SILVABASE: A FLEXIBLE DATA FILE MANAGEMENT SYSTEM

Steven J. Lambing
NASA Marshall Space Flight Center
Huntsville, AL 35812

Sandra J. Reynolds
Boeing Computer Support Services
Huntsville, AL 35812

ABSTRACT

The need for a more flexible and efficient data file management system for mission planning in the Mission Operations Laboratory (EO) at the Marshall Space Flight Center (MSFC) has spawned the development of Silvabase. Silvabase is a new data file structure based on a B+ tree data structure. This data organization allows for efficient forward and backward sequential reads, random searches and appends to existing data. It also provides random insertions and deletions with reasonable efficiency, utilization of storage space well but not at the expense of speed, and performance of these functions on a large volume of data. Mission planners required that some data be keyed and manipulated in ways not found in a commercial product. Mission planning software is currently being converted to use Silvabase in the Spacelab and Space Station Mission Planning Systems. Silvabase runs on a Digital Equipment Corporation's popular VAX/VMS computers in VAX FORTRAN. Silvabase has unique features involving time histories and intervals such as in operations research. Because of its flexibility and unique capabilities, Silvabase could be used in almost any government or commercial application that requires efficient reads, searches, and appends in medium to large amounts of almost any kinds of data.

INTRODUCTION

Context

In the Mission Operations Laboratory at the Marshall Space Flight Center, payload operations for Spacelab and non-Spacelab shuttle flights are planned and conducted. Mission planning includes projecting and designing the Space Shuttle Orbiter's trajectory and attitudes, determining from that the periods of opportunities for execution of payload experiments, scheduling those payload operations, and scheduling and managing the two way flow of scientific data between the orbiter and the ground. This endeavor calls for a variety of forms and amounts of data to be stored in secondary computer storage (disk files). Mission Planners require that their software be able to efficiently read this data sequentially forward and backward, efficiently search for a random key, and append new records to the existing sets of records. Additionally, they require the ability to make random insertions and deletions, and utilize storage space well without adversely impacting access speed.

The forms that all of this mission planning data takes are not homogeneous. Some of the data is simply a collection of related data items of different data types. Other data takes a tabular form consisting of numerous records, each having the same format (being homogeneous) with one data field serving as the key and usually representing time. There are numerous other forms. Additionally, mission planners have one unique type of data and methods of manipulating it. This data consists of a key which is made up of two values. The two values represent a starting time and an ending time of a time interval. Associated with this interval key, or these "On/Off" times, will be zero or more other data values, depending on what event the interval represents. A set of records, keyed by these intervals, are used to represent an intermittent recurring event or set of conditions. This type of data representation is frequently used in mission planning.

History

During the latter 1970s and the 1980s, mission planners stored much of their data in a set of file formats known as MIPS files. MIPS stands for the Marshall Interactive Planning System. This term is used to refer to all of, or several different components of, the Mission Operations Laboratory's mission planning computers and software. For the purposes of this paper however, MIPS only refers to these file formats and associated file access software used by

mission planners before the advent of Silvabase. MIPS development began around 1974 on a Sperry 1100 computer. When Digital Equipment Corporation's VAX computers were obtained in 1979, MIPS was migrated over to that platform. In 1986, mission planners began to question the future of MIPS. It had become problematic because it was not originally designed for the VAX nor was it implemented in such a way as to be maintainable. Software analysts determined that minor fixes to MIPS problems were not cost effective due to decreasing confidence in it and major modifications would be equivalent to a rewrite [1]. MIPS had served the mission planners well, but it was time to move on.

It was determined to replace the MIPS file system with a new data file system that was designed to work in the VAX/VMS environment and address all of the mission planning requirements. An obvious option, the use of a commercial database system, was rejected. A commercial database could not be used because there were indications that data retrieval and record insertion would be too slow. Also, the capability to correctly handle interval-keyed data and provide the special functions for interval-keyed data required by mission planners was not found in any commercial product. Finally, and possibly most importantly, it must be possible to freely distribute mission planning software. Other organizations outside of MSFC's Mission Operations Laboratory have been and will in the future be required to use the mission planning software. If a commercial product of any kind was required to be purchased by such users, distribution of the mission planning software would be greatly hindered.

Since 1986, the development of Silvabase and the conversion of mission planning software from MIPS has been ongoing. Boeing Computer Support Services (BCSS), a programming support contractor at MSFC, has performed the task of designing, coding, and documenting Silvabase according to requirements established by the Mission Operations Laboratory. Silvabase owes its existence to the bright team of programmers at BCSS.

A USER'S PERSPECTIVE

What is it?

Silvabase consists of a flexible file format, a library of subroutines for reading and writing Silvabase files, a utility program for interactively manipulating Silvabase file contents, and substantial documentation. Silvabase is implemented on Digital Equipment Corporation's VAX 11 family of computers using the VMS operating system in VAX-11 FORTRAN. Currently, Silvabase is only being utilized from FORTRAN programs, but it should be useable from programs in other languages that adhere to the VAX subroutine calling standards. The name "Silvabase" is created from the Latin word "Silva" meaning forest. As is described below, Silvabase files are based largely on the B+ tree data structure, and thus the concept of a forest is brought to mind.

Silvabase is a data file management system that is particularly suitable for storage of small to medium-large amounts (up to 2.1 gigabytes) of data in which the principal data is organized into collections of homogeneous records using a single key. The key is the first field in a data record with a unique value and is used to locate that record. Silvabase provides for the key to be one of nine different data types, from integer to a string of eight characters. Other data items in Silvabase may be one of twenty two different data types. Multiple collections of records, called "subjects" may exist on one Silvabase file at the same time. Subjects may contain data items outside of and associated with the data records, which are normally viewed as constant with respect to the records. Files may contain data items outside of all the subjects which are viewed as constant across all subjects. Exactly what file variables exist on a given file, what subjects are included in a given file, what subject variables appear on each subject, and the structure of the data records on each subject is left entirely up to the applications programs which write the file and subjects. Although most of the capabilities of Silvabase were created with mission planning requirements in mind, this flexibility and tiered structure of the data makes Silvabase potentially applicable to a very wide range of data storage needs.

Because of the requirement that mission planning software be easily distributed, Silvabase was created with portability between VAX/VMS systems in mind. There is nothing in Silvabase that hinders its implementation on another VAX running the VMS operating system. Silvabase uses standard VAX file names. The files may be accessed on a remote node via network connections. And Silvabase does not add to or restrict the standard VAX file protection, expiration or automatic deletion functions.

A User's Concept of a Silvabase File

Figure 1 shows a conceptual diagram of a Silvabase file illustrating the components as seen from the viewpoint of a file user. These are described below.

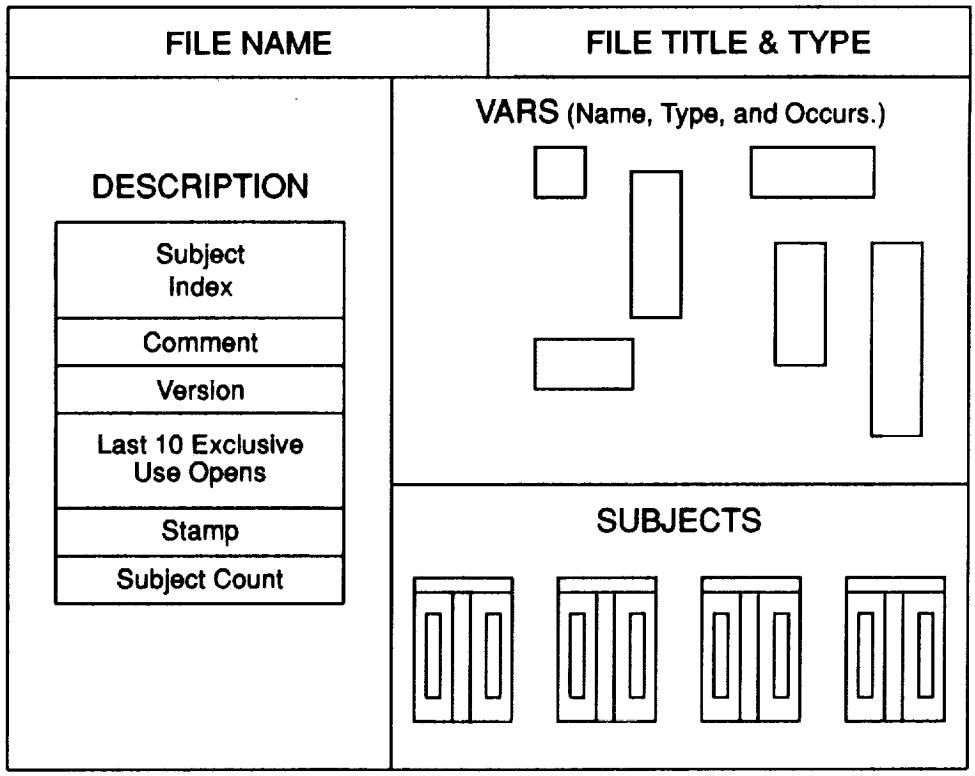

Figure 1: **Components of a Silvabase File**

- FILE NAME - This is the VAX/VMS name of the file. It is not actually contained in the Silvabase file.
- FILE TITLE - This is an optional 72 character title that may be written to the file.
- FILE TYPE - This is a four-character identifier which would be set to a predefined value and is intended to indicate that the file has certain standard contents to any user who may read the file.
- SUBJECT INDEX - This is an ordered list of the names of all of the subjects that are contained in the file. This list may be reordered by the user to suit his requirements.
- COMMENT - The file comment is a 216-character long string that may contain whatever description the user chooses to write.
- VERSION - This field is set by the Silvabase software and indicates the version of Silvabase software used to create the file.
- LAST 10 EXCLUSIVE USE OPENS - Silvabase keeps track of the times that a file was opened for "exclusive use," i.e. with intent to write. Silvabase maintains this list which contains the dates and times of the last 10 exclusive use opens.
- STAMP - The stamp is the so called "certification stamp." This field may be used by a user in authority to indicate that the file and data it contains is official or verified. This field contains the name of the certifying authority and the date and time of the certification. If the file is opened with the intent to write, it is automatically decertified.

- SUBJECT COUNT - This is simply the number of subjects on the file and equals, naturally, the number of subjects in the subject index.
- VARIABLES - The file variables, as mentioned above, are simply any number and type of data items which the user creates and assigns values to. These are intended to contain data which is descriptive of the entire file. The file variables may be of any of the twenty-two data types available for variables in Silvabase. Also, these variables may have multiple occurrences, i.e. they may be dimensioned as one dimensional arrays.
- SUBJECTS - Silvabase subjects are the principal structure for containing data in a Silvabase file. Each subject is independent of the others. The structure of a subject is definable by the user. The features of a Silvabase subject are described below.

A User's Concept of a Silvabase Subject

Figure 2 shows a conceptual diagram of a Silvabase subject illustrating the components as seen from the viewpoint of a Silvabase file user. Each component of a subject is described below.

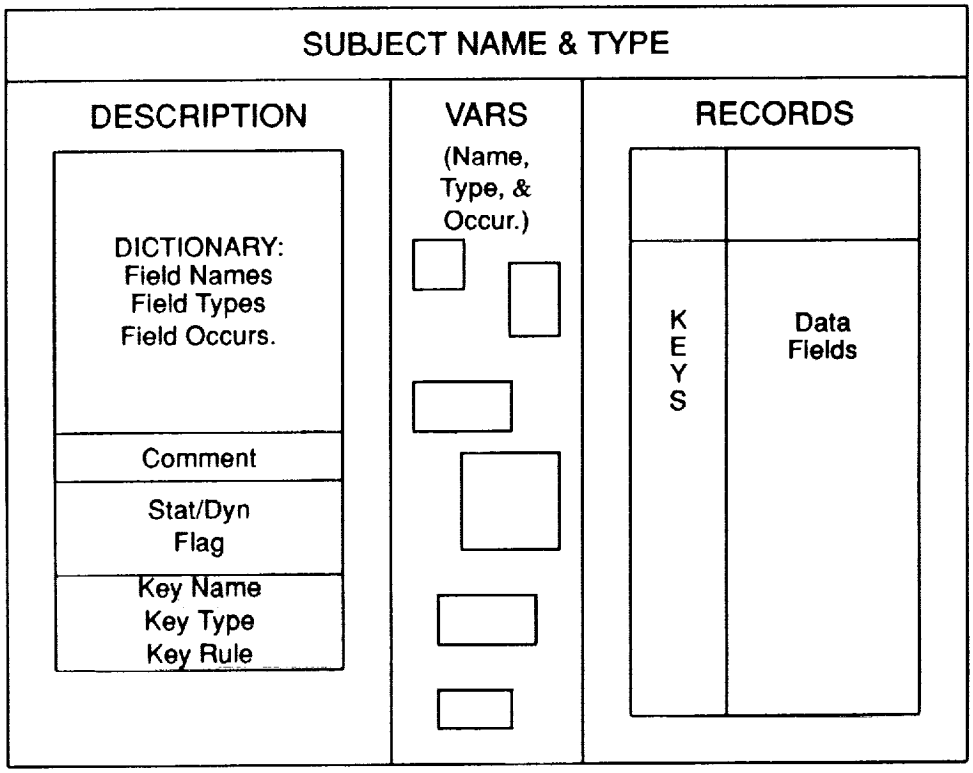

Figure 2: **Components of a Silvabase Subject**

- SUBJECT NAME - Every Silvabase subject has a unique name up to sixteen characters.
- SUBJECT TYPE - This is a four-character identifier which would be set to a predefined value and is intended to indicate that this subject has certain standard contents to any user who may read the subject. It is to the subject what the File Type is to the file.
- DICTIONARY - This is a listing of the names, data types, and number of occurrences (dimension) of all the data fields stored on the subject records. The definition for the key is stored separately (see below) and may be retrieved separately.
- COMMENT - The subject comment is a 72-character long string that may contain whatever description the user chooses to write to describe a subject.

- STATIC/DYNAMIC FLAG - This flag indicates whether this subject is considered "static" or "dynamic." A "static" flag on a subject will prevent that subject from having records inserted or deleted. They may be appended or edited. The flag is intended as a safety feature and may be changed at will.

- KEY NAME, TYPE AND RULE - This information describes the key field of the data records. The key rule is currently only applicable to interval keys and describes how those intervals may relate to each other. Key Rule 1 does not allow the intervals to share a common endpoint. Key Rule 2 allows this. All interval keyed subjects in mission planning are currently defined to use Key Rule 2.

- VARIABLES - The subject variables are simply any number and type of data items which the user creates and assigns values to. These are intended to contain data which is descriptive of this entire subject. These variables may be of any of the twenty two data types available for variables in Silvabase. Also, these variables may have multiple occurrences, i.e. they may be dimensioned as one dimensional arrays.

- RECORDS - The records in a subject are designed to be the principal data structure in a Silvabase file. The records may be conceptually described as row after row of data items, each row having the same structure as the others. Individual records in a Silvabase subject may be read, written or deleted sequentially or randomly. Of the data items in a record, the first one is considered the key field and the others are referred to as data fields. On each record, the key must have a unique value that differentiates that record from the others. The records are stored in ascending order of the key and this order is maintained by Silvabase. The data fields on each record then contain values that logically correspond to the key. For example, in mission planning, one type of Silvabase subject contains a representation of time in the key and the data fields contain information describing the position and velocity of the Space Shuttle in orbit. This type of subject then comprises a time history of the Shuttle's location where each record represents another moment in time.

The key of a Silvabase subject may be one of nine allowed data types, and each data field may be one of the twenty two data types Silvabase allows. How a subject is built, i.e. what type of key is used, what types and how many data fields are used on each record, and what each data item represents, is what makes a given Silvabase subject a unique subject "type." The example described above is a trajectory type of Silvabase subject.

As can be seen from this description of a subject, there are myriad ways of defining a Silvabase subject by selecting a type of key and a number and type of subject variables and data fields. In mission planning, dozens of different standard subject types have been defined, each for storing a different kind of data set such as a trajectory, vehicle attitudes, and a payload's use of resources.

The Tools of a Silvabase User

Naturally, to create and manipulate a Silvabase file, the user will need some tools and accessories. The most crucial tool is of course the Silvabase Library, a library of VAX FORTRAN subroutines which constitute all the primary functions one would need to perform on a Silvabase file, from creation to field by field editing. This library, along with detailed user information, is documented in The MASE/Silvabase Programmer's Guide [2]. (MASE is the Marshall Applications Support Environment, a BCSS developed set of programming tools of which Silvabase is one.)

Another important tool for Silvabase file users will be "SUP", the Silvabase Utilities Program. This is a program still under development which will give file users access to file manipulation functions in an interactive setting. SUP will allow users to interactively create files and subjects, enter and edit data, plot and tabulate data, and perform scripted file operations. Currently, development of SUP is in the requirements definition stage, but a predecessor to SUP called ISUP (Interim SUP) has already been developed. ISUP has many, but not all, of the functions planned for SUP. Due to the great flexibility of Silvabase and the special needs of mission planning, both ISUP and SUP will address mission planning requirements and will not be capable of performing every Silvabase function on every conceivable type of Silvabase subject. Like ISUP, SUP will also include a documented library of callable functions which are features of the program above and built from the primary Silvabase functions [3]. SUP will also be documented with a user's guide as is ISUP [4].

As mentioned before, mission planning has defined a couple of dozen standard Silvabase subject types specific to mission planning. These definitions have given rise to three other accessories that make Silvabase programming easier in the mission planning world. First of all, the Silvabase subject types for mission planning are described in a document that details the subject user type, subject variables, key type, data fields and other aspects of each subject

type [5]. Along with that, a Silvabase file has been established which contains one empty example of each standard subject type. These subjects can be used by Silvabase as templates any time a user needs to create another subject of a standard type. Finally, a text library containing fragments of FORTRAN code was created. This library is referred to as the structures library. Each entry in the library contains the FORTRAN declarations necessary to create a data structure suitable for containing the record from one of the standard subject types. Programmers working on applications that must read or write standard mission planning Silvabase subjects can use these code fragments in their programs to quickly and easily create correct data structures for holding records of standard subjects. This simplifies the coding job and insulates application programs from potential changes in standard subject definitions.

One final Silvabase accessory that is not as important to users of Silvabase files is the MASE/Silvabase Internals Document. This document was written by BCSS developers of Silvabase for themselves. It details the internal workings of Silvabase files and how the library software operates on them.

Special Functions

The Silvabase software includes a number of special functions that are required to do the types of data manipulation needed in mission planning. The following are some of those functions that have been built into Silvabase.

- Key Offset — This function will offset each key in a specified subject by a specified amount. It will not affect the position of any key within the subject.
- Data Field Offset — This function will offset a numeric data field by a specified amount.
- Complement — This function stores time intervals not included in a specified subject on an output subject. The intervals to be complemented are within a specified domain of values.
- Union — This function will union the intervals of two or more interval keyed subjects. The results are stored on a new subject whose intervals contain the times included in any of the input subjects' intervals (logical 'OR').
- Intersect — This function will intersect intervals of two or more subjects. The results are stored on a new subject whose intervals contain the times included in all of the inputs subjects' intervals (logical 'AND').
- Intersect and Transfer Data Fields — This function will perform the same operation as the previous intersect function with the additional feature that specified data fields can be transferred to the new subject.
- Interpolate — This function uses the central difference formula to interpolate two numeric data fields on a key value.

INTERNAL IMPLEMENTATION

B+tree

Silvabase is designed to allow sequential as well as random access to records. Sequential access is very desirable because it eliminates the need for disk seeks when reading or writing sequential records. Random access is needed for searches, insertions, and deletions. In order to accomplish these operations efficiently, Silvabase uses a B+ tree for each subject on the file. The B+ tree is a B-tree combined with a sequential set of data records called a sequence set.

The sequence set is extremely efficient for sequential access and appending new records. The set is made up of physically contiguous records organized in sequential order by key. The records are organized into blocks. The blocks are linked together sequentially according to the range of key values contained within each. Each block points both to the next and previous block in the set. This blocking of records allows for easy maintainence of the sequence set because it limits the effects of insertions and deletions to the records within or near the block containing the change. Using sequence blocks allows several sequential records to be read into memory at once rather than reading them one at a time. The file is read sequentially by loading a block of records into memory and reading each record until the last record in the block is reached. The next block is then located using a pointer and loaded into memory for further reading. This greatly lessens the amount of disk seeks needed for transferring records to or from disk. Sequential search performance is also improved since the records can be searched in memory instead of on disk.

Records may be appended, inserted, or deleted in the sequence set. Appending to the end of the sequence set is very straightforward and can be done with tremendous effectiveness. Insertions into a sequence block are made by first performing a binary search for the correct position within the block for the new record. The records are then shifted to make room for the new record and it is inserted. If overflow occurs, the block is split into two blocks and the new block is linked into the list. Deletions are made using simple collapse procedures and concatenating when a block underflows. These operations keep the sequence set in order and eliminate the need for sorting.

A B-tree is used as an index to organize the sequence blocks. It provides fast, efficient random access to records and is completely maintainable. The nodes of the B-tree contain separators. These separators are not the actual keys of the records but indicate the range of keys located within a particular sequence block. Several separators along with child pointers are stored in one node in the form of an ordered dense list. This allows several separators to be read into memory at once. This also simplifies making changes to the node, allows binary searching, and eliminates disk accesses. The same advantages apply here as apply to blocking records for the sequence set. The B-tree is constructed from the leaves up, which keeps the tree well-balanced for efficient, fast searching. Locating a specific record involves descending through the B-tree, loading a node into memory, and performing a binary search on the separators within the node to find the path to the sequence block containing the record. A stack is used to keep track of the path through the B-tree. Once the correct sequence block is located, a binary search is performed on the keys within the block. If the record is very small, the data is stored within that block. Otherwise, the record will have a pointer to the location of the desired data. Even though all searches must descend through the entire tree to the sequence blocks, the performance is so good for the worst case search that this is not a concern.

Chunks

The size of sequence blocks and B-tree nodes is the same. They each fill one physical record of 1024 bytes. Silvabase uses block I/O, since records on the VAX are written to and read from the disk 512 bytes at a time, the physical records must be a multiple of 512 bytes in size. Small records increase I/O and large records require more space. This size of 1024 bytes is large enough to decrease I/O but small enough for good space utilization.

A physical record for a Silvabase file is called a "chunk". When a Silvabase file is created, several chunks are allocated, some for specific potential purposes. Each chunk has a chunk header. This is used used to save information about the chunk and to link it both forward and backward with the other chunks in the file. A chunk is referenced by its sequence number in the file. The first chunk is always the file header.

Structures

The internal organization of a Silvabase file involves the use of several types of internal structures. Each structure is stored in a separate chunk. Some structures require several chunks and are linked together in a "chunk chain". The internal structures of a Silvabase file include the following:

- FILE HEADER — This keeps information about the file such as its title and type. It also keeps track of the chunk numbers of the other internal structures in the file.
- FILE VARIABLES — As defined previously, these are used to store any type of information that pertains to all the subjects on the file.
- SUBJECT VARIABLES — As defined previously, these are used within each subject to store any type of information that is constant throughout the subject.
- SUBJECT INDEX — As described earlier, this contains a list of all the subjects on the file in order. It stores the name, type, and subject description pointer for each subject. A subject may be found using a sequential search for the subject name or by its sequence number in the list.
- SUBJECT DESCRIPTORS — There is one subject description for every subject on the file. A subject description serves as the header record for a subject in the same way that the file header does for the file. The descriptions are kept separate from the subject index for faster searching of the index.
- SUBJECT COMMENTS — Each subject may have a comment associated with it. As defined above, this comment is pointed to by the subject description.

- DICTIONARIES — This stores the format of the record structures in the file. It contains each field name, type, and occurrence.
- B+TREE INDEX SET — This stores the nodes of the B-tree.
- B+TREE SEQUENCE SET — This stores the sequence blocks for each subject. There are 5 types of sequence blocks depending on the size of the key and data to be stored. In many cases, a pointer to the data is saved with the key instead of the data itself.
- DATA BLOCKS — This stores the actual data for the record fields for each subject.
- EMPTY CHUNKS CHAIN — This is a set of empty chunks linked together and available for use by Silvabase.

When a Silvabase file is opened, parts of the file header are read into a File Information Table (FIT) which is located in memory and kept for fast access to the structures within the file. The Subject Information Table (SIT) serves the same purpose for an open subject on the file.

CAPABILITIES

Performance

Silvabase traverses through the B-tree using a binary search on each node to locate keys. This technique eliminates half of the remaining keys in the subject with every comparison of keys. The length of a worst case binary search is calculated using the following formula:

$$W(M,N) = 1 + \frac{ln(N+1)/2}{ln(M/2)}$$

M is the B-tree order - maximum number of children possible for each node
N is the number of keys in the tree
This formula assumes that each node is only half full.

Both the order of the B-tree and the number of records within the subject effect the amount of disk accesses needed for the tree traversal. The rate of additional disk accesses becomes smaller as more records are added. A large B-tree order also reduces disk accesses but must be limited for maximum gains. These effects are shown in Figure 3 and Figure 4.

Capacities

Most of the limitations on Silvabase were chosen in order to accomadate the needs of Mission Planning. Some restrictions of a Silvabase file are the following:

- File Size — This limit exists because Silvabase allows 21 bits for physical record representation. Since the physical record length is 1024 bytes, the maximum number of chunks must be 2^{21} =2,097,151 or 2.1 gigabytes per file.
- File and Subject Variables — There is no set limit on the number of file or subject variables allowed.
- Subjects — The maximum number of subjects that may be on the file is 5000.
- Records — There is no restriction on the number of records per subject, but the number of data fields per record cannot exceed 1000. The maximum size of a record is 8000 bytes.
- FIT and SIT — As many as 12 files and 48 subjects may be open at one time.

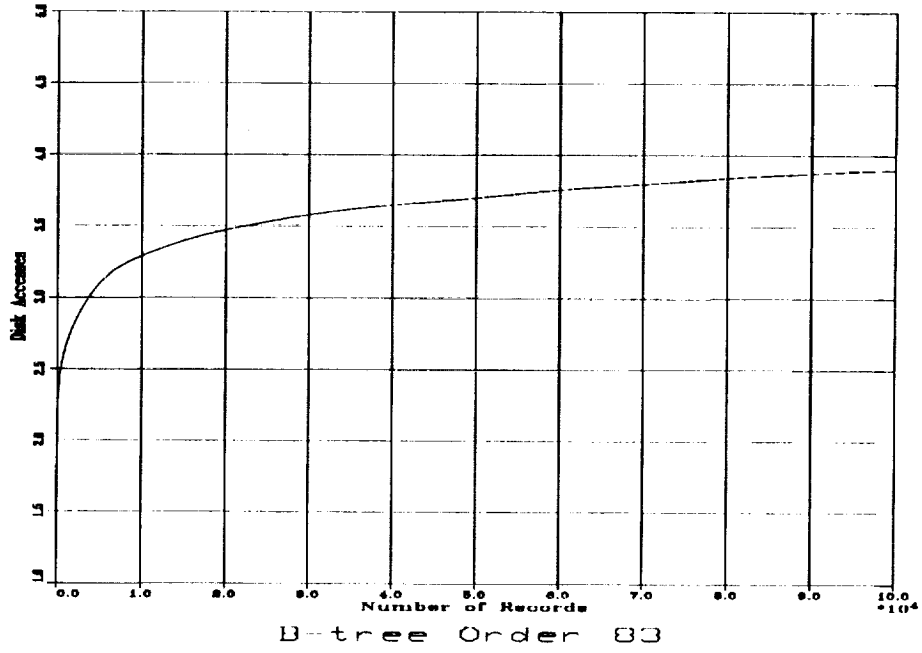

Figure 3: **At Constant B-Tree Order 83**

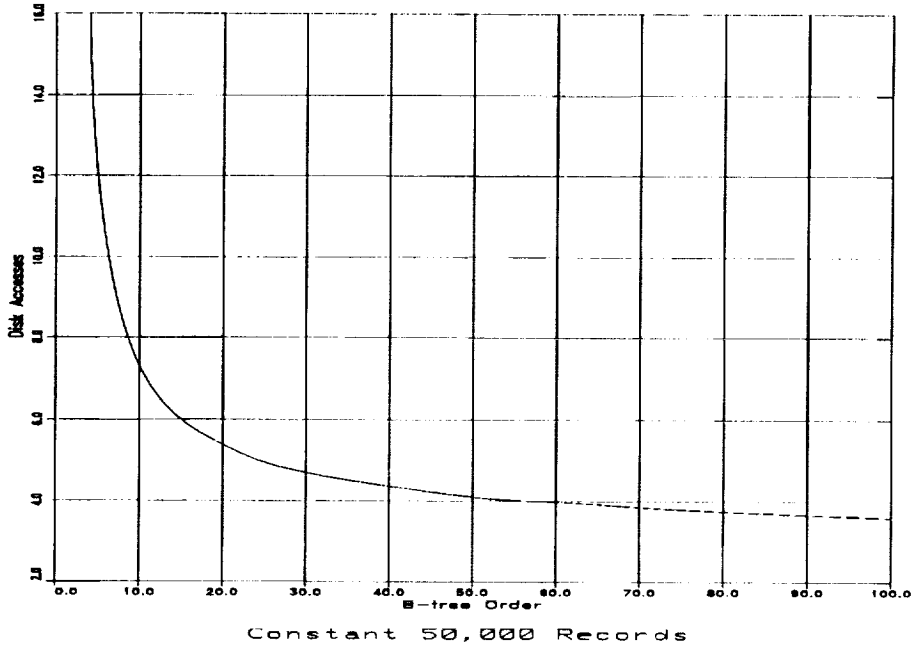

Figure 4: **At Constant 50,000 Records**

SUMMARY

Silvabase was created out of a need in the Mission Operations Laboratory for a powerful, maintainable data file management system that both addressed specific mission planning needs and was flexible enough to adapt to the ever changing requirements in mission planning. Silvabase offers the data-generating user a tiered structure of records within subjects within files, numerous data types, and the flexibility to organize these in whatever way best suits his or her needs. It is based on the powerful B+ tree data structure that gives file accessing its efficiency. This combination of features makes Silvabase an attractive data file management system for innumerable applications.

REFERENCES

[1] S. M. Spillers. *The Future of VAX MIPS*. Boeing Computer Support Services, October 17,1986.

[2] *MASE/Silvabase Programmer's Guide*.Boeing Computer Support Services, March 29,1991.

[3] *ISUP Interim Silvabase Utilities Program Library User's Guide*. Boeing Computer Support Services, July 23,1990.

[4] *ISUP Interim Silvabase Utilities Program User's Guide*. Boeing Computer Support Services, July 2,1990.

[5] S. J. Lambing. *Formats for Mission Planning Subject Types in Mase Silvabase Files*. National Aeronautics and Space Administration, MSFC, August 9, 1990.