A Case Study of File System Workload in a Large-Scale Distributed Environment

Deepinder S. Gill   *Songnian Zhou*   *Harjinder S. Sandhu*

Computer Systems Research Institute

University of Toronto, Toronto, ON, M5S 1A4

{gill, zhou, hsandhu}@sys.toronto.edu

Please note: A smaller version of this paper has appeared in ACM Sigmetrics, May 1994, as an extended abstract.

Abstract

As large scale distributed file systems come into wide spread use supporting industrial organizations, the performance and scalability of file systems becomes important issues. Key factors such as file server bottlenecks and network congestion are heavily dependent on file system workload characteristics, yet no workload study on such systems has been reported to date.

In this paper, we present a case study of a large industrial distributed file system supporting several thousand users. Our study is at the workgroup level, rather than at individual file access level. Our study reveals substantial departures from observations made in previous academic and centralized commercial environments. The average file size is 25-30 Kbytes, much larger than those from previous studies. Over weekly periods, we found that more than one-third of the files are active. Of the active files, read activities dominate writes 9:1. We found a very high level of file sharing: 50-60% of the files are shared among two or more workgroups. On the other hand, a single workgroup remains as the only writer for 97% of the write active files, suggesting that dynamic file replication at the workgroup level may effectively improve file access performance. It is also noted that shared files tend to be larger, and that write activities decrease as file sharing increases. We believe that these observations are representative of an important class of industrial computing environments, and are valuable input to future file system design.

1 Introduction

Distributed systems are rapidly becoming the most prevalent form of computing today. As large-scale distributed systems become more common, distributed file systems are faced with the challenge of providing efficient service in systems with hundreds, or even thousands, of hosts. The ability of the file system to deal with the increased traffic and longer latencies inherent to large-scale environments has a significant impact on the performance of the distributed system. The performance of techniques for managing resources in large-scale distributed systems are largely dependent on the workload of the system.

In the past, workload studies of file systems have been a basis for evaluating file system design strategies [2][7][8][10][11]. These studies have enabled designers to quantify those file system characteristics that have the greatest impact on performance. However, while contemporary file systems have been strongly influenced by the measurements reported in past workload studies, most past workload studies have been conducted in small scale academic distributed environments or in centralized commercial environments [1][2][3][5][6][9][13]. This has led to a significant gap in our understanding of the behaviour of large-scale distributed environments, and, in particular, large-scale distributed industrial environments where usage patterns may differ significantly from those of academic environments.

In this paper, we present the results of a workload study conducted in a large-scale distributed industrial environment. This environment consists of several thousand workstations/users organized into project-oriented workgroups and connected together by a hierarchy of local area and wide area networks and spread across several metropolitan areas. The primary activity of the users in this environment is software development. This case study examines the file system activity of users involved in the development of one large software product. In this subsystem, there are about 7500 users organized into over hundred workgroups, with each workgroup comprising 20-25 workstations and a file server connected by a local area network.

The type of clustering of workstations (into workgroups) found in this environment is typical of large-scale distributed environments. It has been noted by Satyanarayanan [12] that clustering is natural to such systems because it overcomes the length and bandwidth limitations of local area networks, and also simplifies administrative functions and fault isolation. For the purpose of this study, this clustering enables us to distinguish between file system activities that have only local (that is, intra-cluster) impact, and those that may have global (inter-cluster) impact. The scalability and performance of the system is determined almost entirely by the inter-cluster file system workload. In the workload study presented here, we focus exclusively on inter-cluster, or inter-workgroup, file system activity.

Due to the nature of the environment, we are limited in this study to using high level measurement tools that do not modify the operating/file system and do not interfere in anyway with the normal use of the system. The result is that, unlike previous workload studies which have characterized file system activity at the level of individual file references (that is, open, close, read, and write), this study characterizes file accesses at a coarser granularity, at the level of whole files accessed from particular workgroups. While this is not as detailed as a file system trace, the measurements here present an interesting high level case study of one particular industrial environment, and as such, complement the knowledge gained from past workload studies of other types of environments.

The measurements presented in this paper show a number of significant differences from those of past workload studies conducted in academic environments. An examination of the file size distribution shows that files are much larger in size compared to files found in academic environments. The files that have been accessed during the study period have even higher average sizes.

An examination of the file sharing characteristics between workgroups shows that there is a significant amount of sharing of files. Shared files are relatively unstable, being modified on a daily or weekly basis. This is in stark contrast to earlier studies which have found sharing limited to read-only system files. This difference can be attributed to the collective project-oriented nature of industrial environments in general, as opposed to academic environments where users tend to work on individual projects and rarely share data.

In Section 2, we discuss some of the important workload observations of previous studies. Section 3 describes the system environment and the data collection methodology used. The static and dynamic file system characteristics are presented in Section 4. In Section 5, file sharing characteristics among workgroups are analyzed.

2 Past Workload Studies

In analyzing the workload of a particular environment, it is important to understand the similarities and differences of the observations to those of other studies both of similar and different types of environments. Since no other studies of large-scale distributed environments currently exist, we can only compare the observations made in this paper to those of other types of environments. These studies can be classified as studies of academic environments and studies of commercial centralized systems. In this section, we briefly review the primary results of some of these studies.

Academic or research environments have been studied by Ousterhout et. al. [6], Baker et. al. [1], Floyd [3], and Blaze [2]. The observations made in each of these studies are mostly similar to one another. Ousterhout et. al. analyzed a trace of user level activity in a timeshared VAX-11/780 environment at U.C. Berkeley. This environment supports about 331 users during a 3 day trace period. User activity was low, as on the average in a ten-minute interval, only 44 users were actively accessing files and data transfer rates of only a few hundred bytes per second per active user were observed. Files were small in size, with 80% of accessed files below 10 Kbytes and 50% below 2500 bytes. Large files were found to receive a higher proportion of the accesses compared to smaller files. The sharing of files between users was not analyzed. This study was later repeated in a distributed environment (about 40 workstations and three file servers) by Baker et. al. Observations of the workload in this environment were found to be generally consistent with the original study. Files were still small in size, although some multi megabyte files were found to exist. Concurrent sharing was analyzed in this later study and found to be very low, with only 0.34% of file opens resulting in simultaneous sharing. Sequential sharing was not analyzed.

Floyd conducted a study at the University of Rochester in an environment similar to that of the original Berkeley study, and made similar observations. However, files were found to be even smaller; 50% of the files were below 1 Kbyte. Reads as opposed to writes were observed to be the dominant file access mode accounting for 84% of all file accesses. Analyzing file sharing, Floyd reported that only 12% of the total file accesses were to shared files. Most of these files were system binary files and only a few were user files. Another study by Blaze, conducted at DEC-SRC (a research environment) and Princeton University also reports similar observations. Reads were found to be 3-5 times higher compared to writes. Sharing is observed to be higher than previously reported by Floyd in an academic environment, as almost 60% of the read accesses were to files that had been read recently by 2-20 users on different workstations. Almost negligible read-write sharing was reported.

Studies of commercial environments have been analyzed by Ramakrishnan et. al. [9]. They study eight different commercial VAX/VMS environments in which the number of registered users (note that the Berkely study reported active users) in all the sites were in the range of 10 to 550. The sites covered different environments, including scientific research (200 users), program development (10 users), office applications (100 users), scientific applications (150 users), transaction processing (500 and 550 users), batch processing (400 batch jobs), and publishing (320 users). In all the environments, a wide range of mean file sizes between 19 to 277 Kbytes was observed. The file size distribution was found to be skewed since the median file size was in the small range of 1.0-3.5 Kbytes. It was also observed that the transaction processing and batch environments had fewer files but files in these environments were 5-10 times larger than files in the interactive environments.

Only a small fraction of 5-10% of the total files were found to be active (read or write accessed by users). Reads were 3-4.5 times more than writes. Sequential sharing was observed to be the common mode of sharing; 23-37% of all the files accessed were sequentially shared between two or more processes while only 6-11% of the files were simultaneously shared.

About 30- 40% of all the active files were read-only shared. Read-write shared files accounted for 30-55% of the active files. Shared files were accessed more often than ordinary files; 77-90% of the file opens, 50-70% of the reads and 50-70% of the writes were to shared files.

Sharing of files in commercial environments has also been analyzed by Kure using traces from three IBM centralized installations [4]. Kure observed that 23-47% of the active files were shared. He observes that shared files were accessed more than ordinary files, were larger in size, were more stable (lasted longer), and received fewer updates. However, most of these shared files are read-only and shared typically by only two users.

From the workload studies discussed above there are some common workload characteristics observed among academic and commercial systems. For instance, read accesses are higher than write file accesses, and files are typically sequentially accessed. File access traffic mainly comprises whole file transfers. Active files tend to be large in size and receive higher numbers of accesses. The two types of environments, however, differ in the workload characteristics of file size and sharing. In academic environments, files are generally found to be small. Sharing is low and generally restricted to system files. Conversely, commercial studies report larger files and higher degrees of sharing. In addition to system files, a higher fraction of user files are shared. Widely shared files are observed to be more stable (longer lifetime) and account for a high proportion of file accesses.

3 System Description

3.1 Architecture

The system under study is part of a large industrial company that develops specialized computer software. Like many other large organizations that once relied solely on mainframes for their computation and data storage, there is a gradual shift towards distributed computing and distributed data management. As a result, the current system is a hybrid one in which both centralized mainframes and network-based workstations are used. The mainframes are predominantly used as data storage repositories, although there are a few users that work interactively on the mainframes.

The system is organized hierarchically in both the physical and logical sense. Physically, a hierarchy of networks, including local area networks (LANs), metropolitan area networks (MANs), and wide area networks (WANs), impose a hierarchical structure on the system that is classified into workgroups at the lowest level, communities at the next level, and divisions at the highest level. This system architecture is shown in Figure 1.

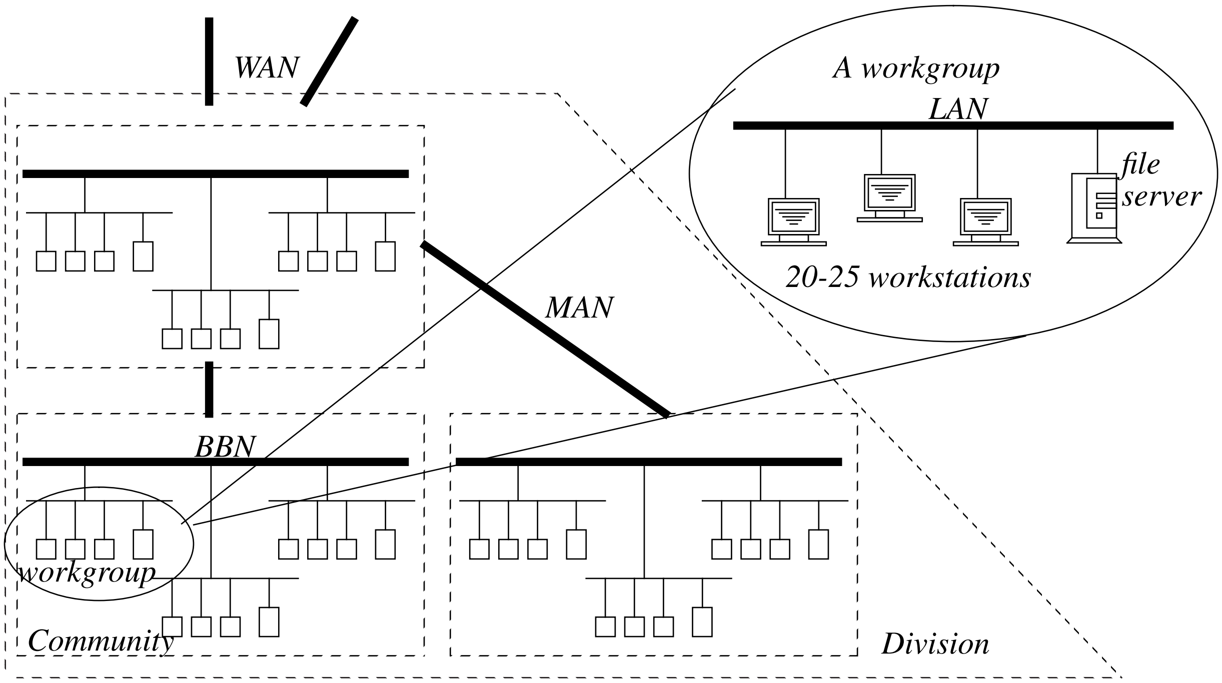

Figure 1: System architecture

A workgroup typically consists of 20 to 25 workstations and one or more file servers connected by a LAN. Workgroups are comprised of users that work closely together on the same project and share information and data extensively. Within the organization there are hundreds of workgroups. Workgroups sharing a common area of interest and falling within the same geographical location (metropolitan area) are grouped together as a community. There are typically between 10 and 30 workgroups in each community, connected together by a combination of LANs and MANs. Communities within the same metropolitan area are in turn connected together by a set of MANs to form a division, with different divisions connected together by WANs.

The file system structure mimics this same hierarchical structure. Data servers exist at each level which serve components of the system below them in the hierarchy. Within a workgroup, there are file servers which are intended primarily to serve accesses within that workgroup. Each community has a set of community data servers (both mainframes and workstations which are not within any workgroup) which serve file access requests from within the same community. At the next level are *master* data servers which serve the entire system. In addition, at each level data is cached or replicated to the next level below in order to improve file system performance and to alleviate the load on the interconnection networks.

In this study, file system activities of users working in a software development project are observed. These users are spread across 53 workgroups and 5 metropolitan communities. The community servers maintain a replica of all the files accessed by these users. The *master* data server manages replication among all the community servers and is responsible for maintaining consistency among them. Our study concentrates on the interaction between the workgroups of the users of this project, and ignores the interactions of the file system both below (intra-workgroup) and above the workgroup level (community or division).

3.2 File System

There are three types of project files that are analyzed in this study. These are *source* files, *object* files, and *meta* files. Source files contain program source code, object files are compiled binary files, and meta files are system directory files that contain information about source and object files. Source and object files can be accessed directly by users, while meta files can only be accessed or updated by the system.

The source files are organized into a library that stores and manages these files. Meta data files contain information about the structure of this library. The system supports multi-site development by replicating the library on all the community data servers. The users register themselves with the library and are given unique user ID's. The library acts as a control layer between the file system and the users and controls changes made to source files through ownership rules. It also provides a full development environment to the users with features to edit, compile and link software.

Compiled object files are generated and released globally (system wide) once per week. These object files are compiled on specialized file servers and then downloaded to all of the community data servers. Users can access these files to link and test their software modules, but are not allowed to directly update the global versions of these files.

Other types of files exist within the system but are not captured in this study because they are accessed only from within a workgroup. The operating system running on the workstations and the workgroup file servers, for instance, are part of the *local* system (that is, local to the workgroup) and all files that are part of this local system are thus also private to the workgroup. Additionally, files that exist within a users home directory are private to that user and cannot be accessed directly from outside the workgroup.

For the files that are part of this study, the characteristics of the three file types are very different from each other. Analyzing these files together as a single file type might result in the dominant file type masking interesting characteristics of the other files (e.g., read-only object files over source files). For this reason, the three types of files are characterized separately.

3.2.1 File System Consistency

The file system replicates and caches data at all levels of the system hierarchy. We observe that the mechanisms used for consistency are, however, dictated by the way these files are used, as well as by the system performance constraints. It is, therefore, interesting to study how real systems are trying to cope with large-scale performance problems. The replication/caching and consistency mechanisms are briefly described in this subsection for each type of file.

On a read access to a source file from a workgroup, a copy of the source file is requested from the community server by the workgroup file server and cached locally on its disk. The workgroup server maintains a copy of all the source files that have been accessed by the users of that workgroup. The cache on the workgroup server is kept consistent to within half an hour of the community server copy. Every half hour the workgroup server checks with the community server to fetch new copies of files that have been modified since the last check. If there have been changes, the source files and the meta data files are downloaded. Meta data files are primarily used to keep track of each replica or cached copy of the library and to determine which files have changed.

Modifications to source files are performed locally by a user within a workgroup but do not affect copies of the file in other workgroups until an explicit transaction is sent by the user to the master server informing it of the update. That is, for files that are to be modified, users in a workgroup typically acquire the file for write, update it a number of times over an extended period of time, and then send the modified version to the master server which propagates the changes to all the community servers. During this time, users in other workgroups may have acquired this file for read but not for write.

Thus, for source files, we observe that sharing patterns are quite different at different levels of the system hierarchy. Within a workgroup, multiple users may be using a file for both read and write, and consistency is maintained by the workgroup file server. Sharing between workgroups is either read-only, where all workgroups only read that file, or single-writer-multiple-readers, where one workgroup modifies the file and a number of workgroups read that file. The interaction between these levels in order to maintain consistency is observed to be deliberately long (1/2 hour) in order to avoid placing too much load on the networks and file servers.

Object files may be created locally by users within a workgroup, but the objects created within a workgroup never propagate to higher levels of the system. Instead, the modified source is compiled into object files at the highest level of the hierarchy and propagated down to each community. The workgroup file servers access object files directly from the community servers, and cache these files locally if that file has been accessed three times. New versions of the object files are created on a weekly basis, and there are up to four versions of an object file existing at any particular time.

3.3 Data Collection Methodology

The data logs used in this study are collected from different levels in the system. Since the system is unavailable for the instrumentation required for a detailed file system trace, these logs are extracted from information already being collected by the system administrators to monitor the system. The logs contain high level information about files, for instance which files have been accessed from a particular workgroup, and do not provide information at the level of individual read and write accesses to files. The logs for the four-week collection period are roughly nine Gbytes in size. A brief description of each log and the information extracted from it is described below.

A static snapshot of the source files is taken by listing the attributes of all the files on the master server (similar to a central server) which maintains full replicas of all files in the library. This static snapshot captures some information about the files such as its size and the time of the last update. To capture the write activity to source files, all the updates received by the master data server are logged in a transaction trace. The information included in this trace are: the user ID of the user sending the update, the names of files being updated in the transaction, the time of the transaction, and the workgroup identification number.

Other logs collected for the source files are as follows:

1. a log of all source files accessed from the community server (i.e., all cache misses on the workgroup server).
2. a list of source and meta files downloaded every half hour to keep the cached files and the library structure on the workgroup server consistent with the community server.
3. a list of accessed source files maintained by the workgroup servers. This list is updated by the workgroup server every night based on user file accesses during the day. However, this list does not include the number of accesses from within the workgroup.

The static information about the object files (such as file sizes) is collected once every week by listing the attributes for all the newly generated files. The dynamic access information is collected from the workgroup servers which maintain a history file for every object file accessed from the workgroup. The information logged in the history files is: first access time, number of accesses, and last access time.

4 File System Characterization

The data logs described in the previous section were collected for a total of four one-week study periods. Each one-week study period is analyzed separately and the four study periods are compared to each other. In this section, we present a characterization of the file system based upon this analysis. Information about file size distributions and file usage patterns is described. A study of file sharing between workgroups is presented in the next section.

Files are categorized into *in-active* files and *active* files. A file is considered to be *active* for a specific one-week study period if it has been accessed by a user in that period. Active files are further classified as *read active* or *write active* depending on the type of access. A file is write active if it is updated at least once during the study period; otherwise, it is read active (also called read-only).

Data from the logs is presented as follows:

1. Static characteristics of the system: number of users and static file size distributions.
2. Dynamic file access characteristics: read and write activities and size distribution of active files.

4.1 Static Characteristics

There are 7598 users registered in the library. These users are divided into different classes based upon the access privileges given to them. There are 3467 active class users, those that are allowed to update files. The rest of the users are observers, those that can only read files but are not allowed to modify them. Users who belong to sites in the same metropolitan area as the master server account for approximately 50% of the total user population.

Source files, meta files, and object files are the three constituent file types studied. Table 1 presents the total number of files, their size distributions and the total amount of data for each of them. A cumulative size distribution plot is also shown in Figure 2.

Type of File	**Number of Files**	**File Size Distribution (Kbytes)**			**Total Data (Mbytes)**
		Mean	**50 percentile**	**90 percentile**	
Source	63,052	29.7	12.0	72.0	1918.0
Meta	321	342.2	8.0	364.0	112.0
Object	366,943	24.9	12.0	66.5	9390.0

Table 1: File system characteristics

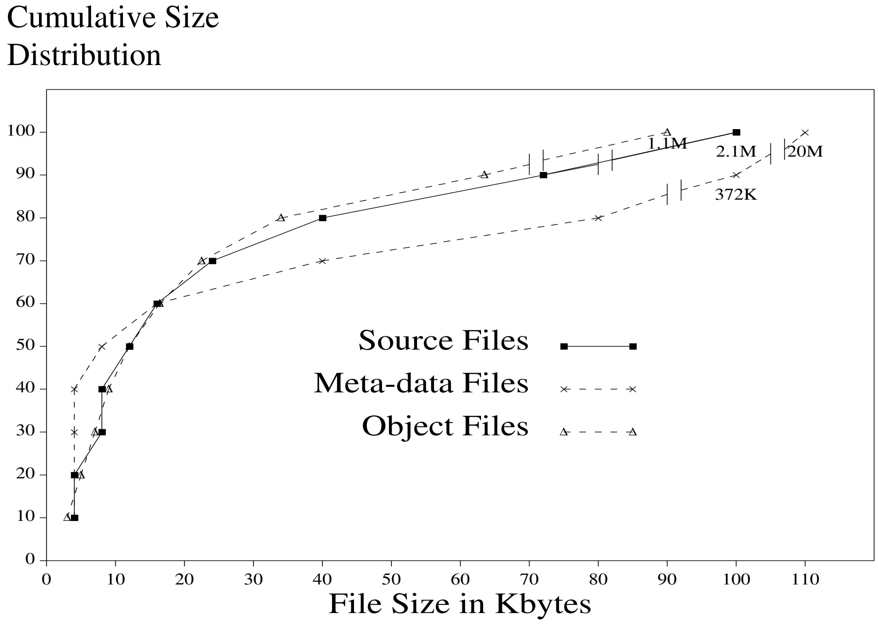

Figure 2: File size distributions

The following observations may be made from Table 1 and Figure 2:

- The median file size for all three types of files is 8-12 Kbyte. In other words, 50% of the files are smaller than 12 Kbytes.
- The mean file size is 2-2.5 times that of the corresponding median size for source and object

files. Both types of files show similar size distribution curves.

- There are only 321 meta-data files. These files have a high mean of 342.2 Kbytes and a low corresponding median of 8 Kbytes. This is attributed to the characteristic of these files: there are a small number of very large meta files that contain the basic structure information of the library.
- The number of object files is higher than the number of source files. This is because four different versions of object files are maintained separately on the object file servers.

On comparing to past workload studies, we observe that files are much larger in this environment than in other systems [1][3][6][13]. For example, Floyd reported an average file size of 1 Kbyte [3]. Baker et. al. analyzing workloads at U.C. Berkeley, observed that 80% of the files were below 10 Kbytes [1].

Studying an industrial workload, Ramakrishnan et. al. find file sizes to be in the range of 8 to 706 Kbytes [9]. However, a median file size between 1.0 to 3.5 Kbytes reported in their study is much lower compared to the median of 8-12 Kbytes found in our study.

Also note that the number of source files for a system of this scale (7598 users and 3467 active users) is low compared to past workload studies conducted in smaller systems. For instance, in Ramakrishnan's study, as many as 706024 files at a site of 500 users are studied. One reason for this large difference is because in this study we only consider project files, whereas Ramakrishnan et. al. also included user home directory and other system files.

4.2 Dynamic Characteristics

The number of active files and their access patterns are examined in this subsection. Non-active files occupy storage but do not affect file access performance of the system. In order to design a file system that performs well, it is important to understand the active usage pattern of the file system.

The meta data files are found to be almost 100% active since the structure of the library is constantly changing. Almost all of the meta files are accessed to update the library structure every half-hour. Therefore, we do not analyze these files further.

Week #	Active Source File (%)	Active Object File (%)
1	39.0	4.6
2	33.5	3.7
3	38.5	3.4
4	38.9	4.2

Table 2: Active files

Week #	Read Active File (%)	Write Active File (%)
1	90.0	10.0
2	87.8	12.2
3	89.9	10.1
4	89.8	10.2

Table 3: Read and write active source files

In Table 2, the percentage of files that are active for all four weeks is shown for both source and object files. Table 3 shows the read and write percentages of the active source files. As observed in Table 2, more than one-third of the source files (34-39%) are active in an one-week period. Analyzing the further breakdown of the active source files, Table 3 shows that 10-12% of the active files are write accessed (and possibly read accessed), while the rest are read-only accessed.

The object files have a low active file ratio between 3.4-4.6%. Initially, it was suspected that most of the user file accesses go to the newly created version and that the past three versions are not accessed frequently. However, on further analysis it is found that 65% of the active files are from older versions.

The other possibility for a low active ratio may be a high locality of accesses; a small number of files receive a disproportionately high number of accesses. This is investigated by conducting a case study for some of the workgroups in the same metropolitan area as the master server. (These workgroups account for 50% of the users.) The user read access distributions for the active object files is shown in Figure 3. A skewed distribution is observed where 50% of the active files are accessed less than 10 times while 90% are accessed less than 80 times. This implies that while most files are being accessed less than 80 times in a week from all workgroups, 10% of the files receive between 80 and 4578 accesses. A possible reason for this can be that users only need a small number of main object files to link and test their own set of software.

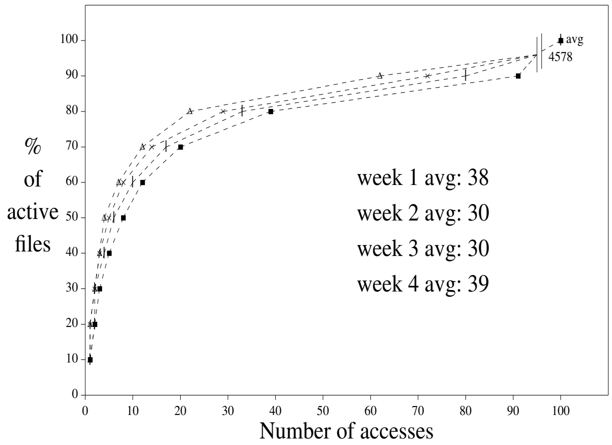

Figure 3: Access distribution of object files

While the average number of accesses to an object files is rather low, the number of accesses is quite high when considering all of the object files together. On the average, 80,000 object files are generated on the master server every week. The new files are then downloaded from the master server to all the sites over slow T1 and T3 backbone networks, in order to create replicas on the community servers. Although we do not report any performance measurements of this file system, it is worth noting that file system activity related to both source and object files creates significant network congestion affecting other file system activities also.

4.2.1 Source File Updates

In this section, write activity to the source files is further analyzed in the following ways: (1) Distribution of updates which shows the percentage breakdown of updates; and (2) 24-hour distribution of updates, presented as maximum, average and minimum number of updates in an hour for all days, in a representative week.

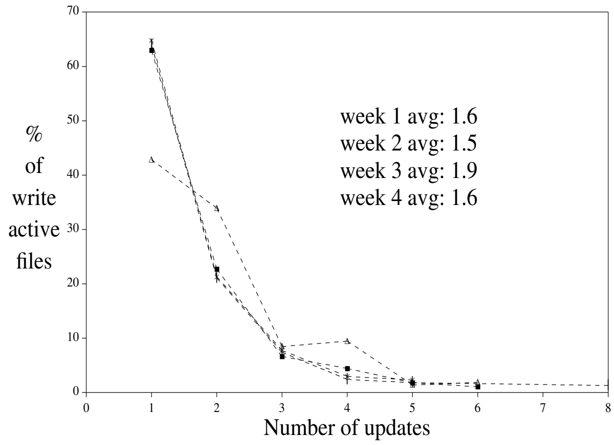

Figure 4: Distribution of updates

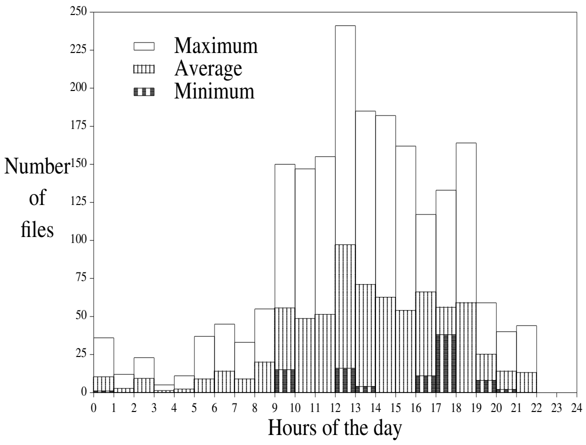

Figure 5: Hourly distribution of updates

The distribution of the number of updates to write-active files is shown in Figure 4. It is observed that 60% of the files are updated only once. The average number of updates received by a file is around 1.5. This implies that the total number of updates is almost equal to the number of files updated.

Figure 5 shows the 24-hour distribution of updates generated in the file system, for a one-week period. Similar distributions are observed for other weeks also. Activity is higher between 9 am and 6 pm (prime-time). It is found that in an one-week period, on the average, 4000 update transactions are received by the master mainframe server. There are approximately 100 updates in a prime-time hour. The peak activity is observed to be between 120 and 240 file updates per hour.

The low amount of write activity observed can be attributed to the library system behaviour where the library provides the users with a copy of the file for private use. The user may work on this private copy and make repeated updates to it over a period of hours, days or even weeks. A file is changed globally only when the user, after extensively testing the changes, sends a transaction of the modified source file to the master server.

It is also worth noting that the frequency with which users send updates to the master server may be distorted by the observed response times at the master server. (The master server is also responsible for managing files for other software products). For instance, some users report response times on the order of hours between submitting an update to the master server and receiving a reply, and this may cause them to submit fewer updates than they would if response times were shorter.

4.2.2 Active File Size Distribution

In a distributed environment, the size of the active files has impact on both the network traffic generated and file server utilizations. In this section, file sizes of read and write active source files are separately analyzed. The size distribution of the object files is also discussed.

In Figure 6, size distributions of read-accessed and write-accessed source files are shown. Comparing active file size distribution with the static file size distribution in Figure 2 and Table 1, active files are found to be larger in size. In the static characteristics, an average file size of 30 Kbytes was observed. In this section, average sizes of 48-51 Kbytes for read active and 80-90 Kbytes for write active files are found. Write active source files are on average approximately two times larger than read active source files.

The active object files, as shown in Figure 7, also have a higher average size of 35 Kbytes compared to the static mean file size of 24.9 Kbytes observed previously in Table 1.

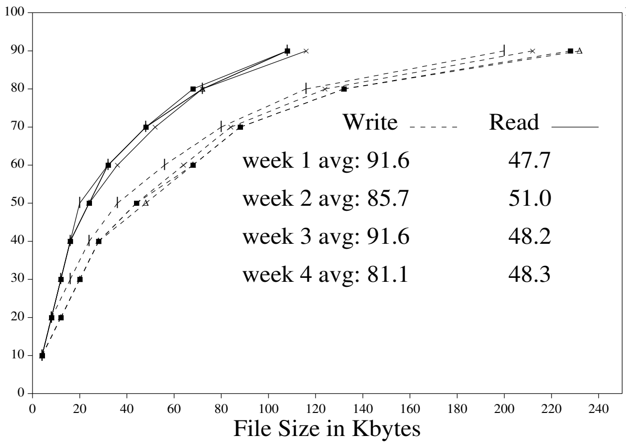

Figure 6: Size distribution of active source files

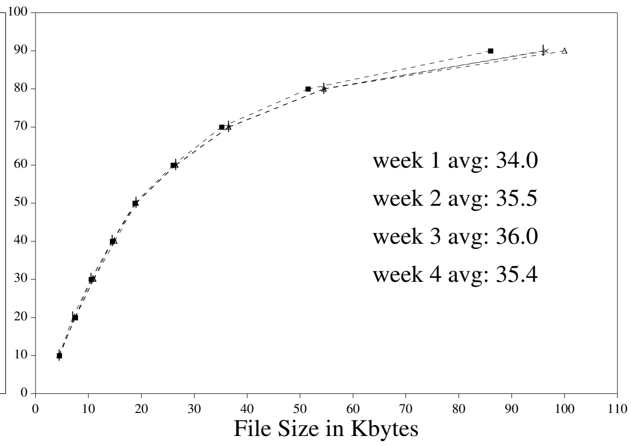

Figure 7: Size distribution of active object files

Past studies have also found active files to be larger than static files. For instance, Smith reports a static average file size of 7 Kbytes and an active average size of 19 Kbytes [13]. However, academic studies report smaller file sizes than past commercial environments, and this is consistent with the observations in this study.

The correlation between the size distribution of write-active files to read-active files have also been observed by Ramakrishnan et. al. who find that the average size of write-active files is always larger than that of read-active files [9]. This is similar to the observation made in this study that the write active file sizes are about twice those of the read active files.

5 Sharing between Workgroups

Understanding data sharing characteristics among users is extremely important for designing scalable distributed file systems. The complexity and performance of the policies used by the file system to manage user file accesses is directly dependent on the degree of sharing. For example, the information on file sharing is invaluable in devising algorithms for file caching. The policies to migrate or replicate files in a distributed system are also strongly influenced by the degree of sharing. Shared files that are updated by more than one user usually incur large management overhead for maintaining consistency. On the other hand, shared files that are read-only accessed impose a much smaller overhead.

In this section, sharing of files at the workgroup level is analyzed. The number of workgroups sharing a file define the degree of sharing for that file. Files that are accessed for both read and write from different workgroups are called read-write shared files. Since the library regulates all modifications to a file, no concurrent write sharing is observed. However, sequential write sharing, in which the writing workgroup changes after some period is allowed.

Due to the way in which object files are created and removed (described in Section 3) and to the fact that users never directly modify object files, only read-only sharing behaviour is observed for object files, with old versions disappearing once a week. The set of meta-files on the other hand, are used by all workgroups and updated every half-hour. The sharing patterns for meta-files are not analyzed further.

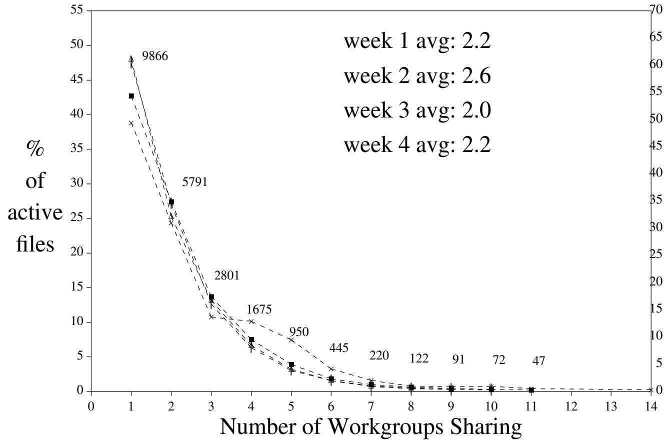

Figure 8: Degree of sharing of source files

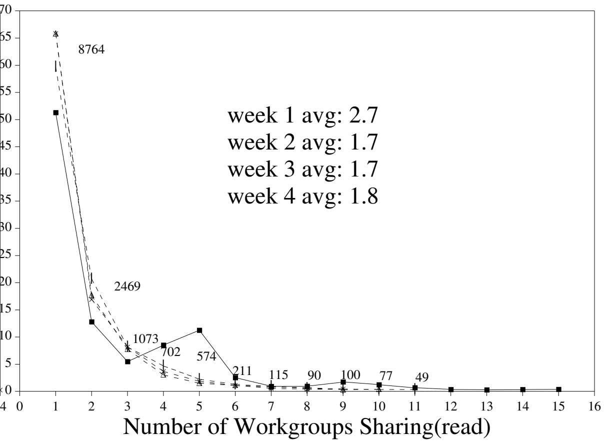

Figure 9: Degree of sharing of object files

The degree of file sharing among workgroups for the source and object files is shown in Figure 8 and Figure 9. The source files are read-only and read-write shared while the object files are read-only shared. The absolute number of files for the corresponding number of sharing workgroups is also shown. The average degree of sharing is calculated and labelled for each of the four weeks.

The following observations for the source files are drawn from Figure 8:

- During each week, 50-60% of the source files are shared by two or more workgroups.
- 95% of the files are shared by less than six workgroups.
- The average sharing level is approximately two workgroups.

Similar characteristics are observed for the object files in Figure 9. 40% of the active files are shared by two or more workgroups and the average degree of sharing is between 1.7 and 2.7 workgroups.

Since this study analyzes sharing at a high level (that of workgroups), it is difficult to compare these observations directly to those of past workload studies. However, it is clear that the amount of sharing in this environment is much higher than that observed in past workload studies.

For instance, Kure observes 23-47% of files accessed to be shared between two or more users in a centralized commercial workload [4]. Blaze reports that 60% of active files are shared between 2-20 users [2]. Ramakrishnan et. al. find that 30-55% files are shared between more than two processes [9].

5.1 Read-Write Sharing

In this section, we study the sharing of files that have are also being updated. At any moment, only a single workgroup is given permission to update any file. Files can however be read shared by other workgroups. The degree of sharing of source files that are read and write accessed is analyzed. The correlation between number of updates and degree of sharing is also discussed.

Figure 10 shows the number of reader workgroups that are sharing a file with a writer workgroup. The following observations are made:

- On the average, more than 50% of the write active files are shared.
- A file is shared by two reading workgroups on average.
- 90% of the files are shared by less than six workgroups.

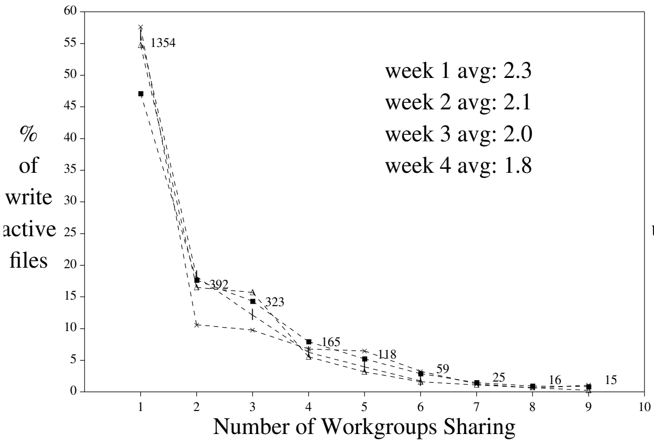

Figure 10: Read-write sharing of source files

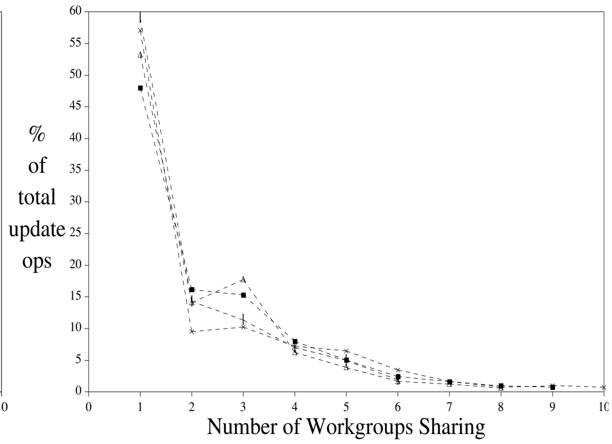

Figure 11: Distribution of updates to shared files

Figure 11 shows the correlation between the percentage of total update operations and the number of workgroups sharing a file. From Figure 10 and Figure 11, it is observed that files that are shared among a larger number of workgroups account for a smaller fraction of write active files and also receive less updates. In other words, as the sharing level increases the updates to the file decrease.

Similar characteristics are also observed in other commercial studies. Kure observes that shared files are accessed more than ordinary files and are more stable and updated less than ordinary write active files [4].

5.3 Sequential Write Sharing

The workgroups can sequentially write share files by requesting the library to change the write ownership. In this section, the fraction of write active files that are shared between two or more workgroups for each week are analyzed.

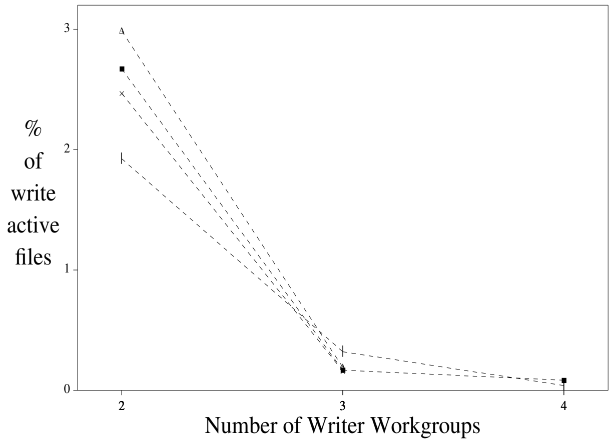

Figure 12: Sequential write sharing

It is observed from Figure 12 that only 2-3% of the write active files are sequentially write shared between two workgroups in an one-week period. The percentage of write active files shared sequentially between more than two workgroups is almost negligible (less than 0.2%) Overall, this reflects a stable write pattern where the writer workgroup does not change for a long period of time.

5.4 Correlation between Sharing and Size

The correlation pattern between the size and the degree of sharing of source files is shown in Figure 13. The average size of files is found to increase as the degree of sharing among workgroups increases. In other words, files that are widely shared among workgroups tend to be larger. One of reasons may be that large files contain more information and are thus of greater interest to a larger group of users.

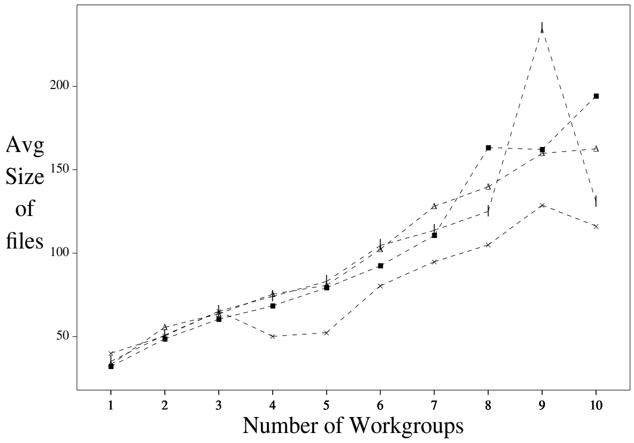

Figure 13: Correlation between size and sharing: source files

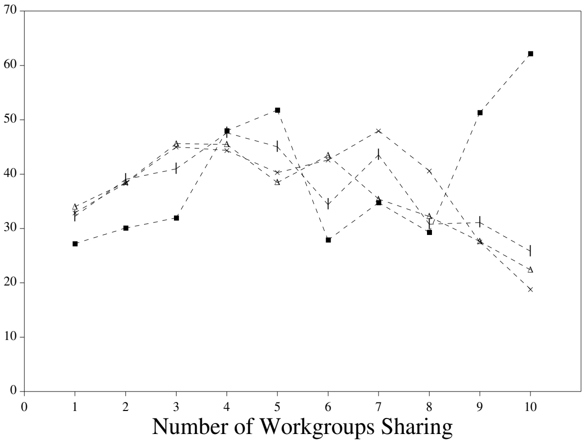

Figure 14: Correlation between size and sharing: object files

For object files, on the other hand, there does not appear to be any correlation between size and degree of sharing (Figure 14). This may be because object files are binary files having no direct correlation to the source files, i.e., one source file can contribute to multiple binary files.

Studies by Ramakrishnan et. al. and by Kure also observe that shared files are larger in size compared to non-shared files [9][4].

6 Conclusions

In this paper, we have presented the results of a workload study conducted in a large-scale distributed system. Although the study does not provide detailed trace analysis of file system activity, it nevertheless shows interesting file system characteristics of a large-scale industrial workload. For instance:

- A static snapshot of the file system shows that 50% of the files are over 12 Kbytes in size.
- More than one-third of the files are active in each week.
- Read active files are smaller in size compared to write active files.
- 90% of the active files are read accessed and 10% are write accessed.
- 50-60% of the active files are shared among two or more workgroups.
- On the average, 50% of the files updated are shared with two other reader workgroups (read-write sharing).
- A single workgroup remains as the only writer for 97% of the write active files (sequential write sharing), for a one-week period.
- As the number of workgroups sharing a read-write active file increases, the number of updates to the file decreases.
- Finally, an interesting correlation between file size and sharing is observed: files that are shared more are larger than less shared files.

The file characteristics observed reflect a number of differences from academic environments. The large-scale and extensive sharing of user files in the system are some of the key differences. Files are generally larger in size and are extensively read-write shared in comparison to small files (few Kbytes) and negligible read-write sharing reported by academic workload studies. Note that this study may also be constricted by the particular system. For example, transaction semantics might have reduced the number of potential updates.

This workload study is complementary to existing studies in two senses. First, it looks at a large distributed industrial environment, rather than an academic or small industrial environment. Second, it focuses on cluster level characteristics, rather than those of individual workstations in small clusters. Our observations are in many ways consistent with existing ones, suggesting that many characteristics at individual workstation level recurs at cluster level. On the other hand, new characteristics emerge at the cluster level. Our work therefore adds an important piece to the overall puzzle of distributed file system workload characterization.

References

[1] M. Baker, J.H. Hartman, M.D. Kupfer, K.W. Shirriff, J.K. Ousterhout, *Measurements of a Distributed File System*, Proc. of the 13th Symp. on the Operating Systems Principles, 25, 5, 198-212, ACM, Oct. 1991.

[2] M. Blaze, *Caching in Large-Scale Distributed File Systems,* PhD thesis, Princeton University, Jan 1992.

[3] R. Floyd, *Short-Term File Reference Patterns in a Unix Environment,* TR-177, University of Rochester, Rochester, New York, Mar. 1986.

[4] O. Kure, *Optimization of File Migration in Distributed Systems*, Ph.D thesis, Computer Science Division, University of California, Berkely, CA, Report No. UCB/CSD/88/413, April 1988.

[5] S. Majumdar, *Measurement and Analysis of Locality Phases in File Reference Behaviour,* Proc. of Performance'86 and ACM Sigmetrics, 180-192, ACM 1986.

[6] J. K. Ousterhout *et. al., Trace-driven Analysis of the Unix 4.2 BSD File System*, Proc. of the 10th Symp. on Operating Systems Principles, 15-24, ACM, Dec. 1985.

[7] J. K. Ousterhout, A. Cherenson, F. Douglis, M. Nelson, B. Welch, *The Sprite Network Operating System,* IEEE Computer, 21, 2, 23-36, Feb. 1988.

[8] Y. C. Pang, D. S. Gill, S. Zhou, *Implementation and Performance of Cluster-based File Replication in Large Scale Distributed Systems,* Proc. of the Second Workshop on the Management of Replicated Data, 100-103, Monterey, CA., Nov. 1992.

[9] K. K. Ramakrishnan, P. Biswas, R. Karedia, *Analysis of File I/O Traces in Commercial Computing Environments,* Proc. of ACM Sigmetrics and Performance'92 Conference, 20, 1, Newport, R.I., June 1992.

[10] H. S. Sandhu, S. Zhou, *Cluster-Based File Replication in Large-Scale Distributed Systems*, Proceedings of ACM Sigmetrics Performance'92 Conference, 20, 1, May 1992.

[11] M. Satyanarayanan, *The Influence of Scale on Distributed File System Design,* IEEE Trans. on Software Engineering, 18, 10, 1-8, Jan. 1992.

[12] M. Satyararayanan, *A Survey of Distributed File Systems,* Annu. Rev. Comput. Sci, Annual Reviews, Inc., 1989.

[13] A. J. Smith, *Analysis of Long Term File Reference Patterns for Application to File Migration Algorithms,* IEEE Transactions on Software Engineering, 7, 4, 403-416, IEEE, Jul. 1981.