarm	1090
Operator Identification Card Reader	4600
66-key EBCDIC Typewriter Keyboard	4630
66-key EBCDIC Data Entry Keyboard	4631
78-key Operator Console Keyboard	4632
78-key EBCDIC Typewriter Keyboard	4633
66-key ASCII Keyboard	4634
78-key ASCII Typewriter Keyboard	4635
Security Keylock	6340
Transmission Speed	7820 or 7821
Keyboard Numeric Lock	4690
Selector Light Pen	6350
Printer Adapter	5550

3277 Display Station, Model 1 or 2:

Optional features supported:

ASCII Character Generator (A)	1085
ASCII Character Generator (B)	1086
Audible Alarm	1090
Operator Identification Card Reader	4600
66-key EBCDIC Typewriter Keyboard	4630
66-key EBCDIC Data Entry Keyboard	4631
78-key Operator Console Keyboard	4632
78-key EBCDIC Typewriter Keyboard	4633
66-key ASCII Keyboard	4634
78-key ASCII Typewriter Keyboard	4635
Security Keylock	6340
Keyboard Numeric Lock	4690
Selector Light Pen	6350

3284 Printer, Model 1 or 2 (for Attachment to a 3271 or 3272):

Optional features supported:

ASCII Character Set (A) (3271 attach only)	1087
ASCII Character Set (B) (3271 attach only)	1088
Forms Stand	4450

3286 Printer, Model 1 or 2 (for Attachment to a 3271 or 3272):

Optional features supported:

ASCII Character Set (A) (3271 attach only)	1087
ASCII Character Set (B) (3271 attach only)	1088

	Feature #
Forms Stand	4450

3284 Printer, Model 3 (for Attachment to a 3275):

Optional features supported:

ASCII Character Set (A)	1087
ASCII Character Set (B)	1088
Forms Stand	4450

33/35 Teletypewriter (ASR):

Feature required:

Four-row keyboard

Local Card Reader:

All local 80-column card readers supported by the operating system BSAM access method are supported by IMS/VS local card reader support.

Local Print:

Local print is supported for all printers, tapes, and sequential direct access data sets supported by BSAM.

If direct access data sets are provided, the option is available to switch data sets at EOV or through operator command and print the data set by means of SYSOUT writer while the alternate data set(s) is being filled by IMS/VS.

Touch-Tone telephone or equivalent operation - Switched Network:

This type of telephone is used with the 7770 Model 3 Audio Response Unit.

Features required: None

System/7 Model Axx:

Features required:

Asynchronous Communications Control	1610

- and one of the following:

Common Carrier Adapter	2165
Line Adapter, Leased Line Type 1A	4751
Line Adapter, Leased Line Type 1B	4752
Line Adapter, Limited Distance Type 2B	4750

Features recommended:

Disk Drive	5022
MSP/7	360A-TX-024

System/7 memory requirements:
Required memory depends on the chosen MSP/7 functions (see the appropriate MSP/7 manual), the needs of the user's application, and the code required to handle the IMS/VS interface. The

	Feature #
IMS/VS System/7 interface requires the pseudobinary support feature of MSP/7 or equivalent. Code required to handle the interface outside of MSP/7 can run from approximately 200 words to several thousand words, depending on the message volume and the handling complexity required by the user.	

System/3:

Disk System - Model 10	
BSCA Adapter, First	2074
Multipoint Tributary	9482
Station Selector	7477
EBCDIC Transmission Code	9060
Text Transparency	7850
Line Facility Attachment - HDX	9392
BSCA Adapter, Second	2084
Multipoint Tributary	9582
Station Selector	7487
EBCDIC Transmission Code	9070
Text Transparency	7851
Line Facility Attachment - HDX	9382
Main Storage and I/O Devices Required by System/3 System Control Program with Multiline/Multipoint Feature	5702 - SC1 6030 or 6031

System/3 programming will normally require:

1. System/3 Disk System Control Program (5702-SC1) with the following features:

Macros Feature	6020 or 6021
Overlay Linkage Editor	6026 or 6027
BSCA Multiline/Multipoint Feature	6030 or 6031

2. Basic Assembler Language (Program Product 5702-AS1) to utilize the above SCP features.

3. Sufficient main storage and I/O devices as required by the above products.

4. Main storage and I/O devices required by the user application program. It is estimated that this function can be provided in approximately 4K bytes of storage depending on the complexity of the program and the message volume anticipated by the user.

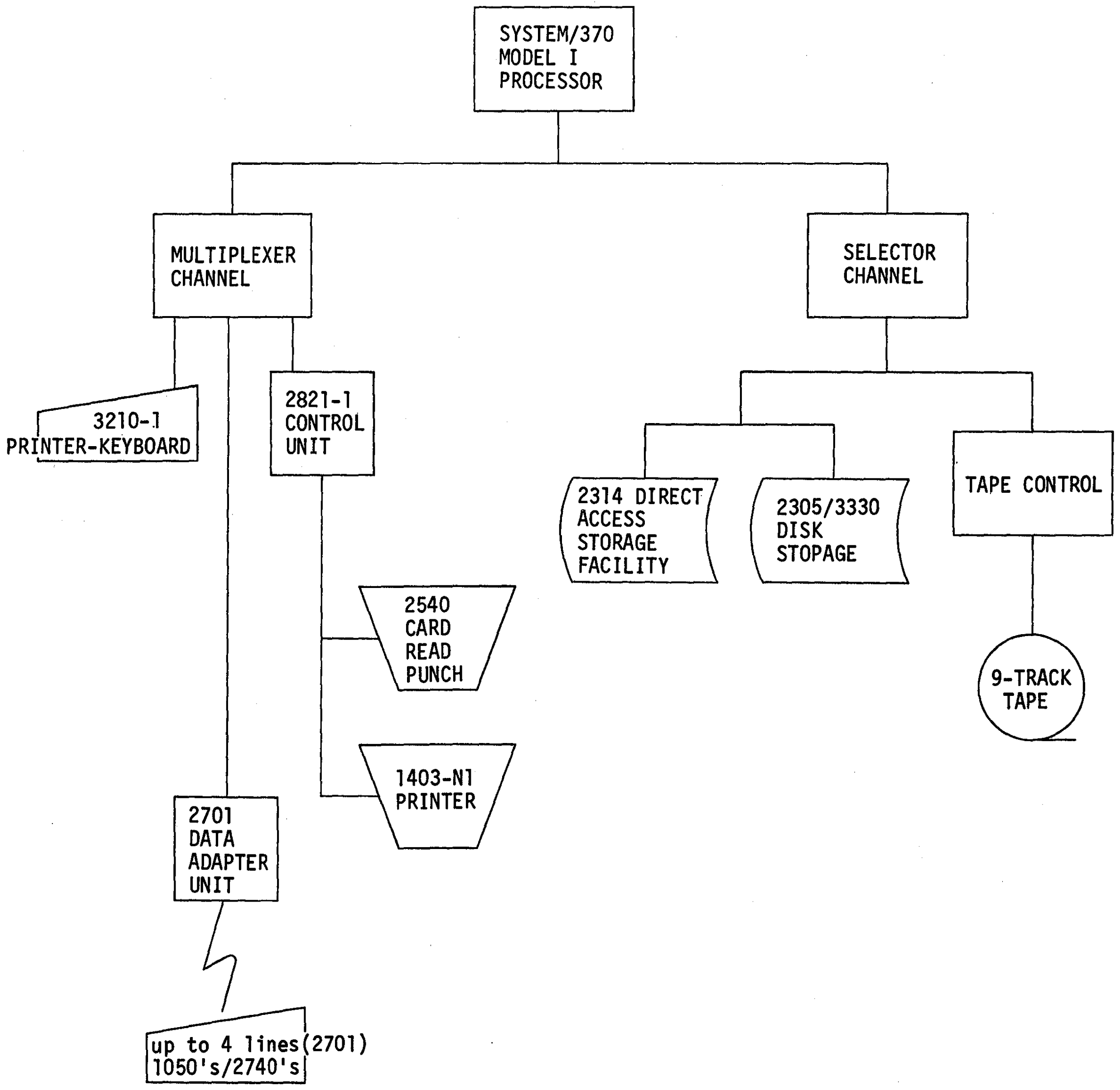


Figure 16. Minimum configuration--teleprocessing IMS/VS

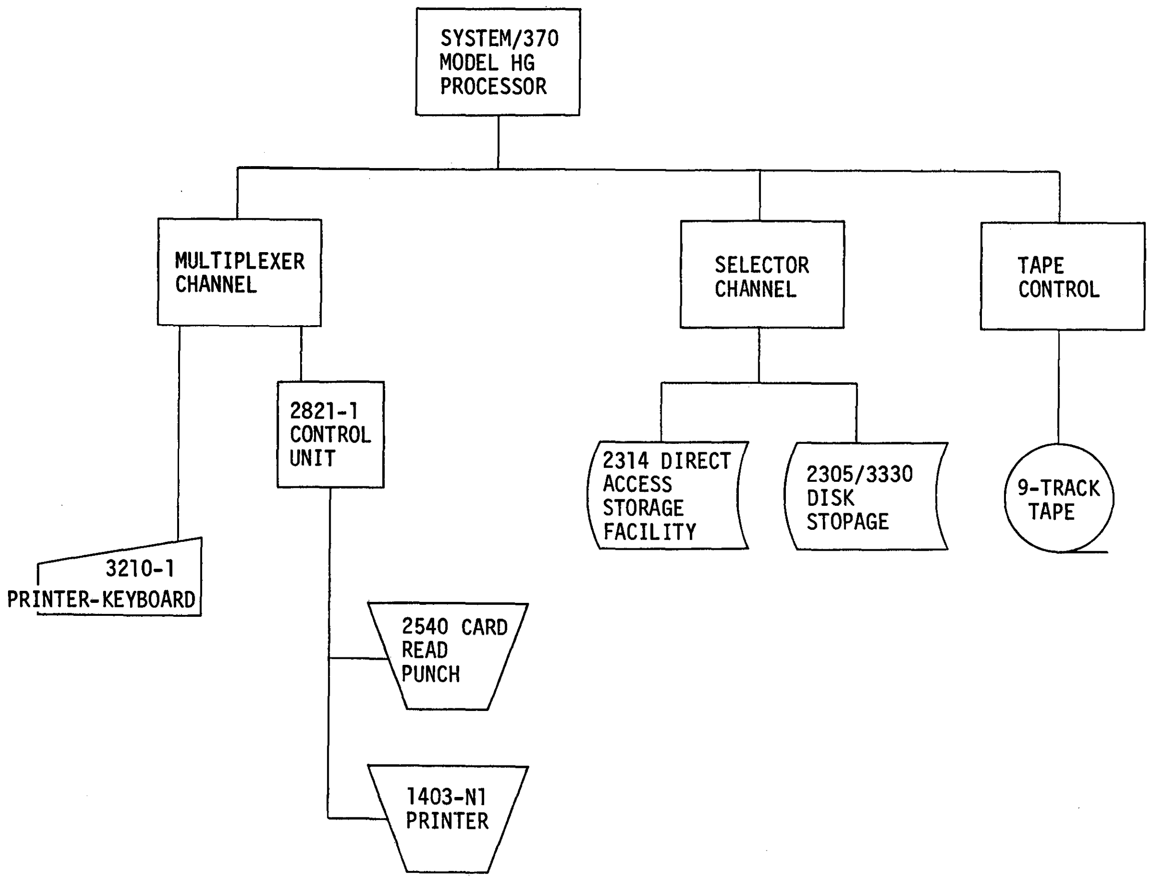


Figure 17. Minimum configuration--IMS/VS batch-only

TYPICAL IMS/VS CONFIGURATION

A System/370 Processing Unit Model HG (768K) with Multiplexer Channel...three Selector Channels...two tape controls...ten 9-track magnetic tape units...one 2314 Direct Access Storage Facility and one 3330 or 2305 Disk Storage...3210 Printer-Keyboard...two 2702 Transmission Controls with 31 Line Expansion (#7955) and the features listed under "Minimum IMS/VS Configurations"...ten 1050 Data Communication Systems with the features listed under "Minimum IMS/VS Configurations"...fifty-two 2740 Communication Terminals with features listed under "Minimum IMS/VS Configurations"...2821 Control Unit Model 5...2540 Card Read Punch...two 1403 Printers Model N1. (See Figure 18.)

Note: The maximum number of input/output devices, including communications lines, that may be attached to the system is in accord with the capabilities of the operating system.

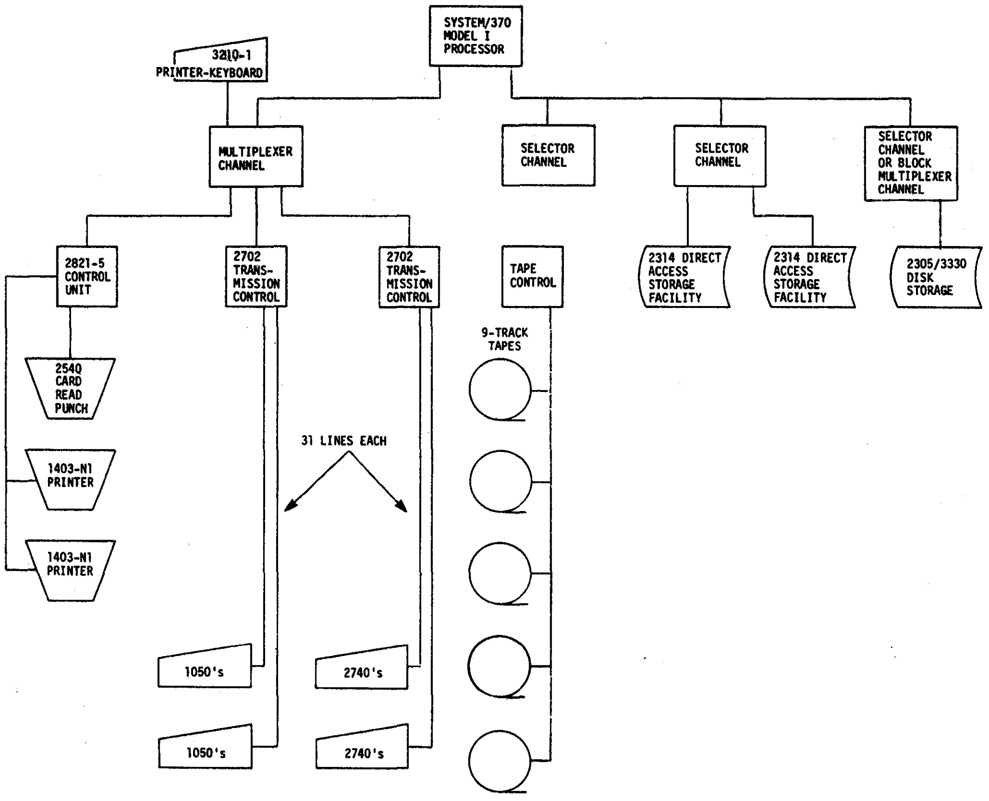


Figure 18. Typical configuration--IMS/VS system

TYPICAL IMS/VS STORAGE REQUIREMENTS

For the storage requirements of OS/VS1 and OS/VS2, please refer to the operating system Storage Estimates Manual (GC24-5094,VS1).

The following are the estimated resident storage requirements of IMS/VS. These figures may be used for general planning purposes in estimating the storage requirements of the IMS/VS control program. Additional main storage is required for message processing (typically 30K) and batch processing.

1. Required code

Resident nucleus	65.0K
DL/I reentrant modules*	90.0K
OSAM reentrant modules*	4.5K
ISAM reentrant modules*	7.8K
Control facility system tasks	8.0K
BTAM reentrant modules*	10.6K
BTAM 3270 local**	5.8K
Total	191.7K

*These modules may reside in pageable link pack area.

**In addition to BTAM modules required for 3270 local.

In an OS/VS1 or OS/VS2 environment, ISAM and OSAM may be replaced by VSAM. The storage requirements for all reentrant modules may be obtained from the pageable link pack area. The virtual storage requirements would otherwise range from 16.3K to 100.5K, depending upon processing options specified.

For information on VSAM storage requirements, see the appropriate VSAM Storage Estimates manual.

2. BTAM device support (include each desired requirement one time)

1050 Autopoll	282 bytes
1050 non-Autopoll	248 bytes
1050 switched	244 bytes
2740-1 Non-Station Control	248 bytes
2740-1 switched	304 bytes
2740-1/2 Autopoll	232 bytes
2740-1/2 non-Autopoll	240 bytes
2741 switched or nonswitched (basic)	78 bytes
2741 switched (in addition to basic 2741 requirement)	160 bytes
2741 nonswitched (in addition to basic 2741 requirement)	128 bytes
BSC1 (bisync point to point)	296 bytes
BSC3 (bisync multipoint)	328 bytes
2980	296 bytes
3270 local	120 bytes
33/35 Teletypewriter (ASR)	176 bytes
System/7 Non-Station Control	248 bytes
with Autopoll	232 bytes
non-Autopoll	240 bytes

2A. BSAM device support (include if local card reader/SYSOUT support is present in the IMS/VS system).

3211 printer	1114 bytes
Any supported RPS device	450 bytes
Spooled SYSOUT support (non-RPS)	848 bytes
Spooled SYSOUT support (RPS)	1298 bytes

Note: If an RPS device is used for direct output, spooled SYSOUT using RPS devices requires only the storage necessary for non-RPS devices.

3. IMS/VS terminal modules (include each desired requirement one time)

2740-1	600 bytes
2740-1 (without Station Control feature)	550 bytes
2740-1 switched	700 bytes
2740-2	1600 bytes
2741 nonswitched	550 bytes
2741 switched	1700 bytes
1050	1400 bytes
1050 switched	1200 bytes
2770 (basic)	6500 bytes
2770 (in addition to basic 2770 req't if MDI (050) attached)	2200 bytes
2780	2100 bytes
2980	3700 bytes
2980 input edit routine (IMS/VS supplied)	1000 bytes
33/35 Teletypewriter	1000 bytes
7770	1800 bytes
3270 local	4800 bytes
3270 remote	2600 bytes
Local Card Reader/SYSOUT	2500 bytes
System/3 and System/7 common	4000 bytes
System/3	2200 bytes
System/7	2600 bytes

4. Conversational processing (include if required) 16000 bytes

4A. Message formatting service for 3270 support (include one time if system contains 3270 support) 5000 bytes

5. Control blocks (include each desired requirement per stated IMS/VS resource)

Each region (msg or batch msg)	600 bytes
Each program (msg or batch message)	40 bytes
Each data base	38 bytes
Each transaction code	56 bytes
Each communication logical terminal	50 bytes
Each communication line	120 bytes
Each communication physical terminal	32 bytes
Each BTAM line group	192 bytes
Each 3270 local line group	264 bytes
Each 3270 terminal	52 bytes
Each local reader line	120 bytes
Each direct output line	120 bytes
Each spooled SYSOUT line (one D/S)	120 bytes
Each additional data set per spooled SYSOUT line	92 bytes

Each System/3	116 bytes
Each System/7	116 bytes

6. Minimum buffer requirements

Message queue buffer pool =	1.5K
General buffer pool =	3.0K

Communication line buffer pool

System console	270 bytes
For each active 2740-1 line	160 bytes
For each active 1050 line	160 bytes
For each active 2740-2 line	450 bytes
For each active 2780 line	410 bytes
For each active 2741 line	160 bytes
For each active 2770 basic buffer line	148 bytes
For each active 2770 buffer expansion line	276 bytes
For each active 2770 additional buffer expansion line	532 bytes
For each active 2980 line	120 bytes
For each active 7770 line	50 bytes
For each active 3270 local line	306 bytes (Notes 1 and 3)
For each active 3270 remote line	344 bytes (Note 3)
For each active reader line	100 bytes
For each active direct output line	See Note 2
For each active spooled SYSOUT line	See Note 2
For each active System/3 line	User-specifiable
For each active System/7 line	User-specifiable

Note 1: Specified at system definition. May be dynamically increased during system operation.

Note 2: Buffer size is user defined and is equal to
DCB blocksize + 16 is RECFM=V or
DCB blocksize + 20 is RECFM=F.

Note 3: Space requirements vary dynamically during system operation, based on requirements of user-defined message formats.

Data base control block pool

For each open data base (Note 1, below)	1200 bytes

Program control block pool

For each message or batch message program per data base used (Note 2)	2K

Data base buffer pool

For each message address space (Note 3)	7K

Message Format Pool

(Include if system provides 3270 support.) For each concurrent 3270 message (Note 4)	1K

Message Format Block Pool

(Include if system provides 3270 support.)	
For each 3270 communication line	500 bytes

7. OS requirements within IMS/VS control address space

IOBs and channel programs required per open data base	1050 bytes
OS/VS Modules And Control Blocks	14K

Note 1: This represents space for a control block associated with each data base. Space may be allocated for a subset of the total number of data bases. However, enough space should be allocated for control blocks associated with frequently used teleprocessing data bases.

Note 2: This represents space for a control block associated with each message program. Space may be allocated for a subset of the total number of message program blocks. However, enough space should be allocated for control blocks associated with message processing programs used to process high-priority messages.

Note 3: This includes buffers for each data base used by the message region. A practical minimum for each message processing address space is two buffers, equivalent in length, for the data set having the largest physical blocks. Additional allocation of buffers will provide greater system performance.

Note 4: This pool contains message format description blocks which vary in size and number depending upon the complexity of individual 3270 message format and the number of transaction codes using the service. Sufficient space should be allocated to maintain concurrently required blocks resident.

Based on the above storage estimates, with 20 message programs, 40 transaction types, 5 of 7 defined data bases open, no conversations, fifty-four 2740 communication lines (maximum of 20 simultaneously active), and 10 message program blocks resident, the typical IMS/VS configuration shown in Figure 19 can utilize three message processing address spaces and one batch processing address space.

If Data Language/I data base batch standalone is considered, the storage estimates may be figured in either of two ways:

1. Place in basic link pack or RAM the modules of QISAM, VSAM, and OSAM and most of the IMS/VS Data Language/I modules. The storage estimate for this is approximately 220K. If this has been done, any address space can concurrently use the Data Language/I facilities of IMS/VS. For nonreentrant code, 12K must be added to each address space using Data Language/I. Additionally, data base buffers must also be added to each address space.

A buffering technique is used that allows sharing of buffers for one or more data bases.

Examples:

Selected programming system storage requirements

+ Basic RAM or link pack requirement
+ 90K for IMS/VS
+ 12K per each address space for nonreentrant code
+ Data base buffers
+ Main storage requirements of application program(s)

Total

This total is the storage estimate for Data Language/I data base batch standalone.

2. QISAM, OSAM, and most of the IMS/VS Data Language/I modules are not placed in the link pack or RAM area. Therefore, each address space waiting to use IMS/VS Data Language/I must add 90K plus 12K plus main storage requirements for the application program to determine the size of each address space.

3. Note that in the OS/VS1 and OS/VS2 systems, except at user option, the link pack area allows overcommitment of real storage. It is therefore advantageous to have these modules in the RAM (RAMF) area.

CHAPTER 9. TYPE I PROGRAMMING SYSTEMS USAGE

IMS/VS operates under OS/VS1 and OS/VS2 and is written in Assembler Language. The teleprocessing and batch processing application programs may be written in either Assembler Language, COBOL, or PL/I. For proper execution of IMS/VS, system definition and system execution must be performed under the same operating system release. Unless required by the operating system, a new IMS/VS system definition does not make it necessary to recompile user application programs.

The batch-only system uses Sequential Access Methods, Indexed Sequential Access Methods, and the Virtual Storage Access Method.

OPERATING SYSTEM/VIRTUAL STORAGE

The teleprocessing and related batch system operates under OS/VS1 or OS/VS2, and, in addition to the items above, uses:

> BTAM (with Communication Serviceability Facilities), VSAM, SER1 or Recovery Management Support, Sort/Merge service program (used by the IMS/VS System Log Utility Program), user-added SVC routines (three SVC numbers must be reserved for IMS/VS)

In addition, Resident Access Methods are highly recommended.

CHAPTER 10. APPLICATION SAMPLES

The General Information Manual for IMS/360 Version 2 presents a description of several applications. The same approach is employed here to provide illustrations and guidance for a broader audience. This chapter presents several examples of the types of terminal inquiry and update transactions that a user might employ. In addition, the data base record structures and organizations necessary in providing efficient inquiry and update processing are described. Examples are taken from several industries, but these are by no means to be considered all-inclusive. The reader may wish to work out other examples from his own industry. The examples are described by considering:

- The information that a terminal operator might want or that a report might contain
- The logical data structure necessary to supply the desired information
- The most suitable data organization and access methods

The IMS/VS data base concept allows for the correction of a data base, using the traditional manner of system design presented above. However, IMS/VS provides the ability to add new segment types into existing data bases as well as the ability to create new data bases with minimal impact to the system user. This data base approach to system design promotes evolutionary system development.

MANUFACTURING INDUSTRY

The following list of questions represents typical requests for information in the manufacturing industry:

- What parts represent the component parts of an assembly, and in what quantity is each component part required?
- Where is a given part used in the composition of assemblies, and in what quantities is the part used?
- What is the inventory status of a part where multiple inventory sites exist?
- What open purchase orders exist for a given part and who is the vendor supplying the part?
- What work orders exist for a given part, and what is the status of each work order?
- What are all the open purchase orders and work orders?
- What operations are performed in the construction of a particular part, and at what work centers are these performed?
- What operations are performed at a particular work center, and what parts are affected by these operations?

The first two questions relate to the structure of a product and can be asked for a particular part, a substructure of a product composed of many parts, or an entire product. One question is really the inverse of the other. The logical data structure for answering both questions is shown in Figure 19.

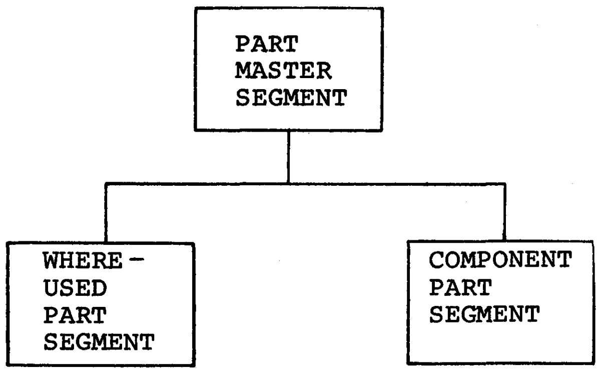


Figure 19. Logical data structure for part data base

However, the where-used information for one part master is the same as the component part information of the part master where it is used. Actually, the where-used information for one part is redundant with the component part information of its assemblies. The pointer segment-target segment concept introduced in Chapter 4 can therefore be employed as shown in Figure 20. The functions of the where-used part segment and the component part segment can be achieved by one segment type. This segment type is called the component part/where-used segment. A one-level bill of material is produced by proceeding from a part master to its component part/where-used segments by physical child and physical twin relationships. One-level where-used information is obtained by proceeding from a part master to component part/where-used segments by logical child--logical twin relationships.

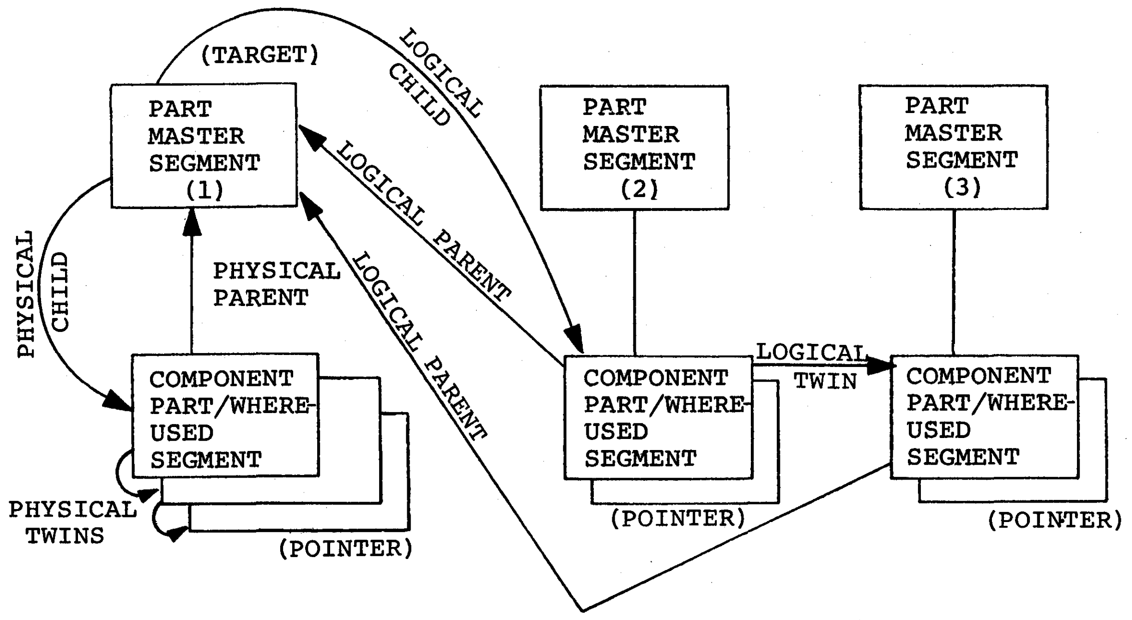


Figure 20. Three interrelated physical data base records for three parts

The part master segment 1 in Figure 20 represents a part that is used as a component for both part master segment 2 and part master segment 3. Figure 21 illustrates a simple hierarchical structure and these relationships. The dependent segment of Figure 21 represents the concatenation of data from the pointer segment and target segment; the data in the pointer segment is intersection data.

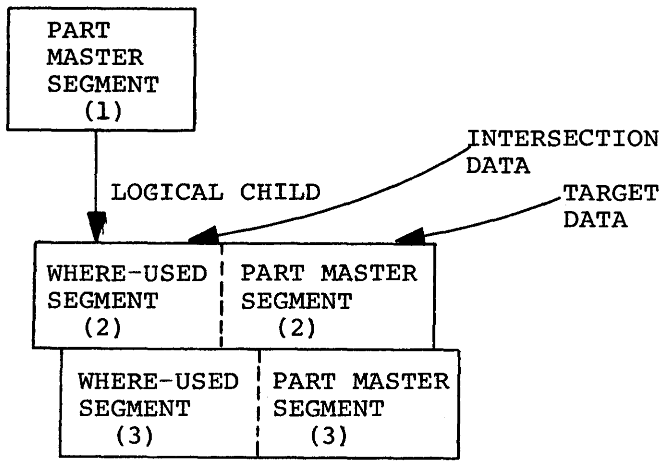


Figure 21. Logical data structure for usage of part 1 of Figure 20

Part master segment 2 in Figure 20 represents a part that is composed of the part described in part master segment 1 in Figure 20 and, let us assume, a part master segment 4. This relationship of component part explosion can again be illustrated by means of a simple hierarchical tree. Notice the concatenation of pointer and target segment data in Figure 22. The physical existence of the component part/where-used segment exists under the part master for which it represents component part data. Therefore, the relationship from the part master 2 to its component part segments 1 and 4 is a physical parent-child relationship. The relationship for part master 1 to its where-used information in the combined component part/where-used segment is by address chains. This is a logical parent-child relationship.

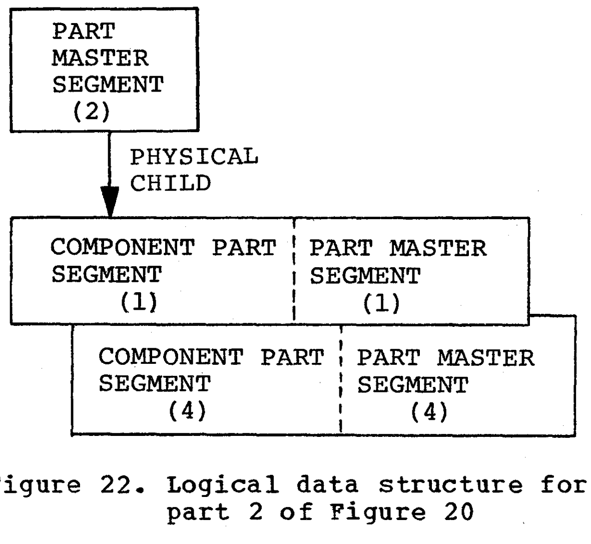


Figure 22. Logical data structure for component part definition of part 2 of Figure 20

The inventory status for a particular part could be supplied by the data structure in Figure 23.

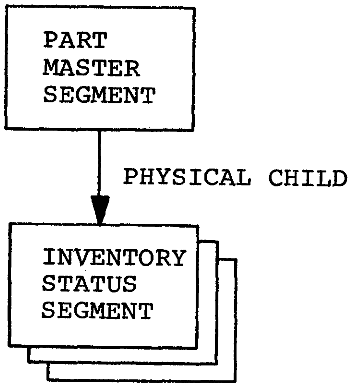


Figure 23. Logical data structure for part inventory status

Each inventory status segment for a particular part is a dependent segment under the part master segment. Each represents the inventory for the part at a particular location.

The open purchase orders and vendors assigned for a particular part could be supplied by the data structure in Figure 24.

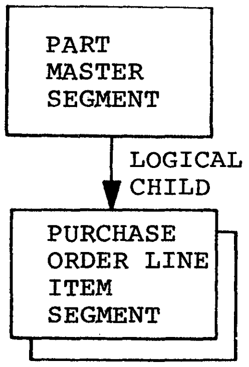


Figure 24. Logical data structure for part purchase order

Each purchase order line item segment represents a line item in an open purchase order pertinent to this part. Of course, the inverse question of what purchase orders have line items that affect a given part might be asked. This problem is similar to the component part/where-used situation and can again be solved with the pointer segment--target segment concept. Let us assume part master segment X in Figure 25 has a line item in two purchase orders, A and B.

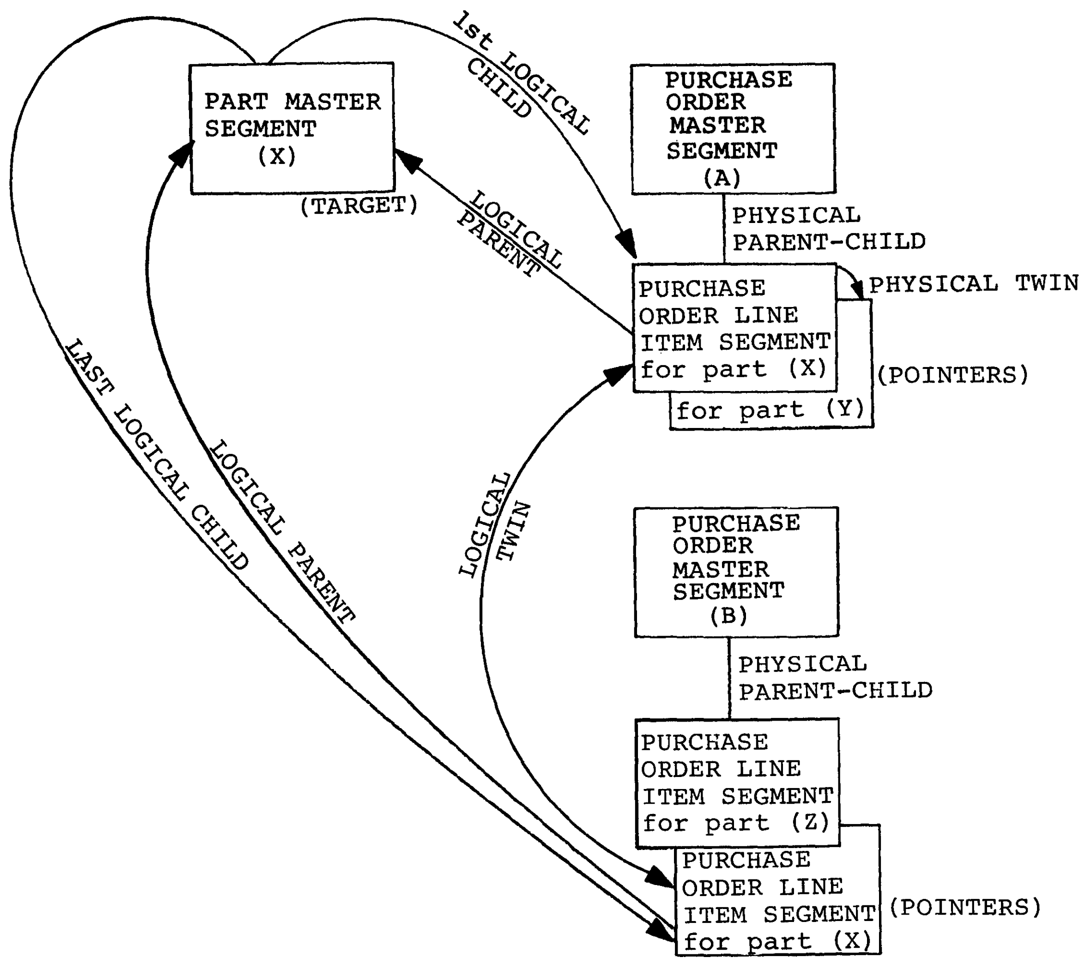

PART MASTER SEGMENT (X)
(TARGET)
1st LOGICAL CHILD
PURCHASE ORDER MASTER SEGMENT (A)
PHYSICAL PARENT-CHILD
LOGICAL PARENT
PURCHASE ORDER LINE ITEM SEGMENT for part (X)
PHYSICAL TWIN
(POINTERS)
for part (Y)
LAST LOGICAL CHILD
LOGICAL PARENT
LOGICAL TWIN
PURCHASE ORDER MASTER SEGMENT (B)
PHYSICAL PARENT-CHILD
PURCHASE ORDER LINE ITEM SEGMENT for part (Z)
PURCHASE ORDER LINE ITEM SEGMENT for part (X)
(POINTERS)

Figure 25. Physical data base records with pointer segment--target segment concept for part purchase orders

The next questions, regarding what operations are performed in the fabrication of a part and what operations performed at a given work center affect a given part, again involve inverses. These can be answered by the two logical data structures in Figure 26.

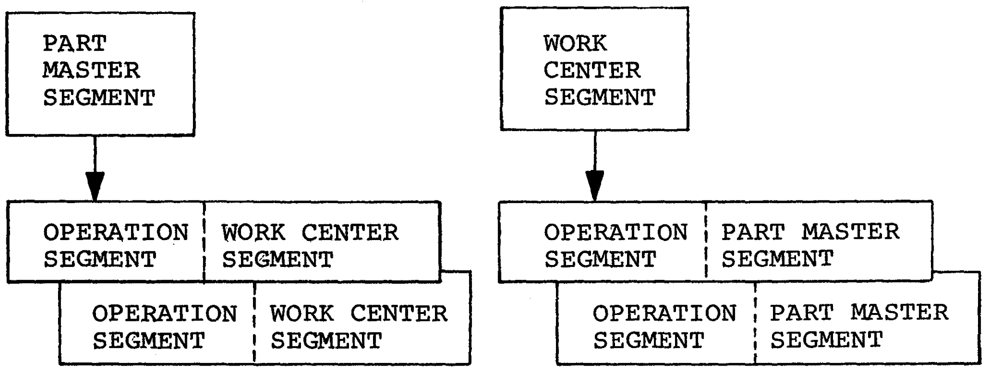


Figure 26. Two logical data structures showing fabrication operations

Using the pointer segment--target segment concept, Figure 27 illustrates how data redundancy is removed. Let us assume that part master A has operations 1, 2, and 3 performed in its fabrication at work centers X, Y, and Z, respectively.

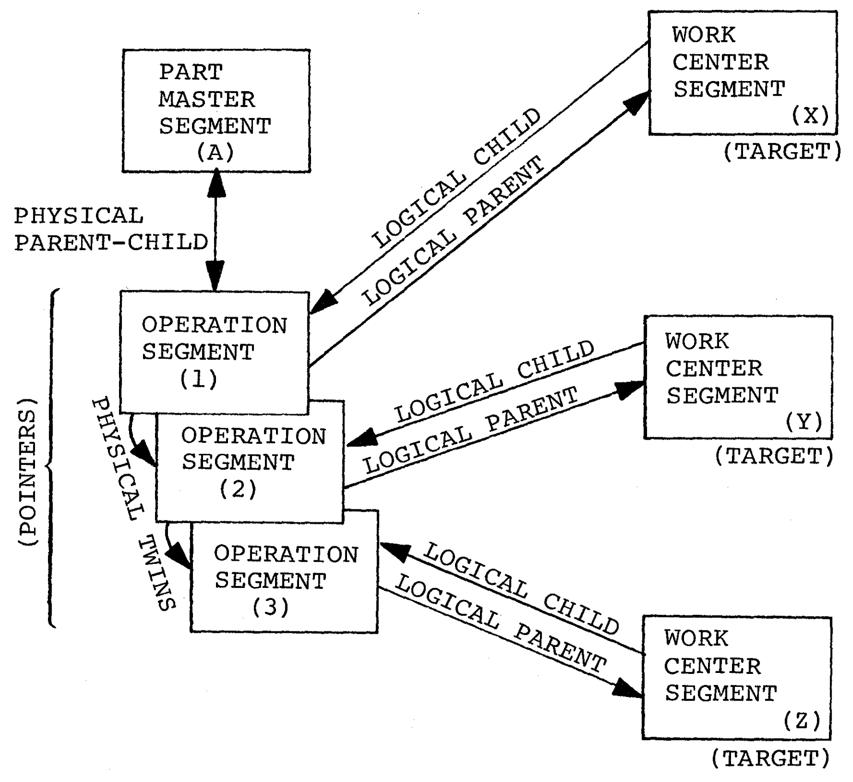


Figure 27. Pointer segment--target segment concept showing elimination of data redundancy

The operation segment under other part masters, where the operation is performed at work center Y, would be logical twins of the operation segment 2 under part master A. By following the logical child relationship from work center segment Y to operation segment 2, and then following the logical twin relationship to related operation segments, the question of all parts affected by operations at a particular work center can be answered.

The following segment relationships can now be formulated into physical and logical data bases. Figure 28 illustrates the physical data base records.

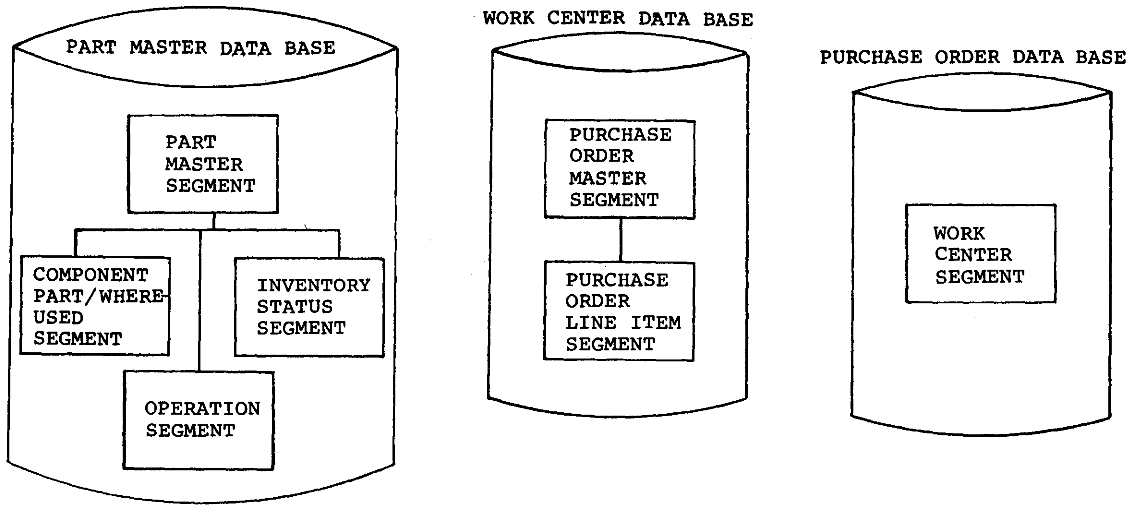


Figure 28. Physical data base records under pointer segment--target segment concept

These three physical data bases are interrelated by the pointer segment--target segment concept as shown in Figure 29. The interrelationships as well as the physical data base records allow logical data base records to be described as indicated in the foregoing discussions.

Let us assume the hierarchical direct organization and HIDAM are chosen for the part master and work center structures. Also assume the hierarchical sequential organization and HISAM are chosen for the purchase order structure. Figure 29 illustrates the physical data bases. While the examples use ISAM and OSAM for access methods, VSAM data sets could be selected.

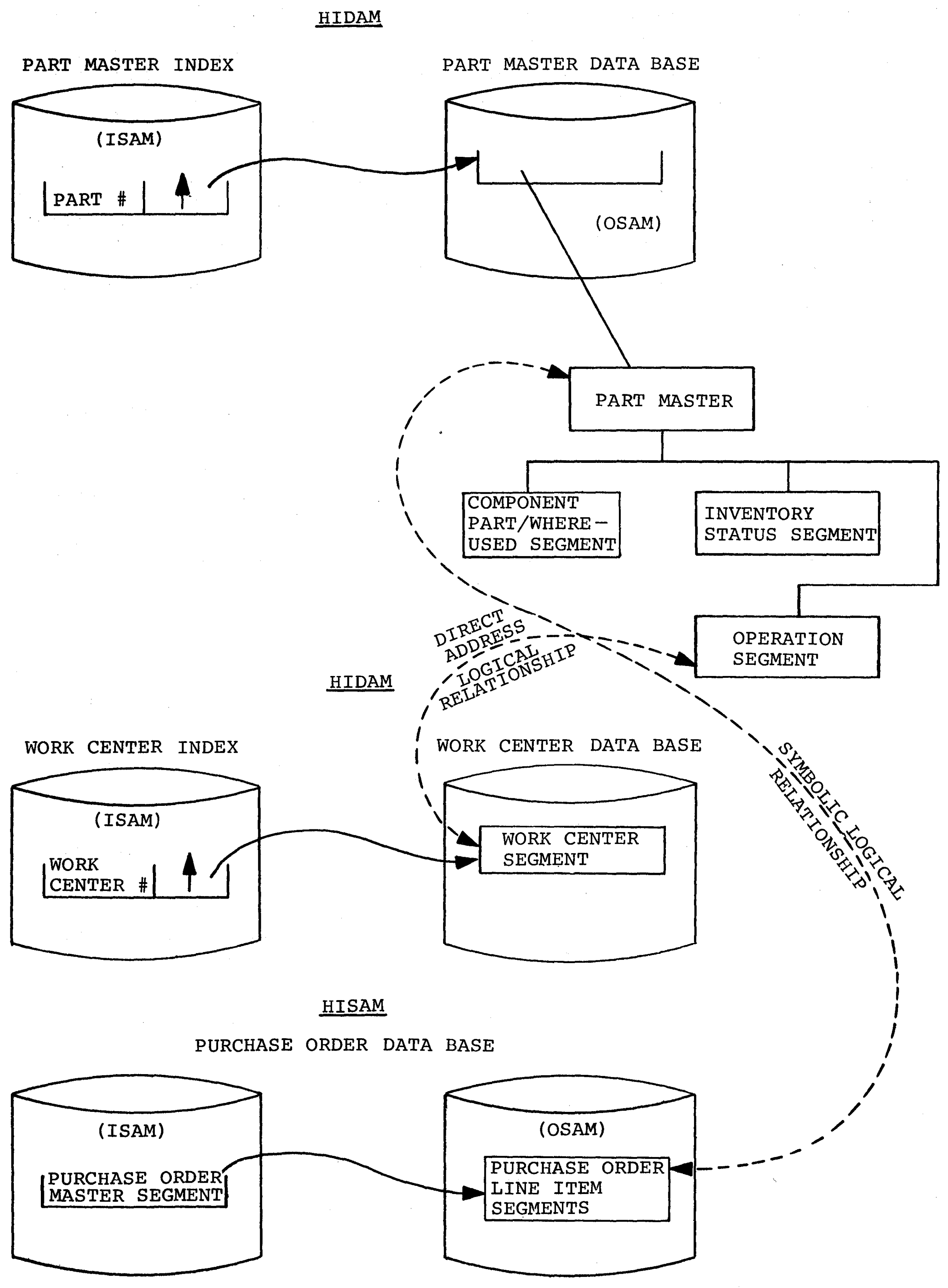


Figure 29. Physical data bases under HIDAM and HISAM

Since the organization chosen for the physical data bases for part master and work centers is hierarchical direct, the logical interrelationships can be accomplished by address chains or symbolic identifiers. However, the organization chosen for the physical data base storage of purchase orders is hierarchical sequential. Here, the logical interrelationships with the part master must be symbolic identifiers.

Although the use of direct address relationships provides greater processing efficiency, simplicity in data reorganization is sacrificed. The needs of a particular user will dictate whether symbolic or direct address relationships should be utilized. The pointer segment (component part/where used) would probably contain intersection data, such as the quantity used. This intersection data may be concatenated with its target segment data as depicted in Figure 21.

FINANCIAL INDUSTRY

The following list of questions represents typical requests for information in the banking environment:

- What are all the accounts associated with a particular individual? This information might be desired when a customer wants to make a deposit but does not know his account number.
- What is the status of loans outstanding to a particular individual? This information might be desired when a bank officer is asked to accept a loan request from a customer.
- Does the checking account of a known account number contain a balance adequate to cash a check?
- What is the amount of money needed to pay off the installment loan for a known loan account?
- What is the property held and what is the par value for each property held in a trust account?
- What trust accounts hold a particular property, such as a particular stock? The additional question might be asked, What is the quantity in shares held of the stock in each trust account?

These questions and the subsequent data structures should stimulate the reader to consider other questions and additional data elements and structures. Both query and update operations against the data base are possible.

The first question, concerning all accounts associated with a given individual, could be answered with the logical data structure of a customer information record in Figure 30.

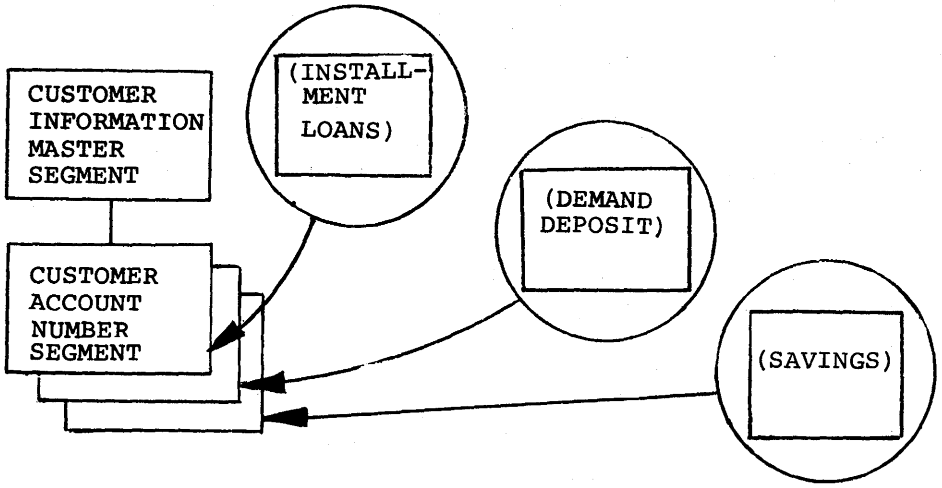

CUSTOMER INFORMATION MASTER SEGMENT
CUSTOMER ACCOUNT NUMBER SEGMENT
(INSTALL-MENT LOANS)
(DEMAND DEPOSIT)
(SAVINGS)

Figure 30. Logical data structure of a customer information record--financial

A customer information master segment would exist for each of the bank's customers and might contain name, address, dates, codes, and a customer-identifying key. The customer account number segment might be keyed on account number and indicate account type and relationship to individual customer.

The second question, concerning the outstanding loans for an individual, can be answered with the same data structure. One or more of the customer account number segments could represent loan accounts. There might also be other segments for each type of account (for example, instalment loans, demand deposit, savings, mortgages, and commercial loans).

The answer to the third question, concerning the current balance in a checking account, can best be answered with a data structure organized by account number. However, we have already established a customer account (number) segment subordinate to customer information master segment in Figure 30. IMS/VS gives the ability to enter the data base either by customer name or account number. This may be approached with the use of the pointer segment--target segment capabilities. Figure 31 illustrates the necessary data structure.

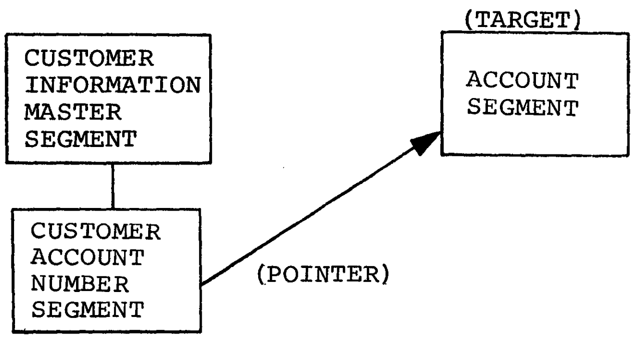

CUSTOMER INFORMATION MASTER SEGMENT
CUSTOMER ACCOUNT NUMBER SEGMENT
(POINTER)
(TARGET)
ACCOUNT SEGMENT

Figure 31. Data structure with pointer segment--target segment relationship

Referring to Figure 31, the pointer segment--customer account number segment--is subordinate to customer information master segment. It contains only a pointer to the applicable account segment and the data describing the relationship between the customer and the account. The account segment with its dependent segments contains most of the information about the account including current balance. The backward relationship from target segment to its associated pointer segments allows the specification of logical data base structures illustrated in Figure 32.

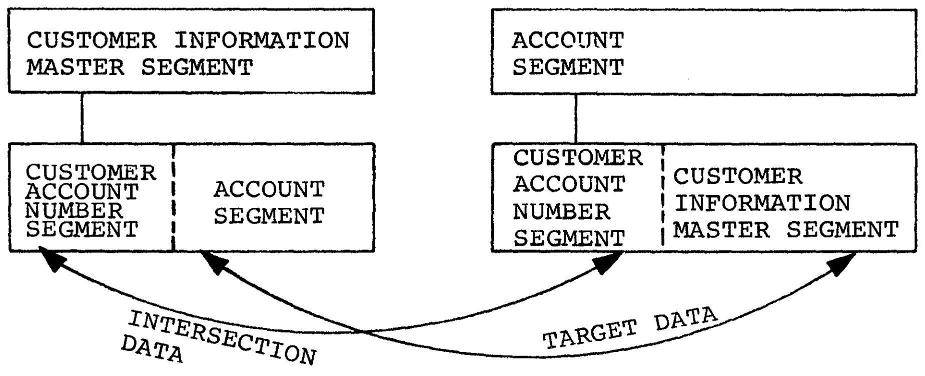

CUSTOMER INFORMATION MASTER SEGMENT
ACCOUNT SEGMENT
CUSTOMER ACCOUNT NUMBER SEGMENT
ACCOUNT SEGMENT
CUSTOMER ACCOUNT NUMBER SEGMENT
CUSTOMER INFORMATION MASTER SEGMENT
INTERSECTION DATA
TARGET DATA

Figure 32. Logical data base structures showing customer information specifications

Notice that the customer account number segment (pointer) is limited in data content to data unique to the relationship between a given customer and a particular account. This represents intersection data.

An example of the physical data base records and the logical interrelationships obtainable is depicted in Figure 33.

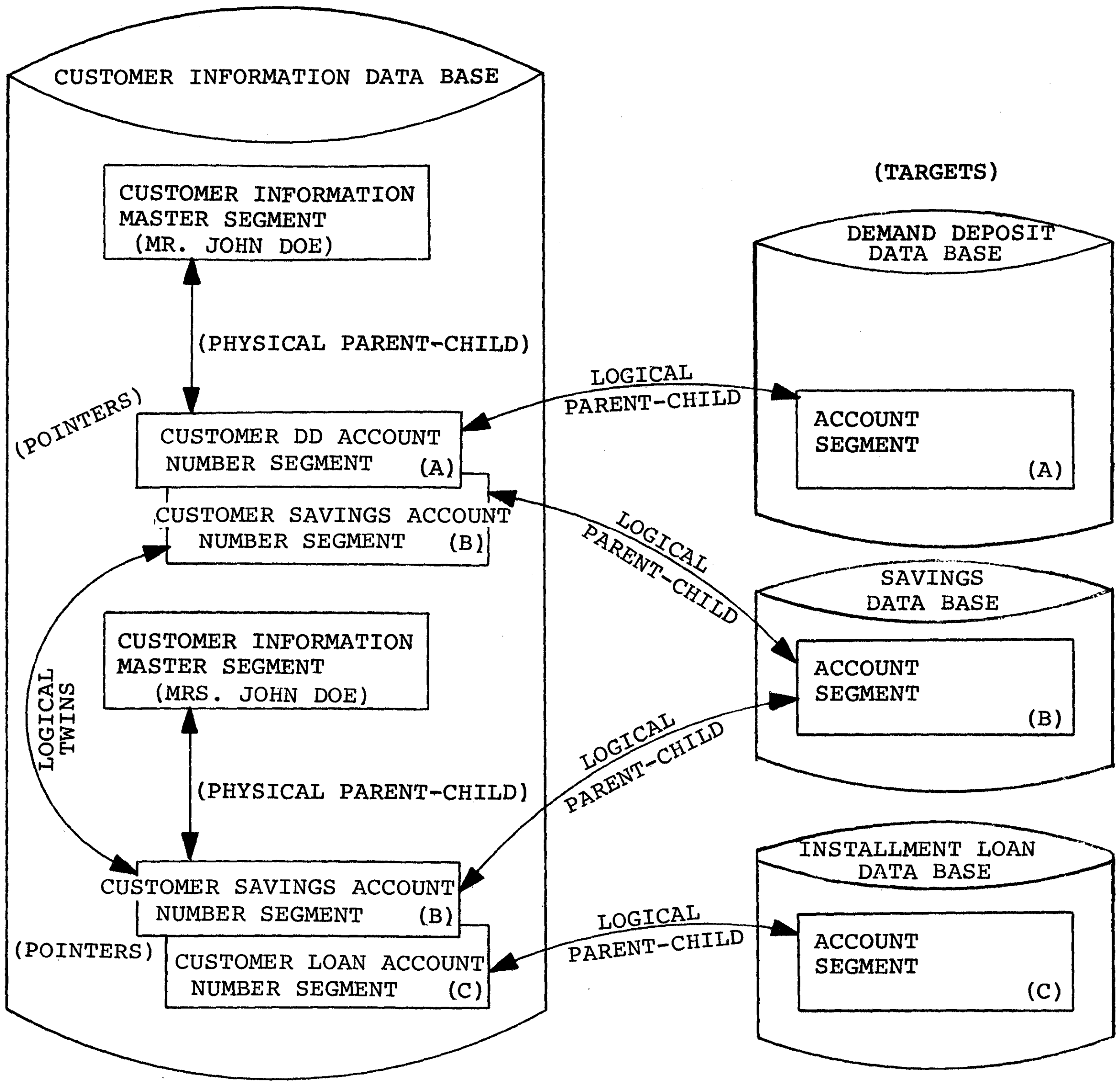

CUSTOMER INFORMATION DATA BASE
CUSTOMER INFORMATION MASTER SEGMENT (MR. JOHN DOE)
(PHYSICAL PARENT-CHILD)
(POINTERS)
CUSTOMER DD ACCOUNT NUMBER SEGMENT (A)
CUSTOMER SAVINGS ACCOUNT NUMBER SEGMENT (B)
LOGICAL TWINS
CUSTOMER INFORMATION MASTER SEGMENT (MRS. JOHN DOE)
(PHYSICAL PARENT-CHILD)
CUSTOMER SAVINGS ACCOUNT NUMBER SEGMENT (B)
(POINTERS)
CUSTOMER LOAN ACCOUNT NUMBER SEGMENT (C)
(TARGETS)
DEMAND DEPOSIT DATA BASE
LOGICAL PARENT-CHILD
ACCOUNT SEGMENT (A)
LOGICAL PARENT-CHILD
SAVINGS DATA BASE
ACCOUNT SEGMENT (B)
LOGICAL PARENT-CHILD
INSTALLMENT LOAN DATA BASE
LOGICAL PARENT-CHILD
ACCOUNT SEGMENT (C)

Figure 33. Physical data base records and logical interrelationships

The questions concerning the properties held in a particular trust account and the trust accounts holding a particular property can be answered with the logical data structures illustrated in Figure 34.

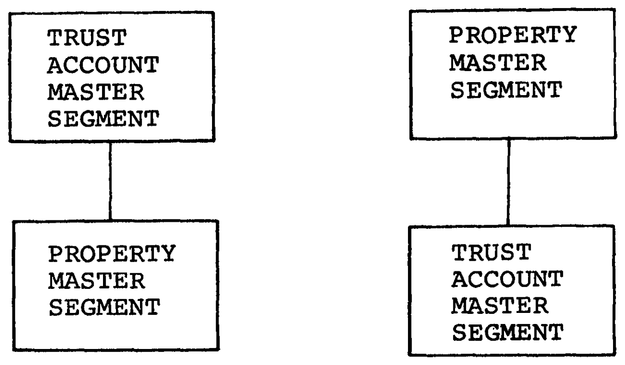


Figure 34. Logical data structures showing properties and trust information

These two questions are answered with inverse data structures and present a problem similar to the customer information master--account master data structures. In addition, there is probably the requirement for intersection data, such as how much (how many shares) of a particular property is held in a particular trust account.

If we utilize the pointer segment--target segment concept, the logical data structure in Figure 34 can be defined as illustrated in Figure 35.

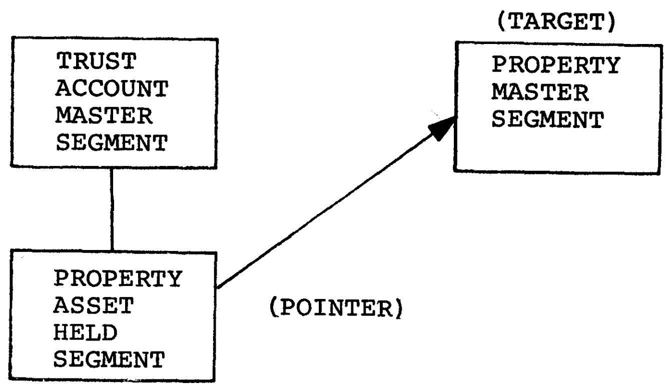


Figure 35. Logical data structure with pointer segment--target segment relationship

The pointer segment--property asset held segment--can contain the intersection data. The logical data base structures that can now be derived are shown in Figure 36.

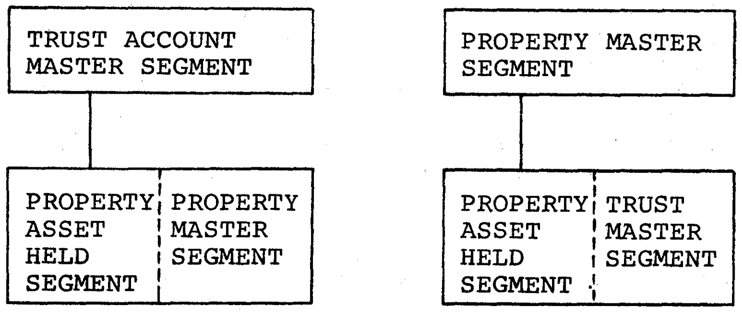

TRUST ACCOUNT MASTER SEGMENT
PROPERTY ASSET HELD SEGMENT
PROPERTY MASTER SEGMENT
PROPERTY MASTER SEGMENT
PROPERTY ASSET HELD SEGMENT
TRUST MASTER SEGMENT

Figure 36. Logical data base structures with intersection data

The physical data base records and the logical relationships for the trust and property data structures are depicted in Figure 37.

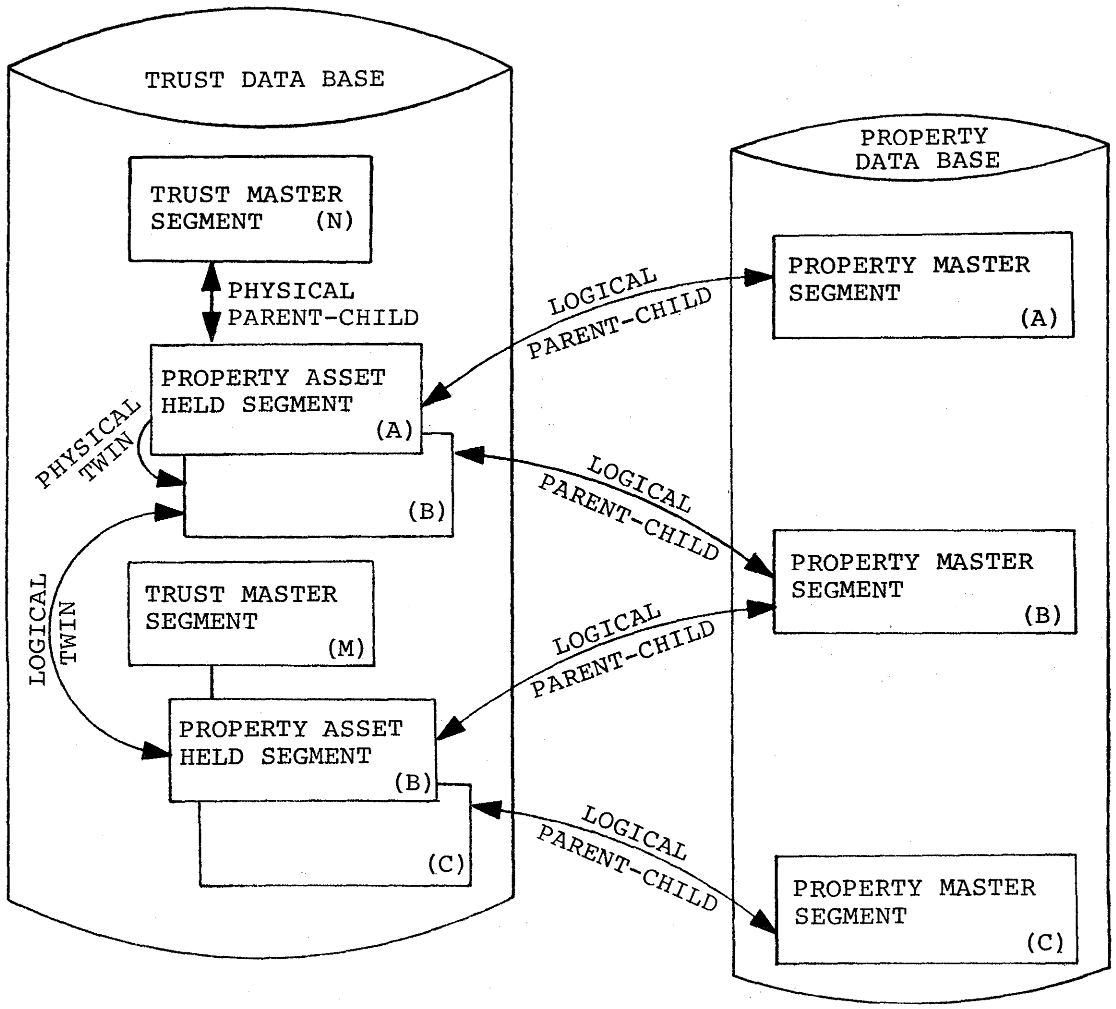


Figure 37. Physical data base records and logical interrelationships

Of course, additional segments can be added to the logical data base structures above that do not participate in pointer segment--target segment relationships. Under the property asset held segment type might exist a log segment type. The lot segment might contain data pertinent to a particular buy or sell of the property, such as date of trade, broker, how acquired, unit price, and total value of trade. Dependent to the account master segment type might be segments that describe deposits and withdrawals on a savings account. Dependent to the property master segment type might be segments that describe different property types, such as stocks, bonds, notes, mortgages, and contracts.

If the hierarchical direct organization and HIDAM are used for storage of the customer information and account data base, Figure 38 illustrates the data base organization and access method.

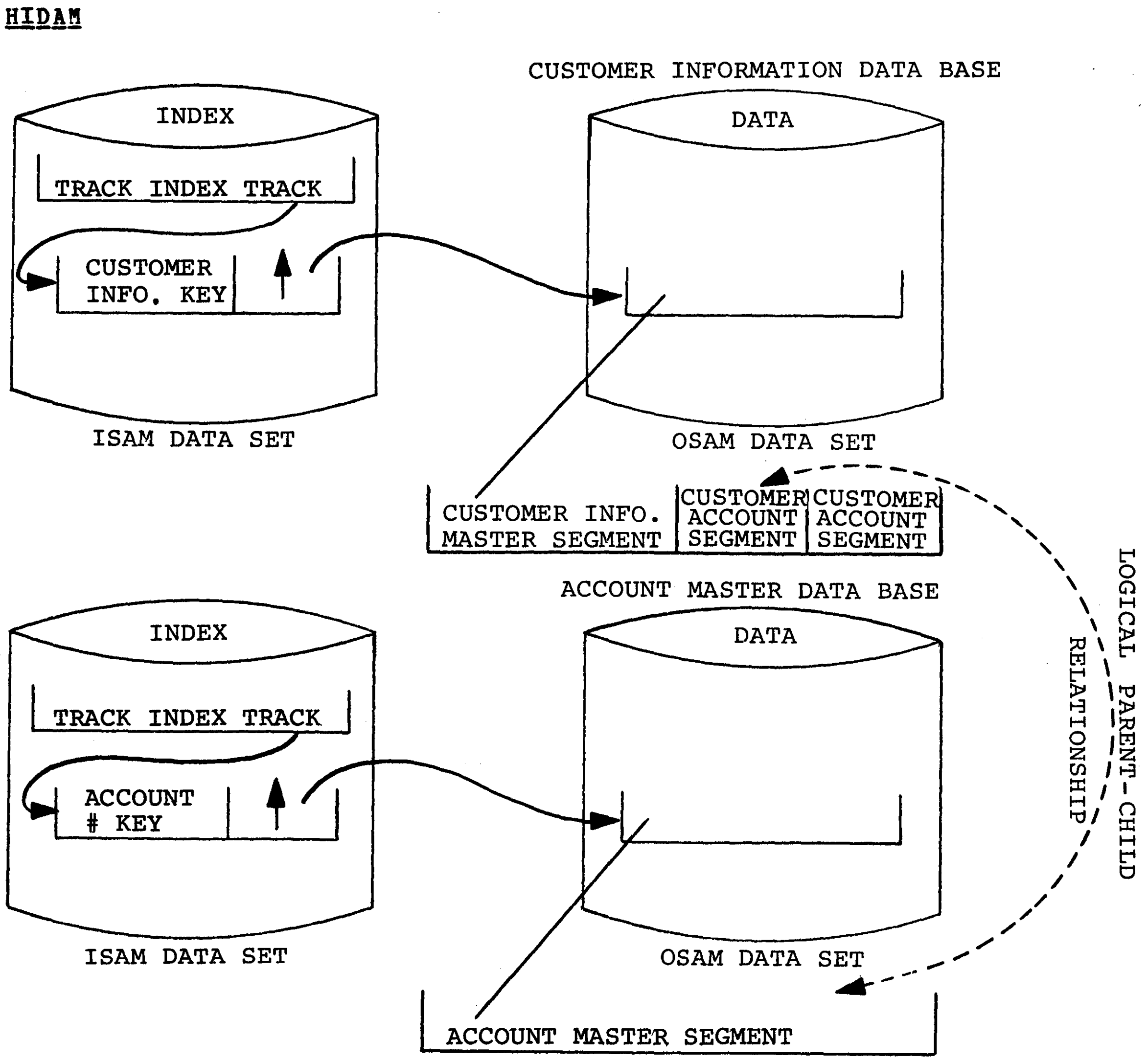


Figure 38. Data bases stored using hierarchical direct and HIDAM

MEDICAL INDUSTRY

In the medical environment a typical use of IMS/VS might be the storage of medical information about patients in a hospital or clinic. Associated with each patient is one data base record with the root segment containing basic information about the patient. This patient master segment may be keyed on patient identification such as social security number. In addition, it contains name, address, age, birth date, sex, and race. (See Figure 39.)

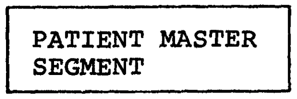


Figure 39. Medical data base record root segment--patient master segment

For each visit to the hospital or clinic, a visit segment might be appended as a dependent segment from the patient master segment, as shown in Figure 40.

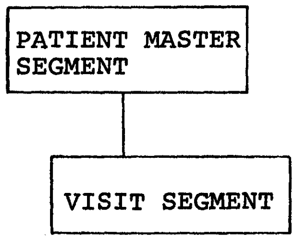


Figure 40. Logical data structure with one dependent segment

The visit segment might contain the date and purpose of the visit as well as the at-ending physician's name. Information obtained during the visit might cause the physician to make a diagnosis of the patient's problem. It is possible to consider a diagnosis segment dependent from the visit segment, as shown in Figure 41.

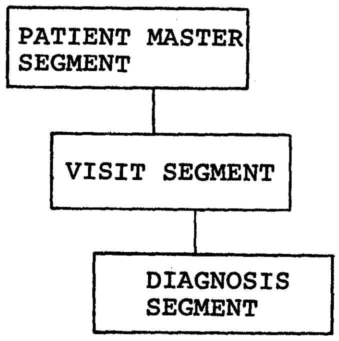


Figure 41. Adding second dependent segment--diagnosis segment--to medical data base record root segment

Certain visits might involve surgical operations performed on a particular site of the human body. Under visit segment may be considered the inclusion of site and surgery segments. An alternative approach may be to consider site and surgery segments directly dependent from the patient master segment.

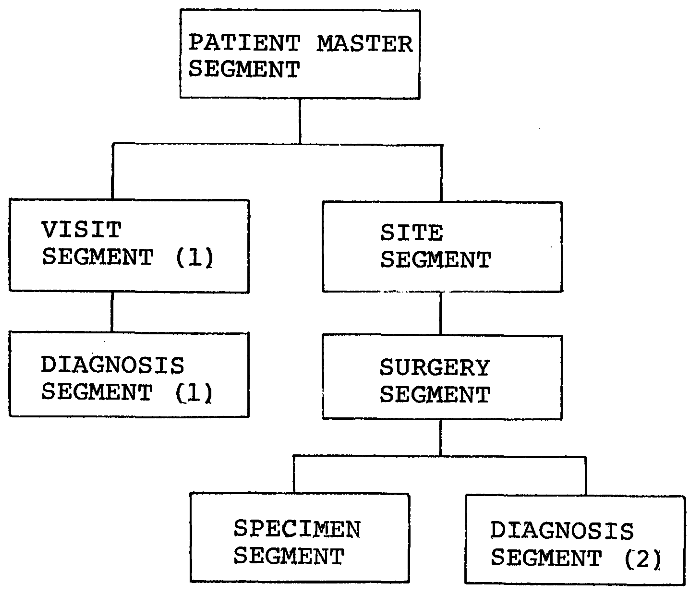


Figure 42. Logical data structure of medical data base record or data base

The second approach is more effective if multiple surgeries were performed on a site at different visits. The relationship between a particular visit and a particular surgery can be achieved through a field in the visit segment relating to a particular surgery.

It would be appropriate to consider specimen and diagnosis segments under the surgery segment. These would be pertinent to a particular surgery.

If the visit to a clinic or doctor did not involve surgery but did involve X-ray or radioactive treatment, additional segment types may be considered as dependent from the site segment. If the visit involved only the application of, or prescription for, a drug or medicine, a drug segment type might be considered dependent to visit segment 1 in Figure 42.

Questions that might be asked of information contained within a data base of the structure depicted in Figure 42 can be of a simple or complex nature. The simple type of question to answer would be a request for information about a particular patient. The answers to questions of this simple type are facilitated because the data base is structured on patient identification sequence. Online terminal inquiry and update are quite practical.

A complex question might involve listing patients or information about patients who received a particular drug, contracted a particular disease, or a combination of such query criteria. The use of pointer-target segment relationships can assist in answering these questions. Additional tree structures can be created that interrelate with the patient tree structure. These tree structures could be associated with drugs, diseases, or other search criteria.

Online Order Entry/Production Control System

A total order entry/production control system has computer control of production from acceptance of orders through manufacturing, shipment, and delivery of the order. An implementation plan for these applications can be developed from the functional relationships that exist between these functions. The plan allows for a logical growth pat-ern through the implementation of online order entry, in-process inventory control, and plant balancing.

One of the major requirements in the installation of such a large-scale system is the ability to implement the individual program in modular increments. Each increment should gradually increase the functional scope of the system without the necessity of reprogramming previously writ-en programs that utilize the same data base. The existing data bases should grow to service the new and additional applications.

A good beginning for a total system is an online order entry application that includes all of the processing operations necessary to accept orders from customers and provide the necessary follow-up until the entire order has been shipped. The initial phase of online order entry would include acceptance of orders for stocked items. A following expansion would place additional information in the data bases that would permit acceptance of orders for items that necessitated initiation of a mill order to produce the item or items requested. When this application is added, it is necessary to add programs that check in-process inventory, operation routing, and facility loading information. It would also be necessary to create mill order plan records. The organization of these required records is shown in the examples.

In addition to the functions performed by the online order entry, the in-process inventory control programs would provide basic material control and order status. Plant balancing programs that would balance long-range scheduling objectives with short-range sequencing objectives in order to optimize production in relation to customer orders could subsequently be added to the application.

If the data bases illustrated and the programs described were implemented, the following types of questions could be answered on communication terminals: What is the availability of a stocked product requested by a customer? What ship date can be promised for an item that requires manufacturing? What is the credit limit of a given customer and what is the total amount of unpaid invoices? What is the status of an existing customer order?

In addition, many changes to the information contained in the data bases can be entered from the communication terminals. This would include the following types of transactions:

- Enter receipt of new stock
- Provide notification of low limits of inventory
- Add new items to an existing order
- Change the quantity previously entered in a customer order
- Change the ship-to location of an existing order

The following logical data structure (Figure 43) can be used as a base to install the described applications, permit the entry of data as listed, and answer the types of inquiries described.

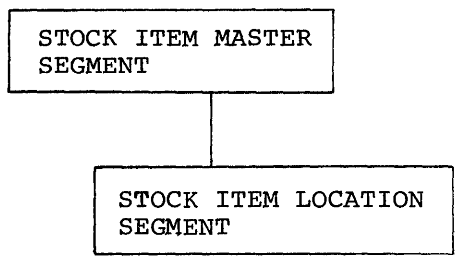


Figure 43. Logical data structure for stock item data base

Figure 43 illustrates a logical data structure in which a stock item master segment exists in a data base for each stocked item. Subordinate to a stock item master segment, one or more stock item location segments exist. These segments describe inventory locations where a given quantity of the stocked items exists and is available.

The question concerning the ability to fill an order from stock can be answered with the data structure in Figure 43 by making inquiries against the described stock item records.

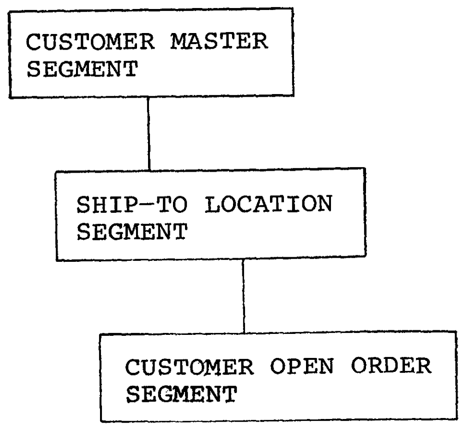


Figure 44. Logical data structure for customer master data base

The ability to do a credit check on a particular customer requires availability of information on each customer. Let us consider the logical data structure in Figure 44. The customer master segment is keyed upon a unique customer identifier and contains information such as credit clearance level, name and address, and total amount of unpaid invoices. This segment provides the answer to the credit check. In addition, this customer may have one or more locations to which he wishes the orders to be shipped. The ship-to location segments provide this information. The customer open order segments indicate all open orders for a particular customer and a particular location. These segments provide the pointers to the details of each open order. This open order data structure describing each open order, Figure 45, is required in addition to the stock item and customer information data structures.

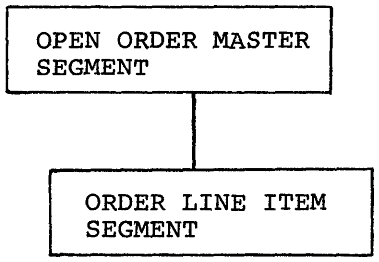


Figure 45. Logical data structure for open order data base

This logical data structure would contain an open order master segment, which is keyed upon order number. Subordinate to the open order master segment are one or more segments describing each line item in an order.

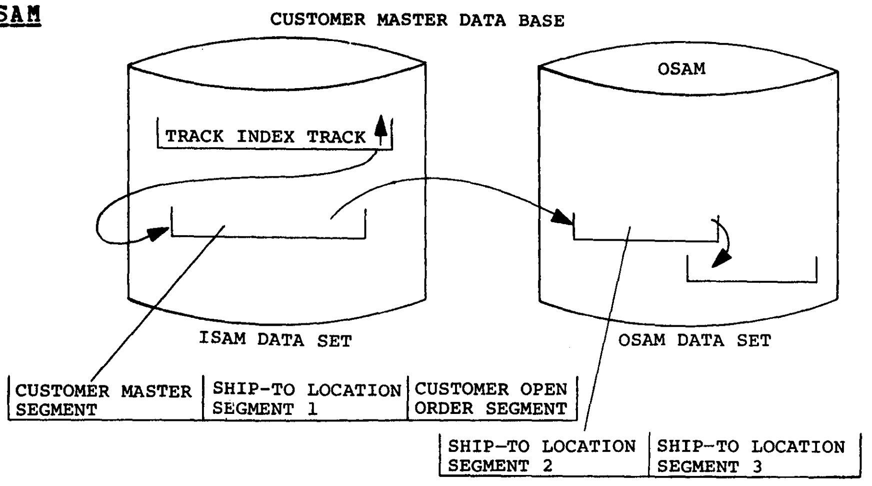
The open order master segment contains the status of the order, the due date, the customer name, and the customer's ship-to location for the order. Using this logical data structure and the stock item data structure previously discussed, order status inquiries can be answered.

Since the customer master segment and ship-to location segment data structure represents stable data, the user may select the hierarchical sequential organization and HISAM.

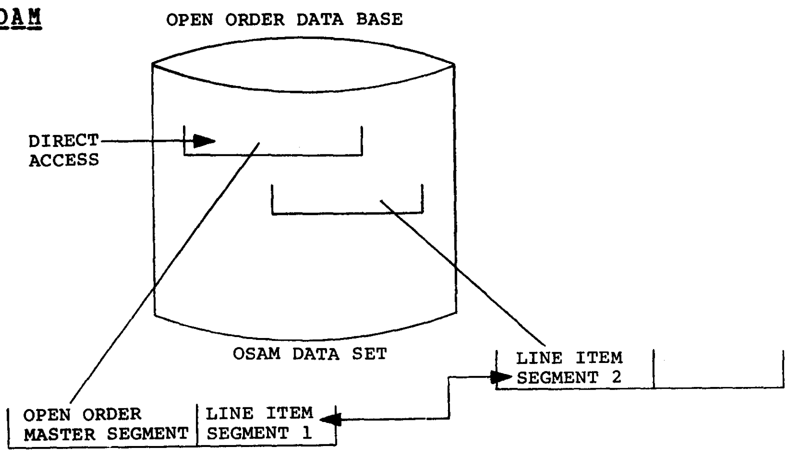
The stock item data structure is more volatile with the updating or inserting and deleting of stock item location segments. The user may select the hierarchical direct organization and HIDAM. One of the reasons for this selection is reuse of deleted segment space with a hierarchical direct organization data base.

The open order data structure is quite volatile. All segments associated with an order exist only for the life of the order. If an order is modified, one or more segments may be updated, inserted, or deleted. In addition, frequent inquiry against the structure may be required for determining order status. Here the user may consider the hierarchical direct organization and HDAM. Figure 46 depicts the three logical data structures stored in the suggested organizations with the indicated access techniques. Although ISAM and OSAM are shown, their use can be replaced with VSAM.

HISAM


CUSTOMER MASTER DATA BASE
OSAM
TRACK INDEX TRACK
ISAM DATA SET
OSAM DATA SET
CUSTOMER MASTER SEGMENT
SHIP-TO LOCATION SEGMENT 1
CUSTOMER OPEN ORDER SEGMENT
SHIP-TO LOCATION SEGMENT 2
SHIP-TO LOCATION SEGMENT 3

HDAM



HIDAM

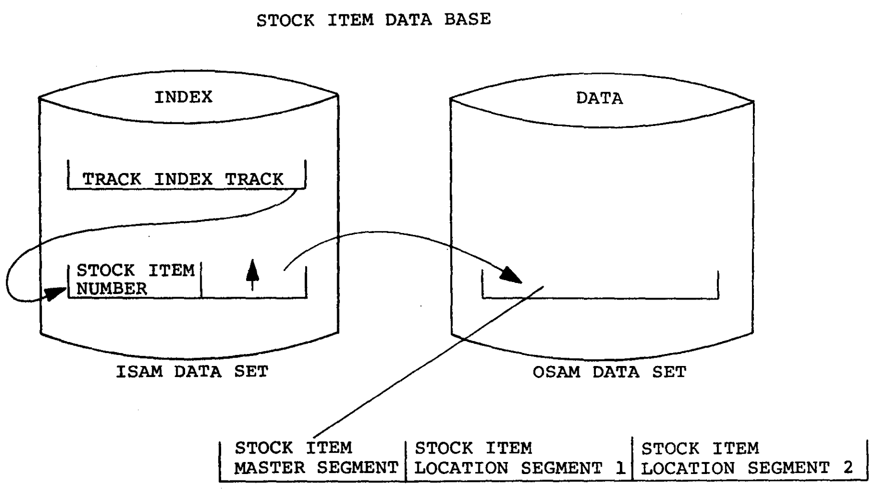


Figure 46. Physical data bases under HISAM, HDAM, and HIDAM

If this application is expanded to include in-process inventory items as well as stocked items for handling orders, the additional questions might be asked:

- What is the status of an open order that required initiation of a mill order to produce the item?
- What is the routing or operations performed in the processing of a given product?
- What is the workload on a given plant facility? If a particular plant facility br eakdown occurs, can another facility be used to complete the order?
- If a plant facility breakdown occurs, what is the effect on an order's status?

In addition to answering questions such as those stated above, the following data structures can allow for data base update processing to assure maximum plant facility usage and minimal time until customer order availability.

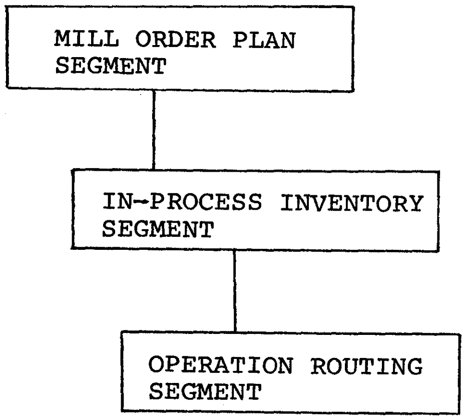


Figure 47. Logical data structure for mill order planning

Figure 47 describes the mill order plan and routing relationship. The mill order plan segment is keyed on mill order item number and includes a plan of manufacture and time schedule. The in-process inventory segment includes the status of a mill order. Each operation routing segment to be under an in-process inventory segment describes an operation performed in the process of producing the mill order item and the facility at which the operation is to be performed.

Figure 48 describes plant facility segment, in-process inventory segment, and mill order plan segment.

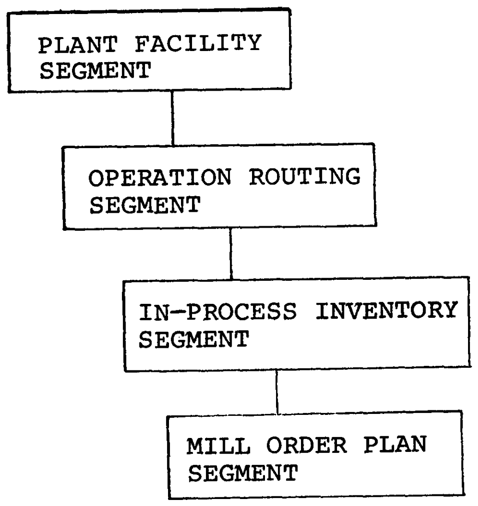


Figure 48. Logical data structure for plant facility

The plant facility segment is keyed on facility number and may contain facility loading data, such as total time scheduled, etc. The operation routing segment contains the individual operations performed on a given mill order item at that facility. Figures 47 and 48 are actually inverse data structures of each other and can be considered for physical storage through the pointer segment--target segment concept. Figure 49 restates Figures 47 and 48 using the pointer-target concept.

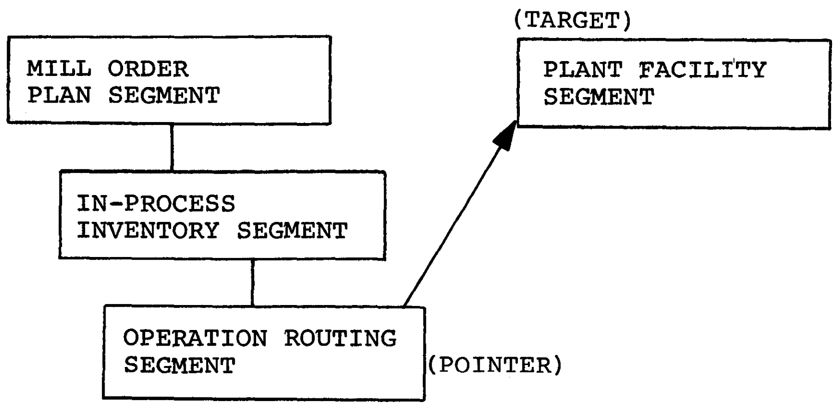


Figure 49. Logical data structure for mill order plan and plant facility using pointer-target concept (Figures 47 and 48)

Referring to Figure 50, consider the mill orders A and B with operations performed at plant facilities X and Y. Figure 50 depicts the physical data base records and logical relationships.

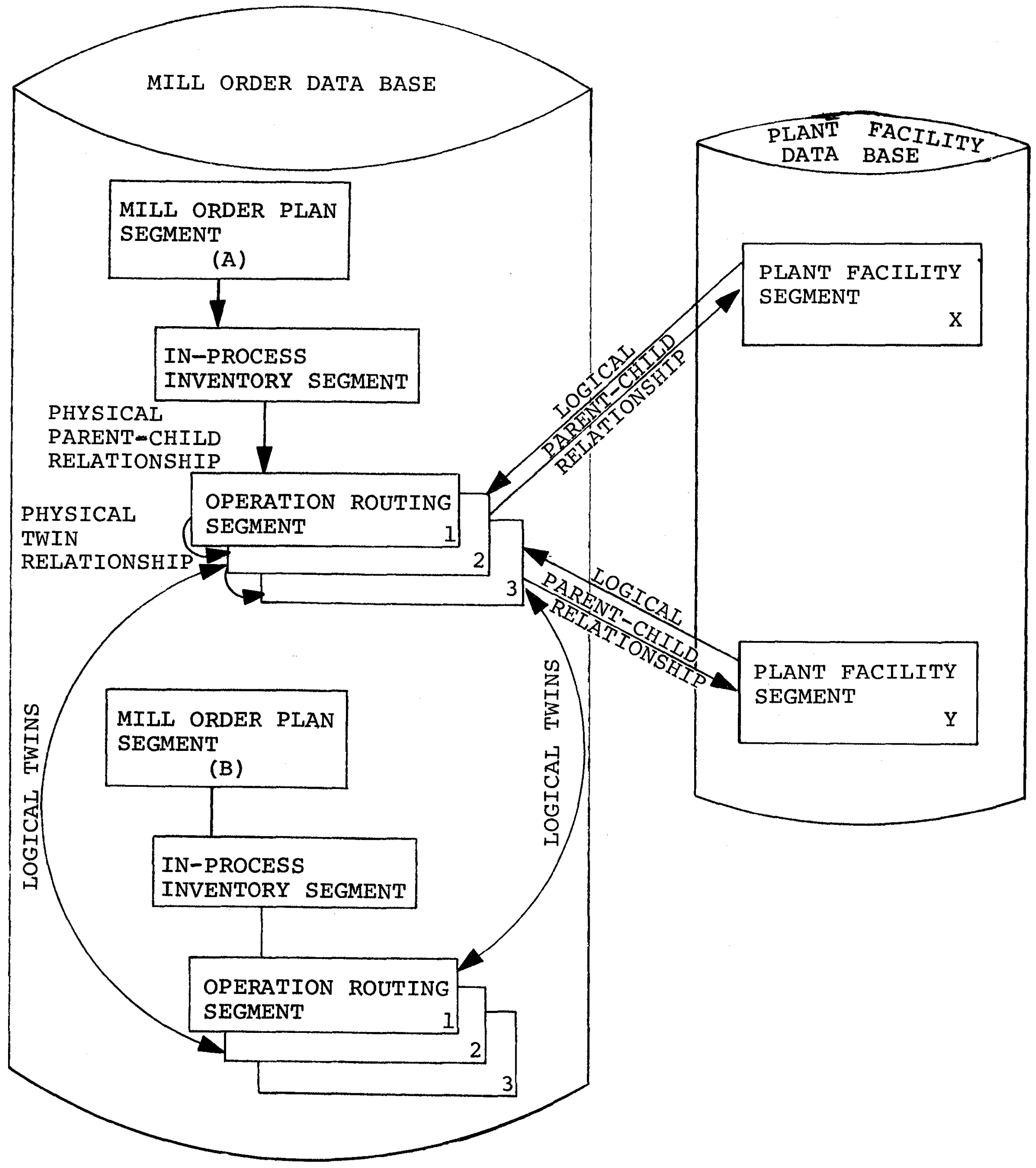

MILL ORDER DATA BASE
PLANT FACILITY DATA BASE
MILL ORDER PLAN SEGMENT (A)
IN-PROCESS INVENTORY SEGMENT
PHYSICAL PARENT-CHILD RELATIONSHIP
PHYSICAL TWIN RELATIONSHIP
OPERATION ROUTING SEGMENT
1
2
3
LOGICAL PARENT-CHILD RELATIONSHIP
PLANT FACILITY SEGMENT X
LOGICAL PARENT-CHILD RELATIONSHIP
PLANT FACILITY SEGMENT Y
LOGICAL TWINS
LOGICAL TWINS
MILL ORDER PLAN SEGMENT (B)
IN-PROCESS INVENTORY SEGMENT
OPERATION ROUTING SEGMENT
1
2
3

Figure 50. Physical data base records and logical relationships--mill order plan and plant facility

In addition to these data structures, a relationship must be stated from the open order data base to the mill order plan segments. This can be achieved with a new segment in the open order data base indicating the pertinent mill order plan segment.

INDEX

READER'S COMMENT FORM

IMS/VS
General Information Manual

GH20-1260-0

Please comment on the usefulness and readability of this publication, suggest additions and deletions, and list specific errors and omissions (give page numbers). All comments and suggestions become the property of IBM. If you wish a reply, be sure to include your name and address.

COMMENTS

fold

fold

fold

fold

• Thank you for your cooperation. No postage necessary if mailed in the U.S.A.
FOLD ON TWO LINES, STAPLE AND MAIL.

GH20-1260-0

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

First Class
Permit No. 439
Palo Alto
CA. 94301

Business Reply Mail
No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department J11
1501 California Avenue
Palo Alto, California 94304

Fold Fold

IMS/VS GIM Printed in U.S.A. GH20-1260-0

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

GH20-1260-0

IMS/VS GIM Printed in U.S.A. GH20-1260-0

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)