universität
wien

MASTERARBEIT

Titel der Masterarbeit

A Study of the Evolution of Low Level File System Features

verfasst von

Christina Ochsenhofer, BSc

angestrebter akademischer Grad

Diplom-Ingenieurin (Dipl.-Ing.)

Wien, 2013

Studienkennzahl lt. Studienblatt:	A 066 935
Studienrichtung lt. Studienblatt:	Masterstudium Medieninformatik
Betreut von:	Univ.-Prof. Dipl.-Ing. Dr. Wolfgang Klas

Acknowledgments

First and foremost, I would like to thank the participants of this study who gave me permission to use snapshots of their file systems. Without them, this study would have been impossible.

I would like to express my sincere gratitude to my supervisor Univ.-Prof. Dipl.-Ing. Dr. Wolfgang Klas and my secondary supervisors Dr. Maia Zaharieva and Dipl.-Ing. Dr. Niko Popitsch for their support and encouragement in this research project. Furthermore, I want to thank Mag. Karin Wallner for her support in reviewing this thesis and for all her good ideas.

I would also like to thank my parents who have always supported me and gave me the opportunity to attend university at all. To my family and close friends, thank you for always believing in me.

Finally, I owe special thanks to Andreas, who has always supported, motivated and encouraged me.

Abstract

For understanding and evaluating file systems and their dynamics, it is important to know what kind of data is stored in the file systems, and how these data change. However, little is known about the amount and kind of data stored, or about their dynamics and organization. The development of advanced storage and organization tools, as well as many related research and development activities, would benefit from more data and knowledge regarding the evaluation of file systems.

This thesis aims to describe methodologies used for monitoring file systems. Two techniques for retrieving file system information and for detecting changes in the file system shall be introduced. Both techniques allow for gathering data about file systems, which can be used to study file system statistics. The study at hand focuses on what fractions of files in a file system are multimedia files and how these multimedia files as well as other files and directories are distributed in the file system. Furthermore, investigations about how file systems and the data stored in them changes over time and if there are file types or regions in the file system that are potentially more dynamic than others are performed.

The first technique for retrieving file system information performs real-time logging of file system changes and the second creates file system snapshots of the whole file system. On the one hand, real-time logging of the file system delivers very accurate data about the file system and its dynamics. On the other hand this technique can be used for observing small file system regions only as the technique needs a lot of system resources and gets inefficient when too many objects must be observed. When creating file system snapshots information about each file and directory found in the file system is stored. The file system snapshots are created at a certain point in time (once, daily, weekly, yearly, etc.) and by comparing consecutive file system snapshots, changes of file system related meta data can be observed. One drawback of this technique is that some changes happening between two snapshots can get lost.

In an experiment, the investigated technique of creating file system snapshots is used to gather file system information of 16 different file systems using a Windows or Mac OSX operating system. The file system snapshots are created weekly over a period of ten weeks. The snapshots are used to analyse file system information such as temporal changes in file and directory counts, file size and age, file-type frequency as well as storage capacity and consumption. Temporal changes in the file system are detected by comparing consecutive file system

snapshots.

Designers of file systems, as well as designers of backup or anti-virus utilities can benefit from more file system information as they can draw conclusions about what kind of data is stored and how often this data changes. This knowledge can help to increase the speed and reliability of file systems, backup processes, and anti-virus scanning.

Zusammenfassung

Um Dateisysteme und ihre Dynamik verstehen und beurteilen zu können ist es wichtig zu wissen welche Daten auf den Dateisystemen gespeichert sind, und wie sich diese Daten verändern. Dennoch ist nur wenig über die Menge und Art der gespeicherten Daten, sowie über deren Dynamik und Organisation bekannt. Die Entwicklung von fortschrittlichen Speicher- und Organisationswerkzeugen, sowie viele Forschungs- und Entwicklungsaktivtäten würden von mehr Daten und Wissen über diese Themen profitieren.

Die vorliegende Arbeit beschreibt Methoden die zum Überwachen von Dateisystemen verwendet werden können. Zwei Techniken für das Abrufen von Dateisysteminformationen sowie zum Erkennen von Änderungen im Dateisystem werden vorgestellt. Beide Techniken erlauben das Sammeln von Daten über Dateisysteme, welche für das Studieren von Dateisystemstatistiken verwendet werden können. Die vorliegende Studie konzentriert sich darauf herauszufinden welcher Anteil von Dateien auf einem Dateisystem Multimediadateien sind und wie diese Multimediadateien sowie andere Dateien und Verzeichnisse im Dateisystem verteilt sind. Des Weiteren werden Untersuchungen darüber durchgeführt wie sich Dateien und Verzeichnisse über die Zeit verändern und ob es Regionen im Dateisystem gibt die potenziell dynamischer sind als andere Regionen.

Die erste Technik für das Sammeln von Dateisysteminformationen führt eine Echtzeitprotokollierung von Dateisystemänderungen durch, die zweite Technik erzeugt eine Momentaufnahme, auch Snapshot genannt, des gesamten Dateisystems. Auf der einen Seite liefert die Echtzeitprotokollierung des Dateisystems sehr genaue Daten über das Dateisystem und seine Dynamik. Andererseits kann diese Technik nur zur Beobachtung von kleinen Regionen des Dateisystems verwendet werden, da die Technik große Mengen an Systemressourcen benötigt und ineffizient wird sobald viele Dateisystemobjekte beobachtet werden müssen. Beim Erstellen der Snapshots werden zu jeder gefundenen Datei und zu jedem gefundenen Verzeichnis eines Dateisystems Informationen gespeichert. Die Snapshots werden zu einem bestimmten Zeitpunkt (einmal, täglich, wöchentlich, jährlich, etc.) erstellt und durch das Vergleichen von aufeinanderfolgenden Snapshots können Veränderungen im Dateisystem beobachtet werden. Ein Nachteil dieser Technik ist, dass einige Änderungen die im Dateisystem zwischen zwei Snapshots geschehen, verlorengehen können.

In einem Experiment wird die Technik zum Erstellen von Snapshots

verwendet, um Informationen über 16 Windows und Mac OSX Dateisysteme zu sammeln. Die Snapshots der Dateisysteme werden wöchentlich, über einen Zeitraum von zehn Wochen erstellt und durch das Vergleichen von aufeinanderfolgenden Snapshots können Veränderungen im Dateisystem beobachtet werden. Die Snapshots werden verwendet um Informationen über Dateisysteme, wie zeitliche Änderungen in der Menge der Dateien und Verzeichnisse, die Dateigrößen, das Alter von Dateien, sowie die Häufigkeit von Dateitypen, die Speicherkapazität und der Speicherverbrauch zu untersuchen.

Dateisystemdesigner, sowie Entwickler von Backup oder Anti-Virus Techniken können von mehr Informationen über Dateisysteme profitieren, da sie Aufschluss darüber geben können welche Daten gespeichert sind und wie sich diese verändern. Dieses Wissen kann helfen die Geschwindigkeit und Zuverlässigkeit von Dateisystemen sowie von Backup und Anti-Virus Software zu erhöhen.

Contents

List of Figures

List of Tables

Acronyms

API	Application programming interface
Blob	Binary large object
FS	File system
FSO	File system object
HDD	Hard disk drive
NTFS	New technology file system
RAM	Random access memory
SSD	Solid-state disk
URI	Uniform resource identifier
URL	Uniform resource locator

Chapter 1

Introduction

1.1 Motivation

A substantial amount of today's multimedia contents is stored in local file systems (FS). However, little is known about the quantity of these data or about their dynamics and organization in local desktop environments. The development of advanced storage and organization tools as well as many related research and development activities are centred around the local desktop and would benefit from more information and knowledge regarding these topics.

The analysis of variations between FSs as well as repeated observations of FSs over a certain time period can help many parties with different concerns regarding FS usage and development. Knowledge about what kind of data is stored in the local desktop environments, how these data change over time, and whether there are file types or regions that are potentially more dynamic than others can support many different companies optimise their tools and systems. With this knowledge the performance of FSs can be increased since performance depends strongly on the characteristics and amount of the stored files. Developers of FSs can increase speed, quality, and reliability of the FSs if they have information such as type, size, and dynamics of the stored material.

Results of earlier studies [6, 24] show, that most of the files stored in FSs are rather small with a file size below 200 KByte. Common FSs are optimised for such small files. However, today, personal computers store a huge amount of multimedia files such as images, music files, video clips, and movies, that tend to have a file size above 200 KByte. This thesis analyses a selection of recent

FSs and compares the results to earlier studies ([26, 9, 6, 10, 24]) to find out common characteristics of FSs, how they differ, and how they have changed respectively.

In addition, developers of backup and anti-virus utilities can benefit from FS information as they can draw conclusions as to what kind of FS material is stored and how often these data change. Detailed information about the stored content and activities can help increase anti-virus scanning performance and malware detection processes as well as backup operations. By focusing on FS information, computer-based forensic tools reduce investigation time and complexity as this material assists in gathering suspicious files and activities. Visualisation of FS information improves the probability of locating criminal evidence. Viewing detailed information about files, directories, and the relations between them can also be very helpful to analyse huge and complex FS materials and find suspicious information.

Furthermore, the gathered FS information can be used to predict the amount of used and free disk space. Knowledge about how free space changes during FS lifetime supports disk manufacturers and suppliers of peer-to-peer systems as well as cloud storage systems to plan for the future. Information about FS dynamics can also assist in detecting programmes or activities causing a high amount of read/write operations on the hard disk. By optimizing read/write operations the loading times on magnetic hard disks (HDD) can be minimised, and the lifetime of solid-state disks (SSD) can be improved by reducing the amount of write operations.

The aim of this study is to examine the methodology of gathering and analysing FS statistics. Although the data collected in this study is not a representative sample, it is compared to results of previous studies in order to show how FS statistics have changed. Furthermore, the collected real-world FS data is analysed and the results are presented in order to show the organization and dynamics of current FSs.

1.2 Outline

This thesis is divided into four chapters that cover the following aspects.

Methodology

This chapter explains the methodology of monitoring FSs including data gathering and analysis. It presents a general workflow for FS monitoring

and describes the techniques used in this thesis. This encompasses the implementation and employment of a Java-based scanning application, which is used to gather real-world data of FSs. Furthermore, this chapter explains how FS events are detected and how the collected data is analysed.

Observations and results

Chapter 3 shows the organization and dynamics of local FSs by presenting results of the real-world data evaluation. These results contain file and directory observations such as file and directory count, file size and age, information about file and directory names, file name extensions, tree depth, etc. Moreover, information about space usage and temporal changes in the FS is presented.

Related work

In this chapter related articles and results published in previous studies are presented.

Conclusion and outlook

Finally, Chapter 5 provides a summary of the work and results presented in this thesis. It discusses limitations and provides some possible future research directions related to the study at hand.

Chapter 2

Methodology

This thesis is motivated by the question of how FSs and the data stored in them change over time and whether there are file types or regions (such as the "My Documents", "My Music", or "My Pictures" folders used on Windows operating systems) in the FS that are potentially more dynamic than others. Furthermore, this thesis investigates what fractions of files in an FS are multimedia files and if multimedia files are clustered in specific regions in the FS.

Results of this thesis are statistical data and insights about distribution and clustering behaviour of various file types in local desktop environments. In particular, statistical FS data such as amount, size and distribution of files and directories, and statistical data about the dynamics of FSs such as the number of FS events (*create*, *remove*, *update,* and *move*) is gathered and presented. To study FSs it is necessary to monitor the data stored in them and to detect how this data changes over time. The following list describes general steps which are performed in the course of this thesis in order to analyse FSs and FS events.

1. FS data collection:
   In the first step, FS data is collected. During this thesis research two different methods for FS monitoring have been identified:
   (1) **Real-time logging of FS changes**. With this technique it is theoretically possible to detect every FS change at the moment it happens. Information about who (operating system, specific programme, specific user) made what change (*create*, *remove*, *move,* or *update*) on a particular file or directory in the FS is collected. This technique provides real-time information about FS changes, but according to [19, 12, 11, 22, 15, 23] this

technique gets inefficient when monitoring a whole FS as this needs a lot of system resources. The technique is explained in detail in Chapter 2.1.1. (2) **Creation of FS snapshot files**, including information about the FS and about stored files and directories. This technique creates snapshots of FSs at a certain point in time (once, daily, weekly, etc.). These snapshots contain different FS statistics such as number of stored files and directories or space usage statistics as well as information about each found file and directory such as name, size, or last modification date. In order to detect changes in the FS at least two FS snapshots must be taken. By comparing two consecutive FS snapshots, changes in the FS can be detected. This study uses the technique of creating FS snapshots in order to gather FS statistics of 16 different FSs. 13 FSs use the operating system Windows and three FSs use Mac OSX. The snapshots are created weekly over a period of ten weeks resulting in 160 FS snapshots. The snapshots contain statistics about each found file and directory such as file or directory path, name, file extension, file size, last modification date, and others. The technique is explained in detail in Chapter 2.1.2. The complete list of gathered statistics as well as the implementation of this technique is presented in Chapter 2.1.3.

2. FS data analysis:
   In the next step, the collected FS data must be analysed. Different FS characteristics, such as average amount of files and directories, average file size, age, or information about how the monitored data have changed during the observation period can be gathered and calculated. See Chapter 2.2 as well as Chapter 3 for detailed information. In case FS snapshots are used for data collection, the FS changes must be calculated. These FS changes are categorized in FS events. Chapter 2.1.4 describes the events and how they are detected.

3. Presentation of results:
   For a better understanding of the data, it is recommended to present the gathered and analysed data in a graphical or tabular manner. Figures, tree-maps, special user interfaces, tables, and the like can be used to present the data in a way that is easy to read and understand. More information is presented in Chapter 2.2.

In an experiment, the technique of creating FS snapshots is used to gather

information from FSs of 15 voluntary participants and from the FS of the author of this thesis. Five participants use their FSs for private purposes only. Eleven participants are students and use the FS for private purposes as well as for university related work. The participants use desktop computers or notebooks. The snapshots are created weekly over a period of ten weeks resulting in 160 FS snapshots and information about more than 7×10^7 files and directories in total. The observation period of ten weeks was chosen as it appeared acceptable for the volunteers.

2.1 FS data collection

Recent research approaches apply mainly two methods for FS monitoring: (1) Real-time logging of the FS changes and (2) the creation of FS snapshot files. In this section both methods are described and tools and studies using these techniques are presented. Furthermore, the implementation used in this study for gathering FS information as well as the methods applied to analyse the gathered data are presented.

2.1.1 Real-time logging of whole FS

This technique detects and logs every FS change at the moment it happens. In addition, meta data about the detected event, such as date and time of the event or entity triggering the event (user, operating system, or specific programme), is logged. In this thesis four different events are described: *create*, *remove*, *move,* and *update*. A *create* event is detected when a new file system object (FSO) is created in the FS. In case an existing FSO is completely removed from the FS, a *remove* event is detected. A *move* event is detected when an existing FSO is moved to a different location. An *update* event is detected when an existing FSO is modified, for example when the content is altered.

Advantage: As mentioned in [11, 12, 15, 19, 22, 23], this technique provides real-time information about changes in the FS, such as information about created, deleted, updated, or moved FSOs, when and by whom an FSO was changed, and further data which can be used for different purposes and analysis. It is, for example, possible to detect programmes or activities causing a high amount of read/write operations on the hard disk. This information can help to decrease the number of read/write operations and prolong the life of HDDs and SSDs. When performing read/write operations on magnetic hard disks the

read/write head of the disk must be moved to the correct track first, and then it has to wait until the correct sector appears under the head. This mechanical process takes some time (standard HDDs in private computers have an access time around 9 ms [34]) and a lot of read/write operations can slow down the system. Avoiding unnecessary read/write operations can speed up the system. Blocks on an SSD can only be written and erased for a limited number of times so avoiding unnecessary write and delete operations can increase the disk's lifetime.

Disadvantage: This approach is not efficient enough to be used for watching a whole FS as today's FSs contain several tens of thousands FSOs. The technique implies that each FSO must be monitored, which is very inefficient, as the computer does nothing more than check the status of each FSO. As mentioned by [11, 22, 23], this permanent checking of the FSO status needs a lot of system resources and slows down the system. If there are too many FSOs and/or not enough resources, checking the status of each FSO can take too much time so events get lost. Furthermore, the created log files get very big and consume a lot of disk space. As described in [12, 15, 19, 22], events cannot be stored anymore and get lost or old events are overwritten in case the log files get too big, For this reason, most of the tools using this approach monitor only small FS regions, with a comparatively small number of FSOs but never the whole FS.

The following listing presents such tools and methods that can be used to monitor FS changes. None of the presented tools allow for monitoring a whole FS as the amount of FSOs is too high. The presented tools allow the selection of a directory, and all FSOs located in this directory are monitored.

- **Windows FileSystemWatcher** [19]: The Windows.Net application programming interface (API) listens to FS change notifications and raises events when an FSO changes. The FileSystemWatcher can be used to watch for changes in a specified directory selected by the user. The API can detect *create*, *remove,* and *update* events. *Move* events are not detected. The FileSystemWatcher is notified by the Windows operating system about changes using a buffer. This buffer can overflow, in case many changes occur in a short time. In this case changes are lost and not logged by the API.

- **System Change Log** [12]: This tool monitors specified directories in an FS for changes, and records a detailed log of file activities. System Change Log runs as a system service on Windows operating systems only.

The tool works with the proprietary new technology file system (NTFS) on locally attached drives. It detects *create*, *remove*, and *update* events. *Move* events are not detected.

- **Event detection using Python scripts** [11]: The presented scripts written in the programming language Python can be used to monitor *create*, *remove*, and *update* events in a user specified directory. *Move* events are not detected**.**
- **Java WatchService API** [22]: This Java API monitors directories specified by the user for *create*, *remove*, and *update* events. *Move* events are not detected. Each directory that should be monitored must be registered with the watch service. Furthermore, the user must tell the service which type of events should be detected.
- **inotify** [15]: This Linux API provides a mechanism for monitoring FS events in a user-specified directory. It can detect *create*, *remove,* and *update* events. Additionally, the API reports two special event types *moved_from* and *moved_to*. When these two properties are merged properly, *move* events can be detected.
- **Jpathwatch** [23]: This Java library monitors directories specified by the user for FSO changes. It can detect *create*, *remove,* and *update* events. *Move* events are not detected.

Currently, this technique of detecting every event at the moment it happens can only be applied for small FS regions but never for the whole FS as the process of monitoring all FSOs would use too many resources. The next section describes a technique that allows for scanning the whole FS.

2.1.2 Creation of FS snapshots

Since real-time logging of the whole FS is too time-consuming and requires too much system resources, FS snapshots are often used to collect and study real-world data of FSs. This technique allows collecting the state of an FS at a certain point in time. An application is used to traverse the directory tree of the FS and collect meta data of each found FSO. This process is called "scanning". The meta data includes attributes such as FSO name, size, time-stamps, etc. For privacy reasons some tools record the FSO names and paths in an encrypted format. However, information about FSO naming and FSO paths

allow for a more detailed analysis of FSOs and can help analyse how users name and organise their data. It is possible to calculate the mean length of file and directory names or to analyse the occurrence of special characters and numbers in FSO names. The scans can be performed once, or repeated periodically in order to get longitudinal FS information. At least two FS snapshots must be collected to be able to detect *create*, *move*, *remove*, and *update* events. The events and how they are detected are explained in Chapter 2.1.4.

Advantage: A lot of FS information as well as FS events can be gathered in a short period of time and in a resource-friendly manner. The scanning application used for this thesis is able to scan a whole FS with about 500 GByte of stored data in less than ten minutes.

Disadvantage: Only one snapshot of the FS at a certain point in time is provided. FS events must be calculated by comparing two consecutive scans, which means that at least two snapshots of an FS must be taken. During two scans FS events can get lost. For example a file is moved from location A to B and then moved back to location A. These *move* events are lost when they happen between two scans. If an FSO is updated several times, only the last *update* can be detected by the snapshot comparison. All other *update* events of this FSO which happened between two scans are lost. Furthermore, it is not always possible to determine when and by whom an event was triggered. For example, in case an FSO is completely removed from the FS, it is impossible to determine when and by whom the FSO was deleted.

This technique of creating FS snapshots has already been used in different studies such as [9, 6, 28, 13, 24]. Unfortunately, the used programmes are described only superficially. Therefore an accurate description of the snapshot creation and snapshot comparison is impossible. The following list presents studies which used this technique for creating FS snapshots.

- In 1999, Douceur and Bolosky [9] developed a simple programme that traverses the directory tree of each FS of a computer. The programme collects snapshots of FS meta data of Windows PCs in the commercial environment of Microsoft Corporation. Snapshots including meta data of each file and directory were collected. This meta data includes name, size, time-stamps, number of containing files and sub-directories in each directory, the parent-child relationship of nodes in the FS tree, as well as some system configuration information. File and directory names are collected in an encrypted form.

In a longitudinal extension of this study, Agrawal, Bolosky, Douceur and Lorch [6] used the programme to collect FS snapshots from more than 60,000 Windows PCs over a period of five years (from 2000 to 2004). The scans were performed annually on a voluntary basis in the commercial environment of Microsoft Corporation. They received very few matching snapshot pairs over the observation period. Only 18 FSs were scanned in each of the five years. The snapshots were used to study temporal changes in file size, age, file-type frequency, directory size, namespace structure, FS population, storage capacity and consumption, and degree of file modification.

- In 1994, Smith and Seltzer [28] collected daily snapshots from 48 FSs on four file servers over a period of ten months. A snapshot contained a summary of FS meta data including information about the size and configuration of the FS, the age, size and location of each file, and a map of free blocks in the FS.

- In 2008, Hicks et al. [13] developed an application that collected the entire contents of 40 personal file spaces. This data collection was performed only once. The tool gathered different file and directory information such as name, size, level in the hierarchical file space, creation and last modification date, etc.

- In 2011, Popitsch [24] scanned 15 Windows and Mac OSX FSs and created a feature vector for each found FSO containing file and directory information such as name, size, last modification date, level in the FS hierarchy, etc.

The study at hand uses the scanning programme from Popitsch [24] to gather FS snapshots of 16 FSs. The following section describes implementation and execution details of the scanning programme.

2.1.3 Implementation

For this thesis a Java-based application, called DiscScanner, is used to get real-world data snapshots of 16 different Windows and Mac OSX FSs. The DiscScanner was originally implemented by Popitsch in 2011 [24]. In order to make the usability of the application as easy as possible, the user interface was

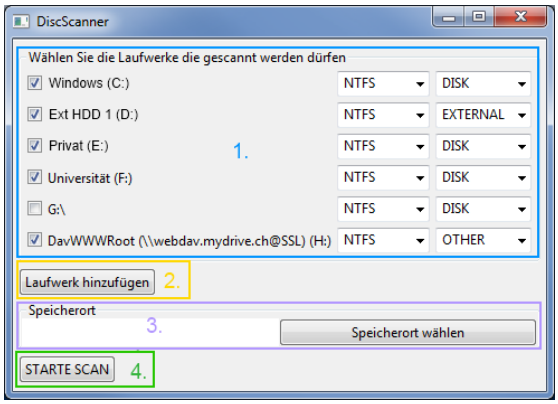


Figure 2.1: Graphical user interface of DiscScanner application.

redesigned. Furthermore, runnable jar files were created for the Windows and Mac OSX operating system, including 32-bit and 64-bit versions.

The application automatically detects all local and network drives (Area 1 in Figure 2.1). For every drive the user has to select the FS type and the drive type by using the drop down boxes on the right side. The application allows for the selection of the following FS types: NTFS, FAT32, HFS, HFS_PLUS, EXT2, EXT3, MIXED, OTHER. Available drive types are: DISK, EXTERNAL, OTHER. The FS types and drive types are only gathered for statistical reasons and do not affect the scan. The check boxes on the left side are used to define which drives are scanned. In case the application misses a drive or a location the user can manually add it by using the button "Laufwerk hinzufügen" (Area 2 in Figure 2.1), German for "add drive". Next, the user needs to select a location where the result files of the scan are to be stored (Area 3 in Figure 2.1). The scan is started by pressing the "STARTE SCAN" button (Area 4 in Figure 2.1), German for "start scan".

The DiscScanner creates two files. The first one includes the FS snapshot, the second one contains statistical data about the scanned FS drives and the scan itself. The FS snapshot file contains a feature vector for each found FSO. The feature vector includes the following file and directory meta data (see Figure 2.2 for an example snapshot):

1. **Uniform resource identifier (URI)**: file or directory path encoded as uniform resource locator (URL).

2. **isDirectory**: Boolean data type. *True* for directories and *false* for files.

3. **Name**: File or directory name.

4. **File name extension:** Suffix to the file name, separated by a dot, used to indicate the file format. For example *jpg*, *txt*, or *exe*.

5. **File size**: In bytes.

6. **Mime type**: Mime types are identifiers for file formats, they are managed by the Internet Assigned Numbers Authority (IANA) [1].

7. **Last modification date**: Unix time, or POSIX time (the number of seconds elapsed since January 1, 1970 [39]).

8. **Number of FSOs found in the directory.**

9. **isExecutable**: Boolean data type. *True* if FSO is executable and *false* otherwise.

10. **isReadable**: Boolean data type. *True* if FSO is readable and *false* otherwise.

11. **isWritable**: Boolean data type. *True* if FSO is writable and *false* otherwise.

12. **isHidden**: Boolean data type. *True* if FSO is hidden and *false* otherwise.

13. **Tree depth**: Depth of the FSO location in the FS hierarchy.

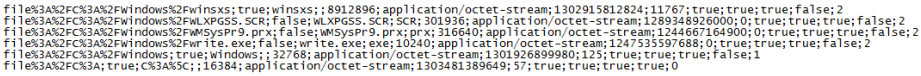


Figure 2.2: Screen shot of an FS snapshot file. The file contains the feature vector of each found FSO.

For processing reasons file paths and names are not recorded in an encrypted form. With this information it is possible to analyse how users name their FSOs and to calculate statistics about FSO names such as the mean FSO name length or the usage of special characters and numbers. Since paths are sensible data of the participants, the collected data set cannot be published.

Figure 2.3 shows an example of a statistical file. It includes date and time at which the scan was started, a list of drives that were scanned, as well as information about total space, free and usable space of each drive in bytes.

Furthermore, it contains the FS type ("Scanned root types") and drive type ("Scanned root device types") of each scanned drive, the scan duration measured in minutes and the amount of found FSOs, directories, and files. For collecting the FS data, the DiscScanner as well as a user manual are distributed to the participants via the file hosting service *dropbox* [2]. The participants also use *dropbox* to return the snapshot and statistical files. The snapshots of the 16 FSs are collected via voluntary participation. Each FS is scanned weekly over a period of ten weeks, resulting in 160 FS snapshots. All snapshots contain more than 7×10^7 FSOs in total and on average 4.5×10^5 FSOs are detected per FS. This data is used for further analysis as described in Chapter 2.2.

```
Scan started: 2011-05-15T21:30:23Z
Scanned roots: file:/C:/[total space 42949668864][free: 12704448512][usable: 12704448512],
               file:/D:/[total space 77081866240][free: 51938451456][usable: 51938451456]
Scanned root types: NTFS,NTFS
Scanned root device types: DISK,DISK
Scan duration [min]: 4
Found 141583 file system objects ( 18898 directories, 122685 files )
```

Figure 2.3: Screen shot of a statistical file. The file contains statistical information about the scanned FS drives and the scan itself.

2.1.4 Event detection

The data stored in FSs is altered by four different FS events: *create*, *move*, *remove*, and *update*. Some tools and APIs additionally use *rename* and *copy* events. In this thesis rename and copy events are categorized as *move* events since only the FSO path, but not the FSO content is changed in both cases. Events can be triggered either by the user, by programmes, or by the system. When real-time logging is used for observing FSs, the events are recorded at the time they happen. For further analysis this captured data can be processed. When snapshots of an FS are created at a certain point in time, at least two snapshots must be created in order to detect events that happened during the scans. By comparing two consecutive FS scans (scan_i and scan_{i+1}) the occurred FS events can be calculated. As mentioned above, FS events can get lost when using this technique, for example when an FSO gets updated several times within two scans, only the last *update* can be detected. Furthermore, it is not always possible to determine when and by whom an event was triggered. For example, in case an FSO is completely removed from the FS, it is not possible to determine when and by whom the FSO was deleted.

Event detection is performed by using an algorithm implemented by Popitsch

[24]. The algorithm is implemented in Java and packed into a jar file which is started via the command line. The algorithm uses two snapshot files ($scan_i$ and $scan_{i+1}$) and compares the feature vectors of each. A result file, which includes all detected events for $scan_i$ and $scan_{i+1}$is created. The following list explains the four different FS events which can be detected by the algorithm:

- ***Create***
  A *create* event is detected in case an FSO feature vector, which is not found in $scan_i$, is found in $scan_{i+1}$. Example shown in Figure 2.4: The text file frog.txt is created in the directory *C:\Test*.

Snapshot i

FSO	Path	Size
A	C:\Test\fileA.txt	10
B	C:\Test\ben.jpg	114
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Snapshot i+1

FSO	Path	Size
A	C:\Test\fileA.txt	10
B	C:\Test\ben.jpg	114
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13
E	**C:\Test\frog.txt**	**85**

Figure 2.4: Detect *create* event.

- ***Move***
  A *move* event is detected when an FSO feature vector is removed from $scan_i$ and a very similar FSO is created in $scan_{i+1}$. The similarity of a removed and created FSO is calculated by the file move detection method introduced by [24]. The method is based on pairwise similarity measures between FSO representations. Similarity values for each pair of removed (in $scan_i$) and created FSOs (in $scan_{i+1}$) are calculated. A set of similarity functions is used in order to calculate these similarity values. One of these functions uses the Levenshtein distance to calculate the similarity of file paths. The Levenshtein distance is a metric used for measuring the difference between two sequences (see [14, 35] for more information). In another function the last modification dates of each pair are used for a plausibility check. This plausibility check verifies that the last modification date of the newly created FSO is after the last modification date of the removed FSO. All similarity functions are combined in an algorithm that calculates one overall similarity value for each pair. Based on these similarity values another algorithm decides if the pair with the highest similarity value is accepted as *move* event or not. (For a detailed description of the file move detection method see [24].)
  Example shown in Figure 2.5: The image ben.jpg is moved from the

directory *C:|Test* to the directory *C:|Test|Pic*.

Snapshot i

FSO	Path	Size
A	C:\Test\fileA.txt	10
B	C:\Test\ben.jpg	114
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Snapshot i+1

FSO	Path	Size
A	C:\Test\fileA.txt	10
B	**C:\Test\Pic\ben.jpg**	**114**
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Figure 2.5: Detect *move* event.

- ***Remove***
  A *remove* event is detected when an FSO feature vector which is found in scan_i cannot be detected in scan_{i+1}. Example shown in Figure 2.6: The image ben.jpg is removed from the directory *C:|Test*.

Snapshot i

FSO	Path	Size
A	C:\Test\fileA.txt	10
B	C:\Test\ben.jpg	114
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Snapshot i+1

FSO	Path	Size
A	C:\Test\fileA.txt	10
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Figure 2.6: Detect *remove* event.

- ***Update***
  An *update* event is detected when some parameters of an FSO feature vector, such as file size or last modification date found in scan_{i+1} have changed compared to the feature vector of scan_i. Example shown in Figure 2.7: The content of the text file fileA.txt is updated and the file size has changed.

Snapshot i

FSO	Path	Size
A	C:\Test\fileA.txt	10
B	C:\Test\ben.jpg	114
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Snapshot i+1

FSO	Path	Size
A	**C:\Test\fileA.txt**	**25**
B	C:\Test\ben.jpg	114
C	C:\Test\cat.mp3	210
D	C:\Test\toDoList.txt	13

Figure 2.7: Detect *update* event.

The resulting file created by the algorithm is a text file including all detected events. An example result file is shown in Figure 2.8. For each detected event the following information is stored:

- Event type: *create*, *remove*, *move,* or *update*.

```
Detected 7004 events
detected CREATE,1311758059171,y2://609838,file:/C:/Users/Christina/testfile1.log,1.0
detected REMOVE,1311758059171,y2://602661,file:/C:/Users/Christina/testfile2.log,1.0
detected UPDATE,1311758059171,y2://605534,file:/C:/Users/Christina/testfile3.log,1.0
detected MOVE,1311758059171,y2://606138,file:/C:/old/testfile4.log,1.0,y2://604172,file:/C:/new/testfile4.log
...
```

Figure 2.8: Event detection result file. (Please note that the file paths are altered due to privacy reasons)

- Time-stamp when the event was detected: Unix time, or POSIX time (the number of seconds elapsed since January 1, 1970 [39]).
- Index: URI of the items.
  In case of a *move* event the index is stored for the original item and the new target item.
- Source path:
  In case of a *move* event the path is stored for the original item and the new target item.
- Confidence value:
  This value is used to express how confident the algorithm is that a particular event took place. The higher this value, the more confident it is (e.g. 1.0 means 100% confident).

These result files are used for further analysis such as calculating event frequencies.

2.2 Data analysis and presentation

The collected data of the 16 monitored FSs (two FSs using Windows XP, four using Windows Vista, seven using Windows 7, and three FSs using Mac OSX 10.6), is analysed using *R*, a free environment for statistical computing and graphics available under the GNU General Public License [5]. File and directory information, such as amount of files and directories, file size and age, tree depth, etc. is analysed.

For all results, tables, and figures, the prefix K, as in KByte, stands for 2^{10} Byte. Similarly M stands for MByte or 2^{20} Byte and G for GByte or 2^{30} Byte. All results are presented and discussed in Chapter 3.

Figures and tables are used to present the gathered data and results. All figures shown in this thesis are created using *R*. The results are split up into results for the Windows and Mac OSX operating system. Furthermore,

the results for both operating systems are presented. The Windows special directories are also of great interest and therefore some evaluations and figures deal with FSOs found in these special directories. For information about Windows special directories see [40]. Table 2.1 shows the general paths of the Windows special directories. The special directory Windows contains mainly operating system related data. The Program Files folder is the place where programmes are installed in general. It contains data relevant for the programmes such as executables, configuration files, log files, language files, etc. The Documents and Settings folder contains directories commonly used by the operating system to store a user's document, audio, video, and image files [37]. Therefore the folders My Documents, My Music, My Videos, and My Pictures are used. These special locations are recommended by the Windows operating system for organising and storing user generated documents and multimedia files and the usage is encouraged, for example by the Windows file browser that displays the folders as the first entries in the file tree.

Table 2.1: General paths of Windows special directories.

Folder name	Location
Windows	C:\Windows\
Program Files	C:\Program Files\ (contains 64-bit programmes) C:\Program Files (x86)\ (contains 32-bit programmes)
Documents and Settings	C:\Documents and Settings\ C:\Users\ (used since Windows 7)

For a more detailed analysis of the results the data of several investigations is broken up into the categories video, audio, image, database, binary large object (blob), documents, and other files as well as files without file extension. The categories and corresponding file extensions are listed in Table 2.2.

Table 2.2: File categories and corresponding file extensions.

File category	File extensions
Video	3g2, 3gp, asf, asx, avi, dps, flv, mov, mp4, mpeg, mpg, rm, swf, vob, wmv, m4v, mpe
Audio	aif, iff, m3u, m4a, mid, mp3, mpa, ra, wav, wma, mpeg3, ogg, aac, oga, spx
Image	bmp, gif, jpeg, jpg, png, psd, pspimage, thm, tif, tiff, yuv, ai, drw, eps, ps, svg, jp2, exif
Database	accdb, db, dbf, ldf, mad, mdb, mdf, ndf, ost, pst, pdb, sql
Blob	bak, bkf, bkp, dmp, gho, iso, pgi, rbf, sys, vhd
Document	ott, odt, dop, otp, odm, mcw, lwp, html,dotx, dot, docx, doc, docm, asc, ans, pages, pap, pdax, pdf, rtf, stw, sxw, tex, txt, uof, uoml, wpd, wps, wpt, xhtml, xml, xps, xht, ps, eps, xls, htm, ppt, pptx, odc, odf, ods
Other	All other file extensions (except non-existing file extension).
No extension	All files with no extension or where the extension has more than five characters.

Chapter 3

Observations and results

The previous chapter describes the process of FS data gathering and analysis. This chapter presents the observations and results found during analysis of the gathered FS data. Some of the results are compared with results from related studies ([26, 9, 6, 10, 24]) published over the past 30 years. These five related studies have been chosen because they have complete data about file and directory counts and file size. Table 3.1 summarises some basic information about these studies.

Table 3.1: Basic information about related studies published over the past 30 years. (*) An FS snapshot was created once a year, over a period of five years. (For only 18 FSs, snapshots were created in each of the five years.) (**) An FS snapshot was created once a week, over a period of ten weeks.

Study	Year of observation	Number of FSs	Operating system	Number of scans per FS
Satyanarayanan [26]	1981	1	Digital PDP-10	1
Douceur et al. [9]	1988	10,568	Windows	1
Agrawal et al. [6]	2000-2004	63,398	Windows	(*)
Evans et al. [10]	2002	22	Windows, Linux	1
Popitsch [24]	2011	15	Windows, Mac OSX, Linux	1
This study	2011	16	Windows, Mac OSX	10 (**)

3.1 File information

This section presents different observations and results in the context of file information such as file count, file size, and file age. Furthermore, the results are compared to results published in earlier studies ([26, 9, 6, 10, 24]).

3.1.1 File count per FS

Previous studies ([26, 9, 6, 10, 24]) show that the file count per FS is continually growing from year to year. FS storage capacity is nearly constantly increasing and getting cheaper, which means that people have more space for collecting and storing files on their computers. FSs must be able to maintain and process this high amount of files, which makes it important for FS designers to have information about file count trends. In addition, programmes scanning the FS, such as virus scans or backup programmes, can take advantage of this information. If the file count increases, the file checks must perform more efficiently in order to keep the scanning time as low as possible.

In the study at hand the file count has not changed significantly and shows no clear trend, which may be due to the short observation period of only ten weeks. Figure 3.1 shows the file count distribution per FS and it includes information about the used operating systems Windows XP, Windows Vista, Windows 7, and Mac OSX. The figure shows that the number of files per FS remains relatively

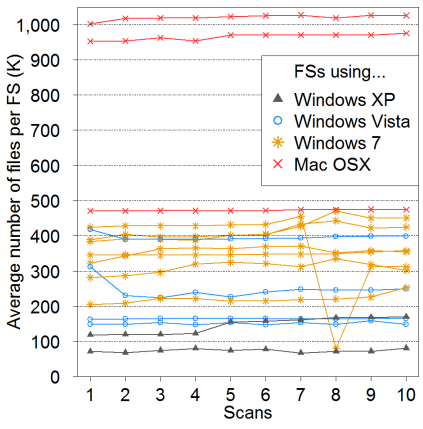


Figure 3.1: File count distribution per FS and per used operating system.

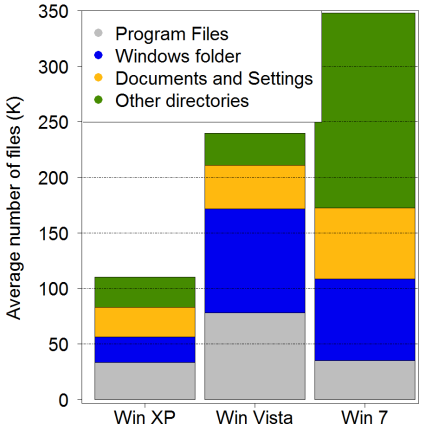


Figure 3.2: File count distribution in the Windows special directories grouped by the used operating system.

constant over the observation period, but the amount of files is different for each operating system. FSs using the oldest scanned operating system Windows XP (released in 2001) hold the least number of files, followed by Windows Vista (released in 2006) and Windows 7 (released in 2009). This observed difference can be explained by the consideration that FSs using an older operating system such as Windows XP often use older hardware such as HDDs with lower storage capacity. HDDs with lower storage capacity cannot store the same amount of FSOs as HDDs with higher storage capacity.

Figure 3.27 from Chapter 3.3.1 illustrates that the total available disk space is, at least for the FSs observed in this study, related to the operating system used. The observed FSs using an older operating system have less available disk space than the observed FSs using a newer operating system. Figure 3.2 shows that this assumption also applies for the Windows special directories. In observed FSs using Windows XP less files are stored in the Windows special directories than in observed FSs using Windows 7. Figure 3.3 shows the average number of files and directories per FS grouped by the operating system used. This figure also shows differences between the observed operating systems. However, it is not clear why the amount of files stored in observed FSs using the Mac OSX operating system is around two times higher than in observed FSs using a Windows operating system, whereas the available disk space and the mean fullness are nearly equal. As stated in Table 3.8 the mean storage capacity for observed Windows 7 FSs is 400 GByte and for Mac OSX FS 455 GByte and the mean fullness is 39% for Windows and 38% for Mac OSX FSs.

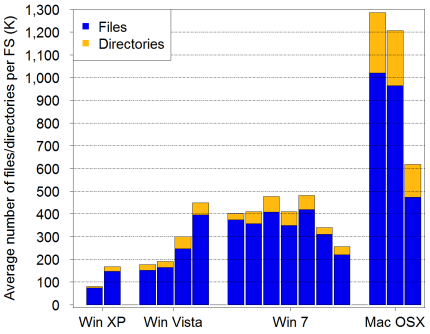


Figure 3.3: Average number of files and directories per FS grouped by the used operating system.

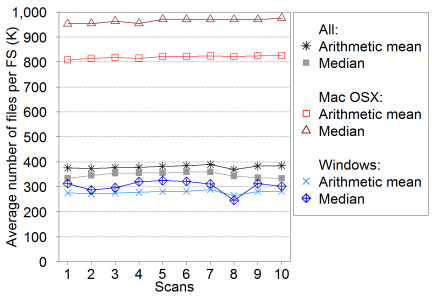


Figure 3.4: Arithmetic mean and median file count distribution per FS.

This peculiarity is a potential topic for further analysis and studies.

Figure 3.1 shows a decrease of the file count for one of the Windows 7 FSs at the eighth scan. This decrease happened because one of the participants completely restored his machine between the seventh and eighth scan. This decrease is also visible in Figure 3.4 which shows the measured arithmetic mean and median file count distribution. The mean and median file count distributions for Windows and Mac OSX FSs show no radical change over the observation period, except for the decrease at the eighth scan. The results of the mean file count published in earlier studies are shown in Figure 3.5. The graph shows an increasing trend of the file count for all operating systems observed. Comparing the measured mean file count results of Windows operating systems from Douceur et al. [9] (1998) or Agrawal et al. [6] (2000) with the results of this study (2011) an increase of the mean file count by more than 870% can be observed.

Table 3.2 presents the distribution of audio, video, image, database, blob, documents, and other files found in the observed FSs. The investigated categories are explained in detail in Table 2.2. Table 3.2 shows that around 25% of all detected files are multimedia files. In Windows FSs the percentage of multimedia files is around 29% and in Mac OSX FSs around 19%. Furthermore, the table shows that image is the most frequently found multimedia file category. Most of the audio, video, and image files found in the observed Windows FSs

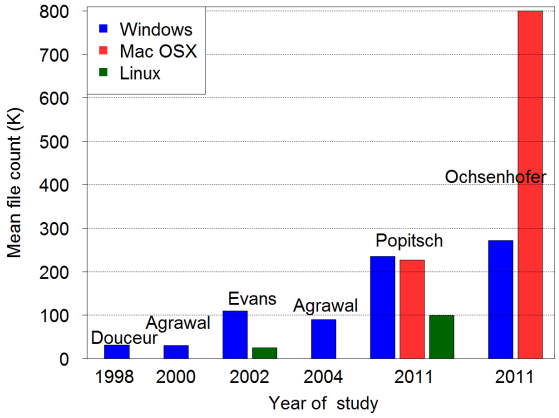


Figure 3.5: Comparison of mean file count results published within the past 13 years. The figure contains the published results of Douceur et al. (1998) [9], Agrawal et al. (2000 - 2004) [6], Evans et al. (2002) [10], Popitsch (2011) [24], and this study (2011).

are stored in the Documents and Settings folder. This suggests that many of the observed Windows users store their multimedia files in the by Windows recommended folders located inside the Documents and Settings folder. Further analyses of the multimedia file distribution show that many audio files are found on a secondary partition and on external HDDs. The external HDDs are mainly used for backups and most likely contain a backup of the multimedia file collections. The multimedia file distribution for the observed Mac OSX FSs shows that a majority of audio and video files is stored in the Users and Library sections. The analysis of the multimedia file distribution shows that image files are scattered all over the FS, while audio and image files are clustered in a few regions.

Table 3.2 further shows that on observed Windows and Mac OSX operating systems around 8% of all files have no file extension. The Documents and Settings folder contains most of the files without a file extension. The AppData or Application Data folder, located in the Documents and Settings folder contains programme related data such as programme settings, cookies, temporary files created by applications, etc. Many of these files use cryptic file names without a file name extension such as "vpCA65TBOJ", "drvlog1", or "Archived+History". File name extensions are analysed in detail in Chapter 3.1.6.

Table 3.2: Percentage distribution of different file categories in observed FSs.

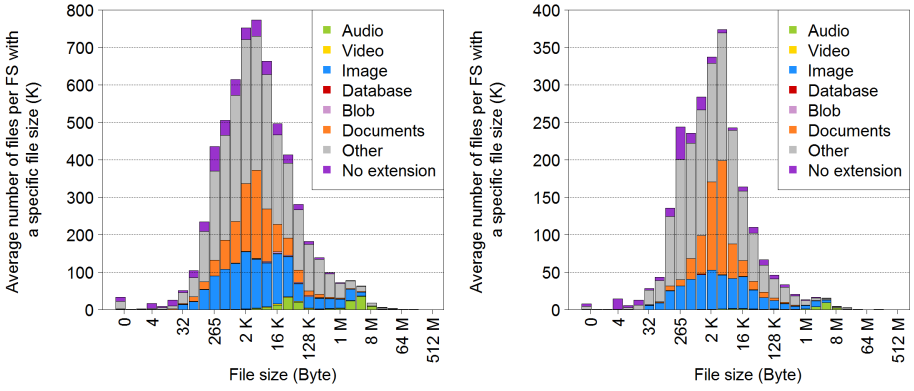
	All	Windows	Documents and Settings	Program files	Windows folder	Mac OSX
Average file count	353,431	258,584	40,252	44,942	71,316	764,435
Audio	2.71%	3.90%	5.12%	0.32%	0.66%	0.96%
Video	0.41%	0.57%	0.87%	1.56%	0.12%	0.17%
Image	21.95%	24.52%	37.43%	24.58%	2.76%	18.19%
Database	0.22%	0.23%	0.33%	0.09%	0.20%	0.20%
Blob	0.40%	0.65%	0.19%	0.19%	1.92%	0.03%
Documents	17.89%	16.44%	18.72%	19.40%	2.46%	20.02%
Other files	56.42%	53.70%	37.34%	53.86%	91.88%	60.42%
Sum	*100%*	*100%*	*100%*	*100%*	*100%*	*100%*
No file extension	9.82%	8.32%	28.55%	7.44%	0.92%	7.49%

3.1.2 File size

In this section results regarding file size are presented. Figure 3.6a shows the mean file count per FS with a specific file size. In addition, the distribution of audio, video, image, database, blob, documents, and other files as well as files with no file extension is shown. The investigated categories are explained in detail in Table 2.2. Furthermore, figures with the distributions for Mac OSX FSs only (Figure 3.6b), Windows FSs only (Figure 3.6c), and Windows special directories (Figures 3.6d, 3.6e, and 3.6f) are presented.

Comparing the Windows results (Figure 3.6c) to the Mac OSX results (Figure 3.6b) shows that the distribution of file types with a specific file size is very similar on both operating systems. Both figures show a maximum of files with a specific file size between 265 Byte and 16 KByte. In this region most of the documents and images are located. In addition, this file size region contains a high amount of files categorised as "other" files. Both figures show an accumulation of audio files between 1 MByte and 8 MByte. An accumulation of audio files between 16 KByte and 128 KByte can be observed in the Windows results only.

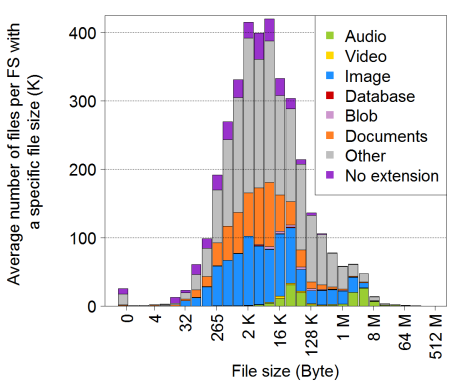
As already presented in Table 3.2, Figure 3.6d shows that the Program files folder contains most of the video files. A majority of these video files are small, with a file size below 1 MByte. Furthermore, the Program files folder contains many small image and document files with a file size between 265 Byte and 16



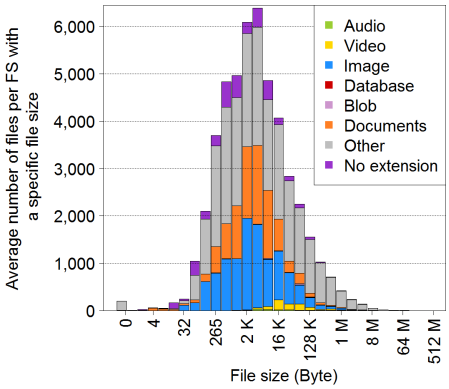
(a) All FSs (Windows and Mac OSX).

(b) Mac OSX FSs only.

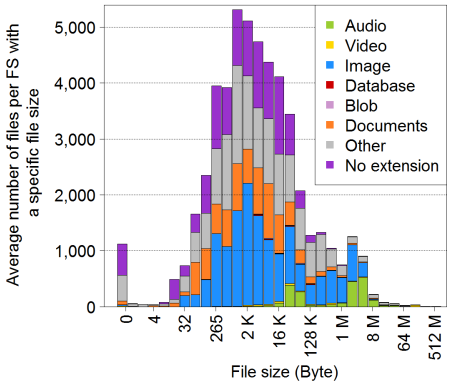
Figure 3.6: Average amount of files per FS with a specific file size in all FSs, Mac OSX, and Windows FSs and in the Windows special directories.



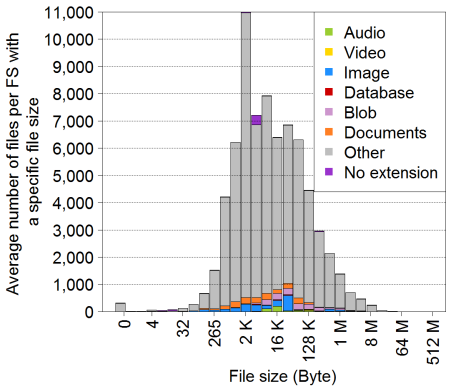
(c) Windows FSs only.



(d) Program files folder.



(e) Documents and Settings folder.



(f) Windows folder.

KByte.

Most audio and image files are stored in the Documents and Settings folder as shown in Figure 3.6e. As described in Chapter 2.2, this location offers the My Music and My Pictures folders to store the user's music and image files. Figure 3.6e shows an accumulation of audio files between 16 KByte and 128 KByte as well as between 1 MByte and 8 MByte. Standard audio files that are part of the music collection stored in the FS have a size between 1 MByte and 8 MByte. Very short audio files are mainly used by the operating system, programmes, or games, for example to notify the user when some error happens or when a task is finished. These short audio files have a small file size between 16 KByte and 128 KByte. The results regarding the My Music and My Pictures folder show that a high amount of the observed Windows users use the special folders provided by the Windows operating system to store and organise their image

and music collections.

According to Figure 3.6f most of the files in the Windows folder fall under the category "other". These are operating system relevant files such as dll files (file name extensions and their distribution are explained in detail in Chapter 3.1.6). Furthermore, the Windows folder contains a higher amount of blob files than the other Windows special folders. Although their name suggests differently, most of the found blob files are smaller than 1 MByte. Only some of them are big, with a size above 100 MByte. There is also an accumulation of audio files with around 16 KByte inside the Windows folder. As described above, these audio files are mainly used by the operating system or by installed programmes for notifications or other short sound effects.

A percentage distribution of file size used by special file types in all FSs, Windows FSs, Windows special folders, and Mac OSX FSs is presented in Figure 3.6. It illustrates that the sum of audio, video, image, and blob files use more than 50% of the total used space. As shown in previous figures FSs contain a high amount of images and documents, but a comparatively small amount of audio, video, and blob files. In total the size of all audio, video, and blob files is much higher than the total size of document and image files.

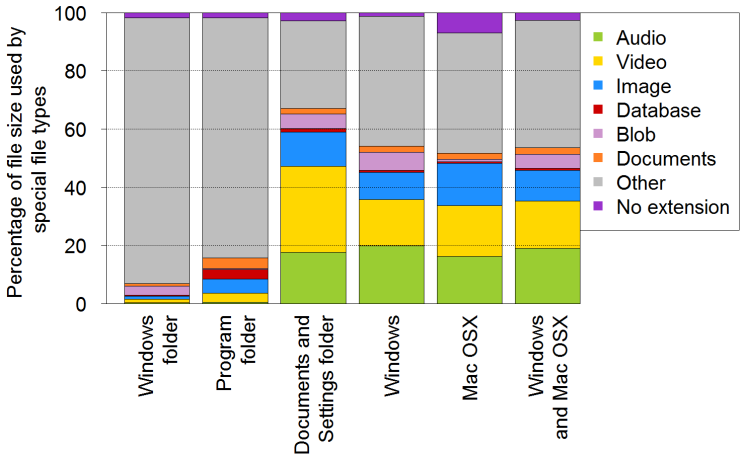


Figure 3.6: Percentage distribution of file size used by special file types.

A mean file size of 349.6 KByte is measured for all observed FSs. For Windows FSs only the mean files size is 464.6 KByte and 196.7 KByte for Mac OSX FSs. Figure 3.7 shows that previous studies such as [26, 9, 6, 24] also measured a small mean file size. In addition, Figure 3.7 shows that the

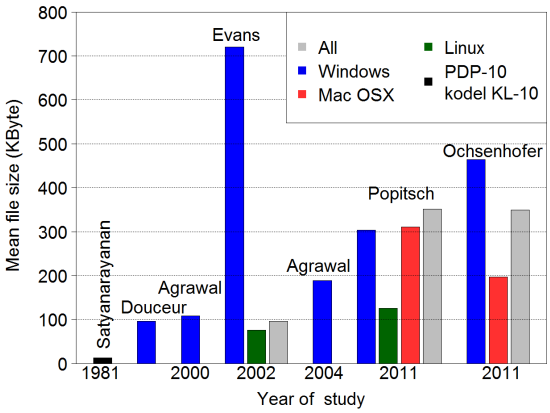
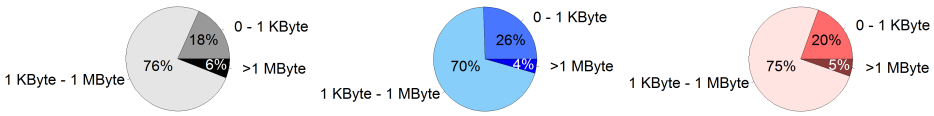


Figure 3.7: Comparison of mean file size results published within the past 30 years. The figure contains the published results of Satyanarayanan (1981) [26], Douceur et al. (1998) [9], Agrawal et al. (2000 - 2004) [6], Evans et al. (2002) [10], Popitsch (2011) [24], and this study (2011).

mean file size has increased during the past 30 years. When comparing the result of observed Windows FSs from Douceur et al. [9] (reported 96 KByte in 1998) to the results of this study (reported 464.6 KByte in 2011) the mean file size has increased by more than 440%. The results of Evans et al. [10] show a much higher mean file size (720 KByte) for Windows FSs than the other studies. The data set used in the study contained FSs with large collections of audio and video files which explain the high mean file size. When removing the multimedia files from their Windows data set, the median file size is reduced to 160 KByte, which fits well into the increasing trend.

Moreover, the calculated median file size is 3.4 KByte for all FSs, 5.2 KByte for Windows, and 1.6 KByte for Mac OSX FSs. These results show that most of the files stored in the observed FSs are small (3.4 KByte or smaller). The median file size reported in previous studies ([6, 10, 24]) is also very similar to the one in this study and shows that most of the files are small (Agrawal et al.



(a) Files of all FSs. (b) Files of Windows FSs. (c) Files of Mac OSX FSs.

Figure 3.8: File size distribution.

[6] reported 4 KByte in 2000, Evans et al. [10] reported 2.2 KByte in 2004, and Popitsch [24] reported 2.3 KByte in 2011). Figure 3.8 shows the distribution of these small files in more detail. Around 18% of all files have a file size between 0 KByte and 1 KByte. Most of the files, about 76%, have a file size between 1 KByte and 1 MByte, and only 6% of the files have a size greater than 1 MByte. The amount of files with zero file size found in this study, which make up about 0.6% for all scanned FSs, is smaller compared to the results of Agrawal et al. [6] (1-1.5%) from 2000 to 2004 and Popitsch [24] (1.8%) in 2011.

3.1.3 Last modification date

During FS data gathering the last modification dates of the FSOs were collected. With this information the interval between the last modification of an FSO and the instant of data collection can be calculated. Figure 3.9 shows the distribution of files by last modification date for all, Windows, and for Mac OSX FSs. The mean interval is 2.5 years for Windows FSs, 2 years for Mac OSX, and 2.3 years for all FSs. A median interval of 1.8 years is calculated for Windows FS, 1.9 years for Mac OSX, and 1.9 years for all FSs. As confirmed by previous studies such as [13] and [24] a majority of the files was last modified more than a month before the FS scan.

Figure 3.10 shows the distribution of file age for the Windows special directories. Most of the FSOs in the Windows directory were last modified between six months and five years before the FS scan, whereas a high amount of files were last modified between one and two years before the FS scan. The

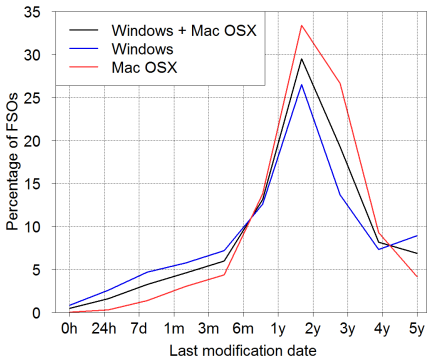


Figure 3.9: Distribution of last modification dates.

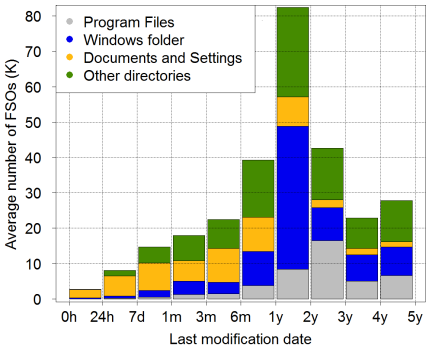


Figure 3.10: Distribution of last modification dates for FSOs in the Windows special directories.

Windows directory, with the typical path C:\Windows, is the location where the Windows operating system is installed. It furthermore contains help, driver, temporary, and other system related files. The content of the Windows directory is mainly updated when some operating system related parameters change or when an update is installed. Unfortunately we do not know exactly how old the scanned computers are, respectively when the operating system was installed the last time. Since the operating system Windows 7 was released by the end of 2009, all seven scanned FSs using the Windows 7 operating system could not have been installed more than two years before the scans were performed. This may explain the peak of files with an age between one and two years in the Windows directory. Additionally, we can suggest that most of the files with an age between six months and five years belong to one of the remaining six FSs using a Windows installation (two Windows XP and four Windows Vista installations). This could mean that a huge amount of Windows operating system related files is not altered after the installation has been completed.

3.1.4 File and directory name

This section summarises information about found file and directory names. For analysing the file names of the data set the file name extensions including the separation dot are removed (e.g. filename.txt → filename). Figure 3.11 shows

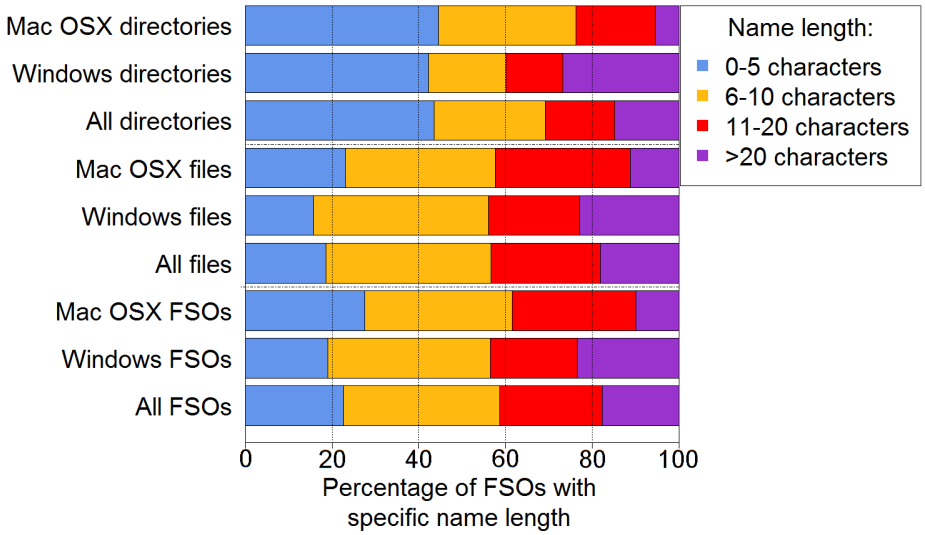


Figure 3.11: Distribution of file and directory name lengths.

the percentage distribution of FSO name lengths for all, Windows, and Mac OSX FSs. In addition, the distributions for file and directory names are plotted. The figure shows that directories in Windows and Mac OSX FSs use shorter names than files. More than 40% of the directory names, no matter if in Windows or Mac OSX FSs, use between zero and five characters. Only between 16% in Windows FSs and 23% in Mac OSX FSs of the file names use such short names. Most of the file names, 40% in Windows FSs and 34% in Mac OSX FSs, have a file name length between six and ten characters.

The mean name length of files in Windows FSs is 15 characters, and 12 characters in Mac OSX FSs. For directories the mean name length is 13 in Windows FSs and eight in Mac OSX FSs. The median file name length is ten in Windows and Mac OSX FSs. And the median directory name length is nine for Windows and seven for Mac OSX FSs. This shows, as confirmed by the results from [24], that most of the FSO names are short.

Figure 3.12 plots the distribution of letters, numbers, and special characters used in the FSO names. The figure shows that more than 40% of the Windows FSO names contain special characters such as: ! # $ % & ' ( ) + , - . ; = @ [ ] ^ _ ` { } ~. Please note that the following characters are not allowed in FSO names: / \ : * ? < | > ". In Mac OSX FSs the percentage of files containing letters only (53%) and numbers only (8%) is higher compared to the results of the Windows FSs. In Windows FSs more than 50% of the FSO names contain numbers or special characters. This indicates that numbers and special characters are used frequently for organising and structuring data, for example adding version or date information to the FSO name.

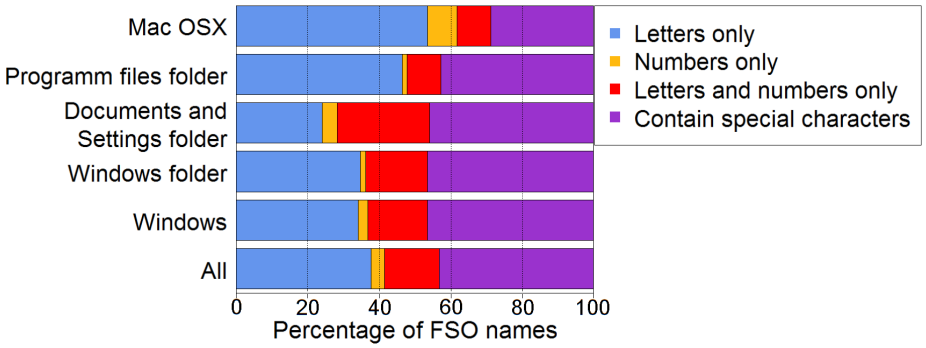


Figure 3.12: Distribution of letters, numbers, and special characters in file and directory names.

3.1.5 Additional FSO parameters

This section presents observations regarding the additional FSO parameters *executable*, *writable*, *readable*, and *hidden*. These boolean permission parameters can be set for every file and directory. Table 3.3 shows the mean percentage of *executable*, *writable*, *readable,* and *hidden* FSOs measured in all FSs, in Windows, and Mac OSX FSs only. The percentage of *executable* files in Mac OSX FSs is lower than in Windows FSs. Please note that this parameter is not only used for executable files, such as files with the extension *exe*, but also for directories to indicate whether the entries of a folder can be accessed or not [29]. The percentage of *writable* FSOs differs between Windows and Mac OSX, which is again higher in Windows FSs in comparison to Mac OSX. With 99.9% the percentage of *readable* FSOs is the same in Windows and Mac OSX FSs. Less then 1.4% of the files are hidden, which means that they are generally hidden from users by programmes or the operating system. This shows that nearly all FSOs are visible and readable to the user. A high amount of the files is writable and can be altered by the user.

Table 3.3: Distribution of additional FSO parameters.

		All	Windows	Mac OSX
Average number of files		452,917	318,245	1,036,495
Executable	true	87.5%	99.9%	33.7%
	false	12.5%	0.1%	66.3%
Writable	true	92.7%	97.7%	71.2%
	false	7.3%	2.3%	28.8%
Readable	true	99.9%	99.9%	99.9%
	false	0.1%	0.1%	0.1%
Hidden	true	1.1%	1.4%	0.2%
	false	98.9%	98.6%	99.8%

3.1.6 File name extension

This section takes a closer look at the file name extensions and their distributions. During file name extension analysis all file names having no dots or more than five characters after the last dot, are taken into the category "no extension", represented with the symbol Æ. The typical usage of the most popular file name extensions found can be looked up in Table 3.4.

Table 3.4: Typical usage of file name extensions. The file name extension information is taken from [3] and [4].

Extension	Typical usage
avi	An audio video interleaved (AVI) file is a multimedia container format for audio and video introduced by Microsoft.
cat	Catalog file format used to index data from various locations.
cr2	Raw camera image created by Canon digital cameras which are saved in a format based on the tagged image file format (TIFF) specification.
dat	Data file which may contain data in text or binary format.
db	Database file that stores data in a structured format.
dll	Dynamic-link library (DLL) files are compiled library files containing a set of procedures and/or drivers referenced and executed by Windows programmes
dmg	Mac OSX disk image file which mounts a virtual disk when opened.
elc	Compiled Lisp file created by Emacs, a customizable text editor.
emlx	E-mail message saved by Mac OSX Mail programme.
gif	Image files using graphical interchange format (GIF) are often used for web graphics, small images, or images that contain text, such as navigation buttons.
hds	Hierarchical data system (HDS) files used to store a hierarchical structure of data.
htm(l)	Web page coded in hypertext markup language (HTML) that can be displayed in a web browser.
ico	Image format used for storing icons.
inf	Plain text configuration file.
jpg	Compressed image format standardised by the Joint Photographic Experts Group (JPEG).
js	JavaScript source code file.
log	Log file used by various operating systems and programmes.
m	Class implementation file used by programmes written in Objective-C.
mk	Makefile used by software compilers and linkers for building programme executables from source files.
mov	Multimedia format used for saving movies and other video files.

mui	Multilingual user interface (MUI) file contains resources that allow the Windows interface to be changed to different languages.
mum	Windows Vista update package.
mp3	Audio file using compressed audio format developed by the Moving Picture Experts Group.
nib	Application support file created by Interface Builder, a software development programme.
pak	Compressed archive file.
pdf	File using the portable document format (PDF).
pnf	Windows precompiled setup information or portable network graphics frame bitmap.
png	Image file using the portable network graphic (PNG) format.
rar	WinRAR compressed archive.
src	Source code file.
svgz	Scalable vector graphics file.
sys	Windows system file.
s2mi	Cache file for the computer game Starcraft 2.
tif(f)	Image using tagged image file format (TIFF).
tmp	Temporary file usually serves as a backup or cache file.
txt	Plain text file.
vdi	Virtual drive format.
wav	Audio file format used for storing waveform data.
xml	File that contains extensible markup language (XML).
yaml	File created in the YAML ain't markup language (YAML) format.
zip	Zip file compressed archive.

Figure 3.13 shows, for the ten most popular file name extensions in terms of file count, the fraction of files having this extension. There are some differences between Windows and Mac OSX FSs such as the fraction of Æ extensions which is smaller in Windows FS, with about 18%, than in Mac OSX FSs with around 25%. The extension *mui* (typical usage can be looked up in Table 3.4) is only used by the Windows operating system and therefore it only shows up in the statistics for Windows FSs. The same applies for the extension *nib* which is only used by the Mac OSX operating system. The most common file extensions, in terms of file count, are image (*png*, *jpg*, *gif*), *html*, *xml*, or operating system related (*nib*, *mui*, *dll*) extensions.

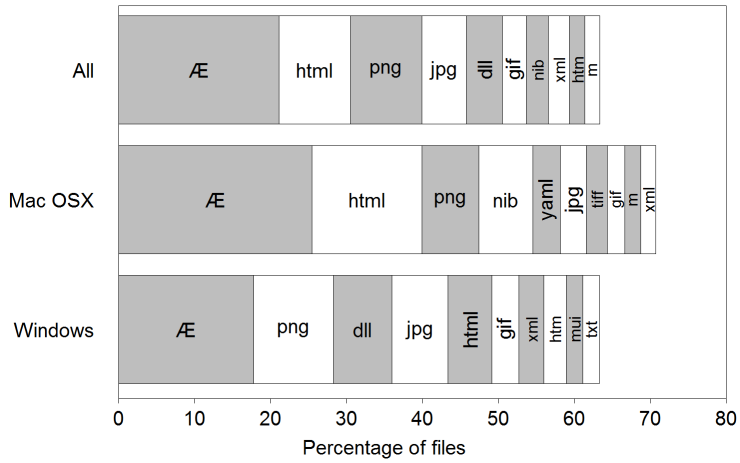


Figure 3.13: Percentage of FSOs within the ten most popular file name extensions.

Figure 3.14 shows, for the ten most popular file name extensions in terms of summed file size, the fraction of file bytes that reside in files using that extension. Files falling into the extension category Æ hold a high amount of the bytes in Windows (14%) and in Mac OSX (21%) FSs. As previously shown in Figure 3.6 in Chapter 3.1.2, multimedia files such as audio, video, and image files hold nearly 50% of the amount of bytes. Comparing the Windows results to the

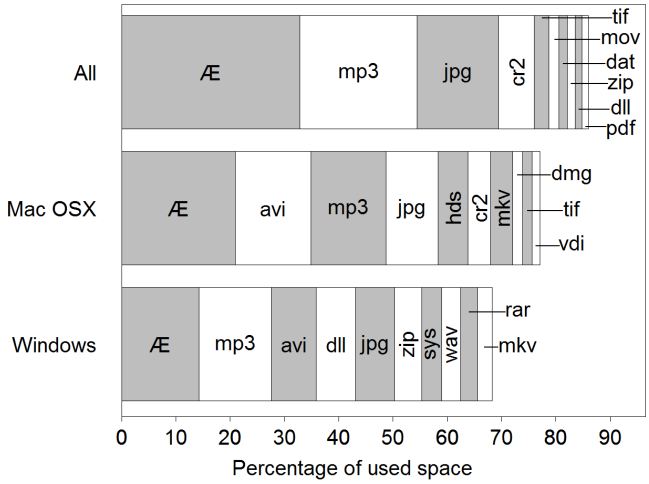


Figure 3.14: Space used by FSOs within the ten most popular file name extensions.

results of Agrawal et al. [6] from the year 2004, shows that the amount of files with a specific extension has changed. The results of the ten most popular file name extensions in terms of file count are presented in Table 3.5. Please note that there might be some discrepancy between the data because [6] scanned FSs of Windows employees and in the study at hand private FSs were scanned. File extensions that are not in the top list anymore are: *h* (moved to position 20), *exe* (moved to position 19), and *cpp* (moved to position 77). File name extensions that are still in both top lists are: Æ, *dll*, *jpg*, *gif*, *htm*, and *txt*. File extensions that are new to this study's top list are the extensions: *png*, *html*, *mui*, and *xml*. Portable Network Graphics (PNG) is a free image format which was developed in 1996 to replace and improve the patented Graphics Interchange Format (GIF) and in 2003 it became an international standard [38]. This could be the reason why *png* did not show up in the list of [6], but is now a very frequently used file name extension. HyperText Markup Language (HTML) is used for displaying information in web browsers. From the technical point of view there is no difference between *htm* and *html*, appart from the letter "l". The extension *htm* was introduced for Windows 3.x and DOS based computers, because they cannot handle files having extensions with more than three characters [3]. Nowadays there is no difference between both extensions, both denote that the file contains HTML code. This development and the fact that the usage of web based content has increased over the last years may be the reason why the amount of *htm* files has decreased and the amount of *html*

Table 3.5: Percentage of FSOs using the ten most popular file name extensions compared with results from previous study [6].

Extension	This study (2011)	Results from [6] (2004)	Trend (2004 - 2011)
Æ	18%	8%	+10%
png	10.5%	-	-
dll	7.5%	7%	+0.5%
jpg	7.4%	3%	+4.4%
html	5.8%	-	-
gif	3.6%	7%	-3.4%
xml	3.3%	-	-
htm	2.9%	4%	-1.1%
mui	2.2%	-	-
txt	2%	4%	-2%

files has made it to the top list. Multilingual User Interface (MUI) files are used in the Windows operating system since version Windows 2000 (released towards the end of 1999) for a technology that allows for the installation and usage of multiple interface languages on a single system [36]. Study [6] was performed between 2000 and 2004 and it is understandable that not all scanned FSs used this new technology back then. As all of the scanned Windows FSs from this study use the MUI technology it is obvious that the amount of *mui* files has increased. Extensible Markup Language (XML) files are used by applications and devices to store, transmit, and view data. It is frequently used in the internet but also in local desktop applications. Due to the rapid development and distribution of XML it is not surprising that it has moved into the top list.

There are also changes in the percentage of used bytes by file name extensions. The results of the ten most popular file name extensions in terms of space usage are presented in Table 3.6. File name extensions that are in the top list of [6] and this study are: Æ, *mp3,* and *dll*. Extensions that are new to this study's top list are: *jpg*, *avi*, *zip*, *sys*, *wav*, *rar*, and *mkv*. File extensions that are not in this study's top list anymore are: *pdp*, *exe*, *vhd*, *wma*, *lib*, *pst*, and *pch*. The results show that about 45% of all files, and more than 50% of the bytes belong to nine file name extensions. This information can be used to optimise FS performance as proposed by [17, 16], where decision trees are used to automatically learn how to classify files and predict the properties of new files.

Table 3.6: Percentage of space used by the ten most popular file name extensions compared with results from previous study [6].

Extension	This study (2011)	Results from [6] (2004)	Trend (2004 - 2011)
Æ	14.3%	3%	+11.3%
mp3	13.4%	3%	+10.4%
avi	8.2%	-	-
dll	7.3%	10%	-2.7%
jpg	7.1%	-	-
zip	5.1%	-	-
sys	3.7%	-	-
wav	3.5%	-	-
rar	3.1%	-	-
mkv	2.7%	-	-

3.2 Directory information

This section presents diverse observations and results for directory information such as directory count or tree depth. Some results are compared to results published in the study of Agrawal et al. [6].

3.2.1 Directory count per FS

Similar to the file count per FS results described in Chapter 3.1.1 the directory count is increasing from year to year. This is happening because people have more available storage space for saving data. Directories are used for organising information in a hierarchical manner. Since more information, which means more files, can be stored, more directories are used to arrange and organise the information. Figure 3.15 shows the arithmetic mean and median directory count per FS. The results for Windows, Mac OSX, and the sum of all FSs are shown. It shows that the directory count per FS has only marginally changed within the sample period. Figure 3.16 shows the directory count distribution per FS and also per used operating system (Windows XP, Windows Vista, Windows 7, and Mac OSX). This figure shows that the number of directories per FS remains relatively constant over the observation period but the amount of directories is different for each operating system. The results are similar to the ones presented in Chapter 3.1.1. Also Figure 3.17 shows that the directory

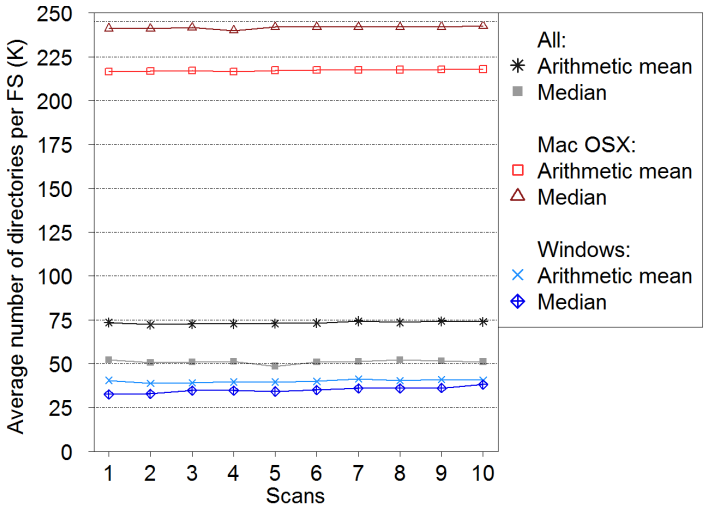


Figure 3.15: Mean and median number of directories per FS.

count is related to the used operating system, at least when looking at the observed Windows operating systems. FSs using an older operating system such as Windows XP contain less directories than FSs using a newer operating system such as Windows 7. A majority of the directories is located in the Windows folder. Since Windows 7 the amount of directories in "other" folders has increased. Similar to the file count results presented above, the number of directories is much higher for the observed Mac OSX operating systems than for Windows. Compared to Windows 7 it is more than four times higher. The results at hand deliver no explanation for this discrepancy. This incident is a potential topic for further analysis.

The results of Chapter 3.1.1 show that the file count of Windows files has decreased at the eighth scan because one of the scanned machines was completely reinstalled. No decrease of the mean and median directory count is visible in Figure 3.15, the values remain constant for all ten scans. There appears only a slight decrease of the directory count in Figure 3.16 for one of the Windows 7 FSs.

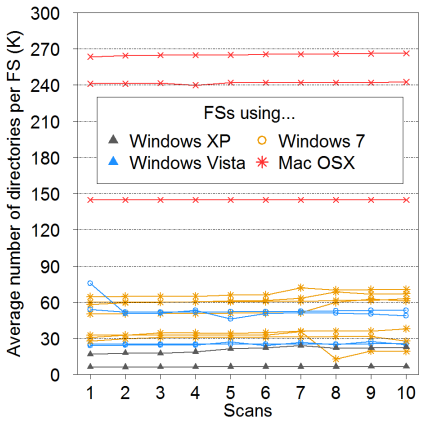


Figure 3.16: Directory count distribution per FS and per used operating system.

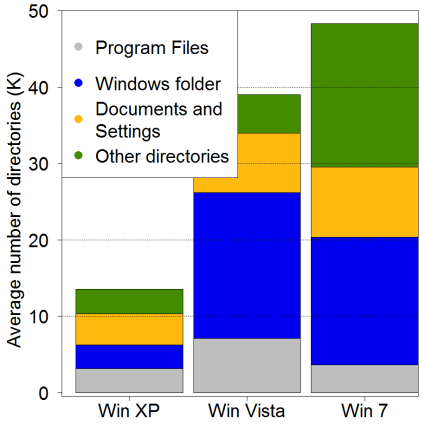


Figure 3.17: Average directory count distribution in the Windows special directories, grouped by used operating system.

3.2.2 Tree depth

In this section findings regarding the depth of FSOs in the FS tree are presented. The FS hierarchy, respectively the tree depth of FSOs in the FS is evaluated. In an FS, directories are used as virtual containers to group and organise files

and other directories, called sub-directories or sub-folders.

Figures 3.18 and 3.19 show the average amount of files respectively directories stored per tree depth. The file distribution in Figure 3.18 shows a maximum of files at tree depth four for Windows files and for the Mac OSX files at depth eight. The directory distribution presented in Figure 3.19 shows a different maximum at depth three for Windows FSs and at depth seven for Mac OSX FSs. The depth containing the maximum amount of directories is one depth before the depth containing the maximum amount of files. To examine the Windows peaks at depth three and four more accurately, the Windows results

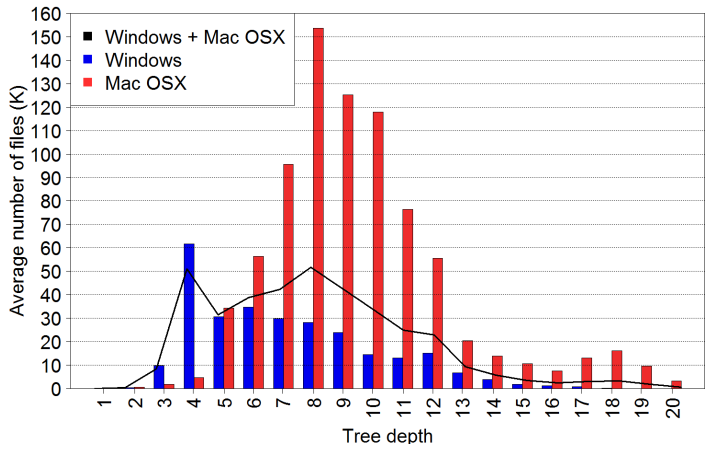


Figure 3.18: Average number of files stored per FS hierarchy level.

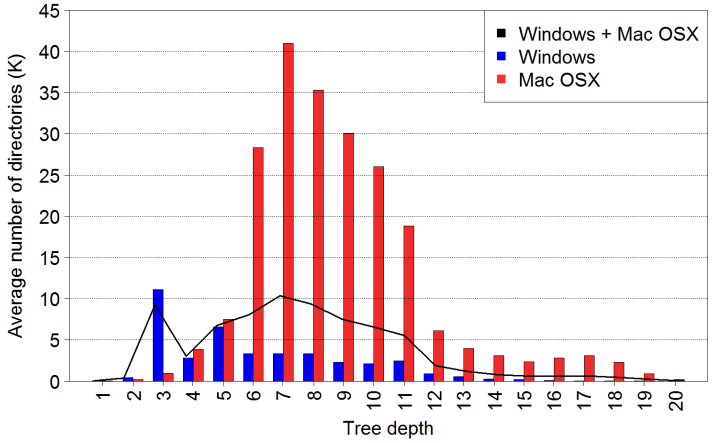


Figure 3.19: Average number of directories stored per FS hierarchy level.

were further analysed. Figures 3.20 and 3.21 present the mean count of files respectively directories per tree depth inside the Windows special directories. Both figures show that most of the files and directories in depth three and four reside in the Windows directory. This shows that a high amount of observed FSOs specific to Windows operating systems is located mainly in two tree depths.

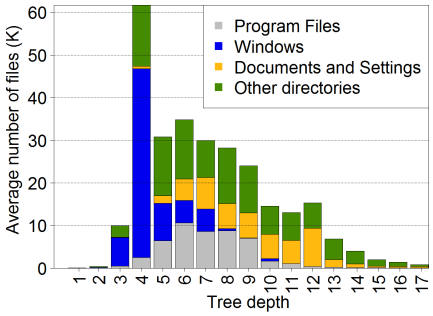


Figure 3.20: Average number of files in the Windows special directories stored per FS hierarchy level.

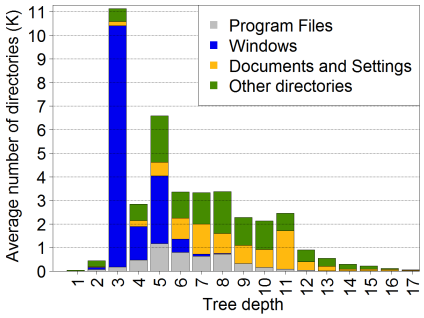


Figure 3.21: Average number of directories in the Windows special directories stored per FS hierarchy level.

As shown in Figure 3.22 nearly 40% of all observed Windows FSOs are located between tree depth one and five, around 45% of all Windows FSOs are stored between tree depth six and ten and only 15% are stored between

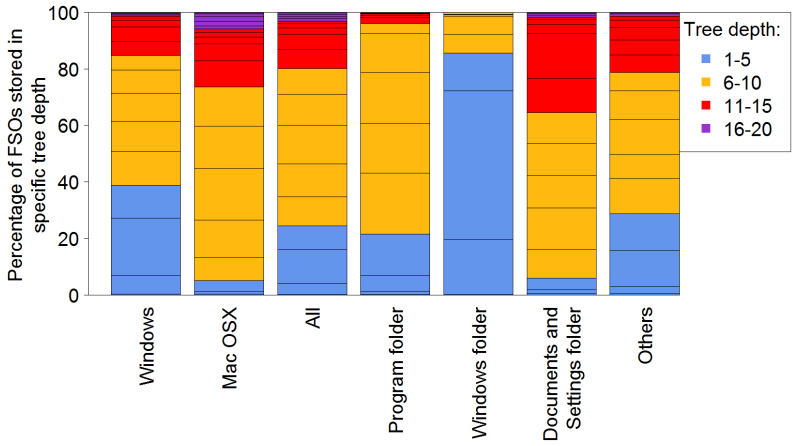
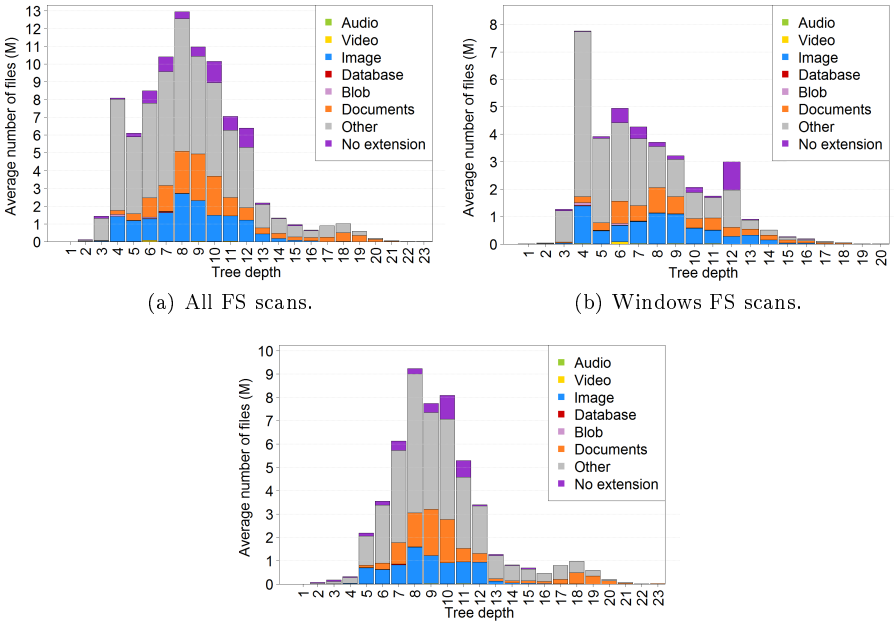


Figure 3.22: Percentage of FSOs stored per FS hierarchy level.

depth 11 and 20. Compared to Mac OSX, where nearly 70% of all FSOs are stored between tree depth six and ten, Windows has a flatter FS hierarchy. To take a closer look at the kind of data stored in the FS hierarchy the data has been broken up into different file type categories. The categories and the corresponding file extensions can be looked up in Table 2.2. Figure 3.23a shows the results for all FSs, Figure 3.23b for Windows FSs, and Figure 3.23c for Mac OSX FSs. The results for Windows FSs in Figure 3.23b show that document and image files are mainly stored between depth four and 14. Blob files are found in lower depths, mainly between depth four and six. And depth six contains a congregation of video files. The Mac OSX results shown in Figure 3.23c are slightly different. Most documents and images are stored between depth five and 12, but there is a second accumulation of documents deeper in the hierarchy between depth 16 and 20.



(a) All FS scans.

(b) Windows FS scans.

(c) Mac OSX FS scans.

Figure 3.23: Mean file count distribution of different file type categories for specific tree depths.

The total size of files stored per tree depth is shown in Figure 3.24. The figure shows a shift between Windows and Mac OSX. Most of the bytes from Mac OSX FSs are stored between depth five and nine, whereas Windows FSs

store most of the bytes between depth one and seven. In Windows FSs depth one contains for example the hiberfil.sys and pagefile.sys files which can have a file size of several GBytes. The Windows operating system uses the pagefile.sys file as temporary storage for the memory dump, and the hiberfil.sys file contains the whole content of the random access memory (RAM). The Windows results correspond to the results from Agrawal et al. [6] and show that files deeper in the FS hierarchy tend to have a smaller file size than the ones with a lower tree depth. This is also true for Mac OSX results, but the decreasing trend starts deeper at depth eight. Shallow depths between one and three contain a very small amount of bytes.

The distribution of bytes in the Windows special directories presented in Figure 3.25 shows again that Windows operating system specific files are accumulated in depth four. Depth four to 12 contains files stored in the Documents and Settings folder. There are no files in depth one or two since the Documents and Settings folder is located at depth two (general path for Windows XP is C:\Documents and Settings\, for Windows 7 it is C:\Users\). A lot of files of the category "other" are stored between depth one and seven. Since 12 of the 13 Windows participants use more than one HDD partition, these other files can be part of one of the additional partitions (such as university related files stored in D:\University\).

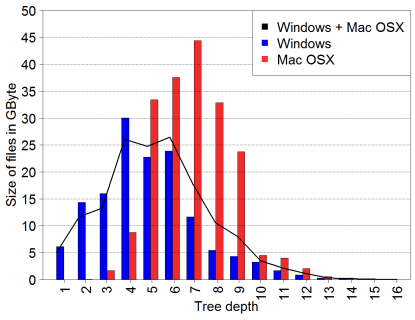


Figure 3.24: Mean size of bytes stored per FS hierarchy level measured for all FS scans.

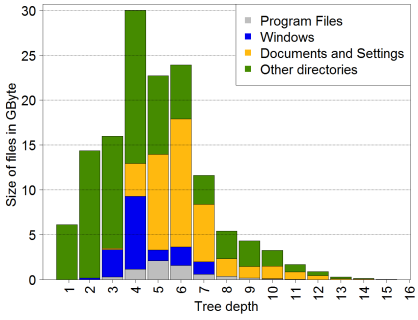


Figure 3.25: Mean size of bytes in the Windows special directories per FS hierarchy level.

The percentage of bytes stored in each depth is shown in Figure 3.26. It illustrates that more than 60% of bytes in the Windows FSs are stored between depth one and five. Around 70% of bytes in the Mac OSX FSs are found between

depth six and ten. This figure also confirms that mainly little bytes are stored deep in the FS hierarchy between depth 11 and 20.

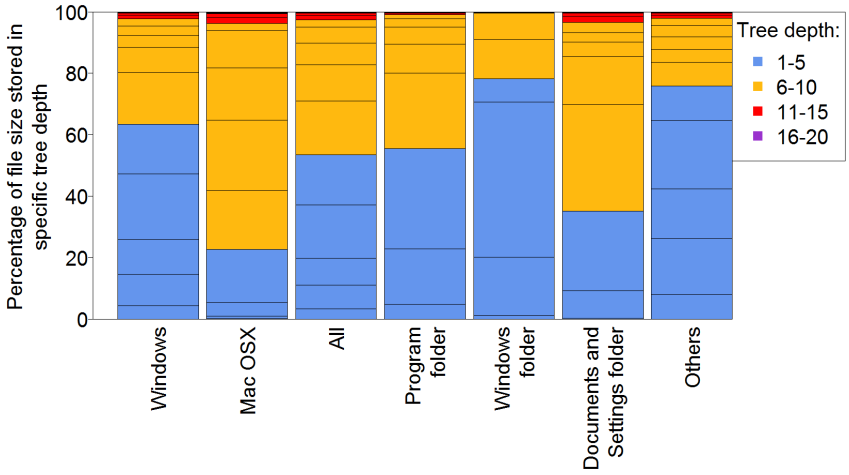


Figure 3.26: Percentage of bytes stored per FS hierarchy level.

3.3 Space usage

In this section results regarding storage capacity and space usage in the observed FSs are presented. As stated above the amount of files and directories as well as the file size is increasing. The following section shows how much storage capacity is available, used, and free, and how the values have changed compared to the results reported by Agrawal et al. [6].

3.3.1 Space capacity and usage

During every scan the scanning application detects and logs the total available, free, and usable space of the scanned FS. With this information it is possible to analyse if and how the space capacity and usage changes during the observation period. As already discussed in Chapter 3.1.1 and Chapter 3.2.1 the amount of available space for the observed FSs strongly depends on the used operating system. The assumption is that older operating systems, such as Windows XP, use older hardware with less storage capacity. Newer operating systems, such as Windows 7 use newer hardware with a higher storage capacity. Figure 3.27 shows that the total available disk space used per observed operating system also follows this assumption very well. With below 100 GByte on average

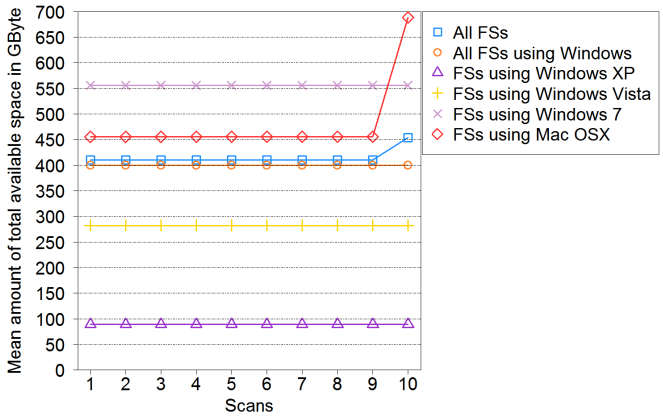


Figure 3.27: Total available space per operating system.

the oldest observed operating system Windows XP has the least amount of storage capacity available. Followed by Windows Vista with below 300 GByte on average and Windows 7 with more than 550 GByte available on average. During the observation period only one of the participants extended the available disc space. After the ninth scan one Mac OSX participant replaced the existing hard disc with a bigger one. Because of this, the average available storage capacity for Mac OSX increases from around 450 GByte to nearly 700 GByte. Since the observation period is too short no general conclusion about how often users extend their storage capacity can be made. It is also possible that some online storage services for storing, sharing or backing up files are used by the participants. These storage capacities were not detected by the scanning application. It might be interesting to find out how many and what kind of FSOs are stored online and how often these FSOs change. This is a potential topic for further analysis and studies.

Figure 3.28 shows the total free and total used space per operating system. The used space is slightly increasing whereas the free space is slightly decreasing for all operating systems. This means that less free space is available, which indicates that on the one hand every time we turn on and use the computer new data is created and stored. Temporary files are created when surfing the web, music and images are added to the library, new documents are created, etc. On the other hand people tend to not delete their data but to archive them somewhere in the FS. Furthermore, deleted files are moved to the recycle bin first and not removed from the hard disc, so the disc space is still used. The

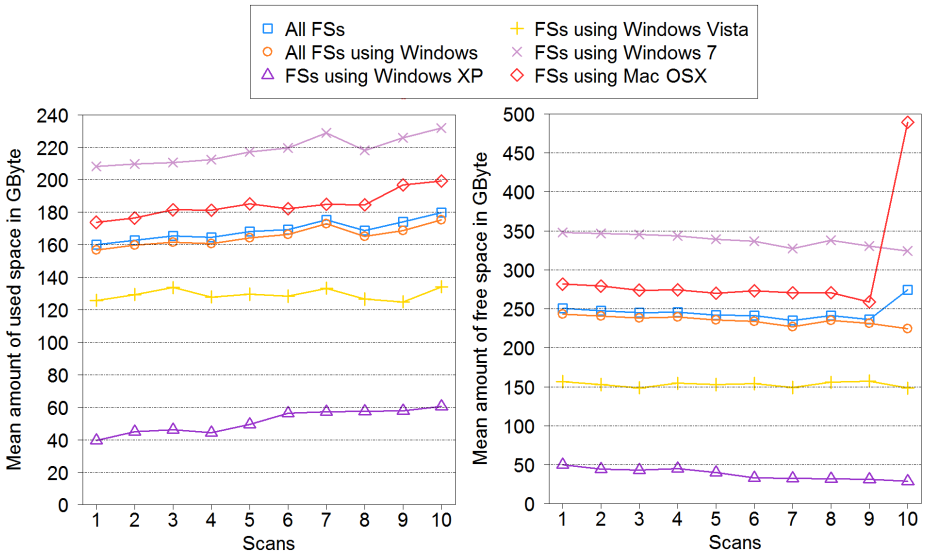


Figure 3.28: Total free and total used space per operating system.

free space increases for the Mac OSX FSs only because one of the participants replaced the HDD with a bigger one.

Table 3.7 presents some storage capacity statistics for the two scanned external HDDs. These results also show that the consumed space is increasing and the free space is decreasing. Both external HDDs are mainly used to back up data such as multimedia collections and important documents, and they are used by the operating system to store operating system related backups. During the ten week observation period the mean fullness of both external HDDs increased by 7%.

Table 3.8 presents the storage capacity, the total consumed space, and the free space of the Windows and Mac OSX FSs. Furthermore, additional statistics, such as the median, arithmetic and geometric mean have been calculated for the results of the first and last scan. The trend column shows how much these values have changed during the observation period. The table shows that the consumed space increases. Additionally, the free space decreases in Windows and increases in Mac OSX FSs since one of the Mac OSX participants replaced the HDD with a bigger one. In addition, the Windows results of the tenth scan are compared to the results reported by Agrawal et al. [6] in the year 2004. The storage capacity, the consumed space as well as the free space have increased compared to the results from [6]. The mean storage capacity has increased by 770% and

Table 3.7: Storage capacity statistics of observed external HDDs.

External hard drives	**1. Scan** (GByte)	**10. Scan** (GByte)	**Trend**
Storage capacity			
Median	466	466	0%
Arithmetic mean	466	466	0%
Geometric mean	466	466	0%
Total consumed space			
Median	234	264	+13%
Arithmetic mean	234	264	+13%
Geometric mean	201	242	+20%
Fullness			
Arithmetic mean	50%	57%	+7%
Free space			
Median	232	201	-13%
Arithmetic mean	232	201	-13%
Geometric mean	199	171	-14%

the mean total consumed space has increased by 872%. The mean percentage of fullness has decreased by 2%. This shows that, although the available disk space has increased a lot over the past years, the average percentage of fullness has not changed. On average more than 50% of the disks are free.

3.4 Temporal changes in the FS

As described in Chapter 2.1.4 an algorithm implemented by Popitsch [24] is used for detecting events that occurred between two consecutive scans. Information about each event detected between a scan pair including event type and FSO path is stored in a so called "compare result file". For each participant ten FS snapshots were gathered, which allows for the creation of nine compare result files. After all events for all scans are calculated, further analysis and calculations such as the event frequencies for each compared scan pair can be calculated.

3.4.1 Number of detected events

Figures 3.29, 3.30, 3.31, and 3.32 show the percentage of *move*, *remove*, *create,* and *update* events. All figures show an increase of detected events

Table 3.8: Storage capacity statistics of Windows and Mac OSX FSs. (*) The results of Agrawal et al. [6] (2004) are compared to the Windows results (10. Scan) of the study at hand.

	Windows			**Mac OSX**			**Agrawal et al. [6]**	
	1. Scan (GByte)	**10. Scan** (GByte)	**Trend** (%)	**1. Scan** (GByte)	**10. Scan** (GByte)	**Trend** (%)	**(2004)** (GByte)	**Trend (*)** (%)
Storage capacity								
Median	289	289	+0%	233	900	+286%	40	+623%
Arithmetic mean	400	400	+0%	455	688	+51%	46	+770%
Geometric mean	290	290	+0%	352	580	+65%	-	-
Total consumed space								
Median	107	141	+32%	196	207	+6%	13	+985%
Arithmetic mean	157	175	+11%	174	199	+14%	18	+872%
Geometric mean	116	130	+12%	126	146	+16%	9	+1344%
Fullness								
Arithmetic mean	39%	44%	+13%	38%	29%	-24%	45%	-2%
Free space								
Median	158	140	-11%	197	584	+196%	-	-
Arithmetic mean	243	225	-7%	282	489	+73%	-	-
Geometric mean	145	140	-3	90	425	+372	-	-

at the comparison of the seventh and eighth scan. As stated before, one of the participants had to completely restore his machine between the seventh and eighth scan and this causes the peak values. Detected *move*, *update*, and *remove* events reach a peak at the comparison of the seventh and eighth scan, the *create* event peak is reached at the comparison of the eighth and ninth scan. Generally, when restoring an FS, first all important data is backed up, for example on an external HDD. This explains the high *move* and *update* events, since already backed up files are updated and not yet backed up files are moved to the backup location. Then the FS is restored, meaning all FSOs are removed and the operating system and programmes are newly installed. Many files are removed from the FS which explains the high *remove* rate. Further, many files are newly created which increases the number of *create* events.

Figure 3.33 shows the total amount of events detected for the restored FS only. This figure shows the increase of *remove* events at the comparison of the seventh and eighth scan as well as the peak of *create* events at the comparison of the eighth and ninth scan. There is also an increase of the *move* and *update* events at the comparison of the seventh and eighth scan, but this increase is minimal compared to the one of the *remove* and *create* events. Figure 3.34 plots the percentage of FSOs which are detected as equal when comparing two consecutive scans. For around 96% of the FSOs detected between the first and seventh scan the compare algorithm could not detect an event and they were categorised as equal. Then the percentage of detected equal FSOs decreases due to the restoring of one of the FSs. These results show that most of the observed

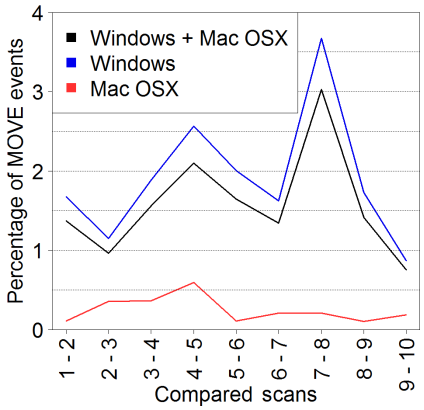


Figure 3.29: Percentage of *move* events detected in all FSs.

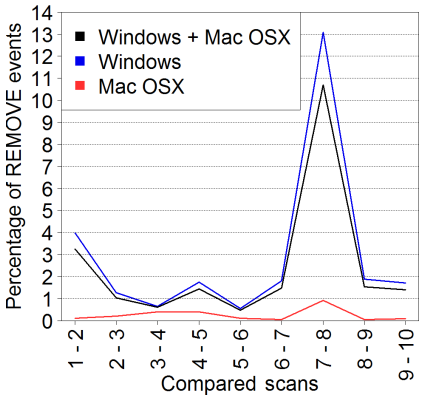


Figure 3.30: Percentage of *remove* events detected in all FSs.

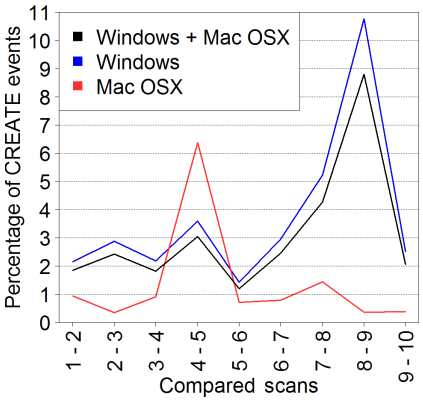


Figure 3.31: Percentage of *create* events detected in all FSs.

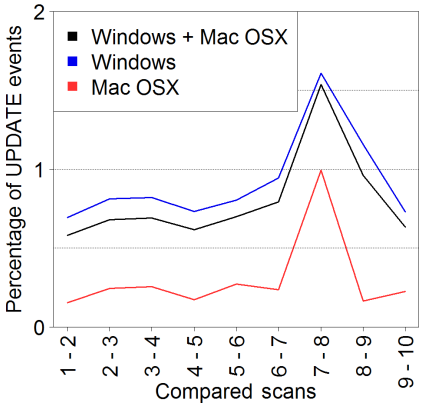


Figure 3.32: Percentage of *update* events detected in all FSs.

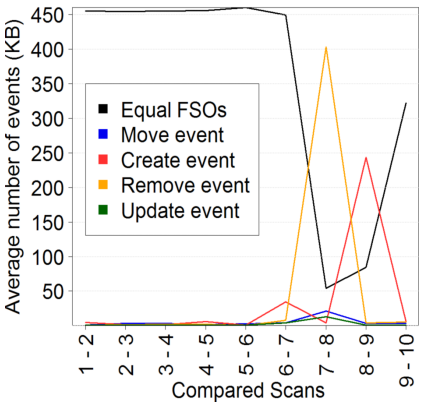


Figure 3.33: Number of events detected in the restored FS.

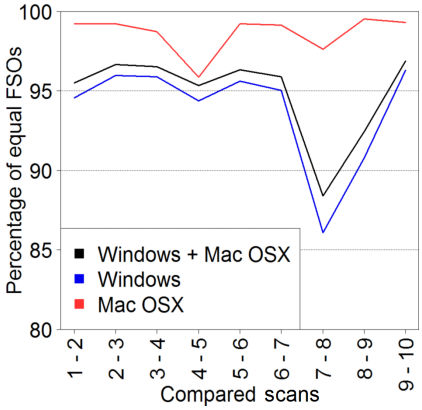


Figure 3.34: Percentage of equal FSOs detected in all FSs.

FSOs stay equal between two scans in case no major FS changes happen, such as restoring the whole FS. The overall results show that only a small fraction of FSOs is created (around 2%), removed (less than 2%), moved (less than 2%), or updated (less than 1%). Whereas the term "small fraction" still speaks of a relatively high number of FSOs, as on average 22,744 events are detected per scan comparison. When removing the events detected for the restored FS, 18,370 events are detected on average. These are 2,524 events per day. The minimum amount is 1,496 events, and the maximum is 441,410 events per FS scan comparison (184,783 when removing the results of the restored FS). This shows that the amount of occurring events can differ considerably. Different

user behaviour causes a different number of events. The installation or update of programmes or operating system related updates will cause a higher amount of events than surfing the web or writing an email. This shows that programmes such as anti-virus tools using real-time protection must be able to deal with different user behaviours and, respectively, different amounts of events. Figure 3.35 shows the percentage of events detected in Windows special directories. It shows that most of the *move* and *update* events happen in the Documents and Settings folder. Most of the new FSOs are created in directories falling under the category "other". *Remove* events are evenly distributed onto Windows, Documents and Settings, and "other" directories.

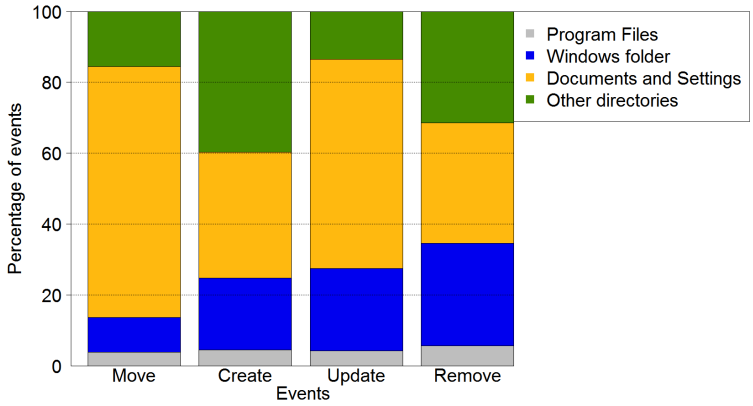


Figure 3.35: Detected events inside the special Windows directories.

It is also interesting to see by whom and how often an FSO was modified, but unfortunately this is not easy to detect. As shown in [20, 8] the NTFS change journal, which contains a list of every change made to files and directories in an FS using NTFS does not record any information about the entity (operating system, user, specific programme) which last modified an FSO. [18, 30] describe how the Windows auditing functionality and event logs can be used to determine the user which was logged on to the system while an event was detected. As the technique used in this study does not deliver accurate data analysing event logs on different operating systems is a potential topic for further analysis and studies. Since events can always get lost between two FS snapshots, real-time logging of FS regions must be used in order to detect all modifications of an FSO.

Figure 3.36 shows how many FSOs were modified more than once. The appearance of an FSO in all nine compare result files is counted. According to

the figure around 12% of modified FSOs are altered two times. Around 4% of modified FSOs on Windows and nearly 7% of modified FSOs on Mac OSX are detected in all nine compare result files. These files were at least modified once in each observed week. Results also show that on average more than 70% of the altered files are detected in only one of the observed weeks. It is possible that an FSO was modified several times between two FS snapshot creations, but only the last modification can be discovered by comparing the snapshot files. For example, a new text document is created, content is added to the text document, and by using the auto save function the document is saved, respectively updated, every five minutes. After the document is finished it is moved to a different location. All these events (*create* document, *update* document several times, and *move* document) are lost and only one *create* document event is detected.

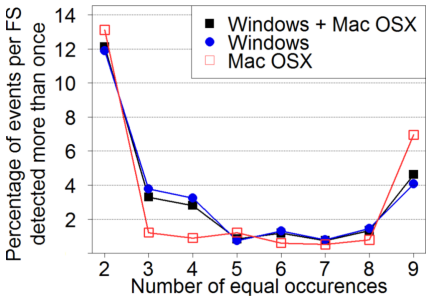


Figure 3.36: Percentage of FSOs modified more than once.

Figures 3.37, 3.38, 3.39, and 3.40 show tree-map visualizations of the event frequency in the different observed operating systems. The figures show how many events were detected in which regions of the FSs. Figure 3.37 shows the results for FSs using the operating system Windows XP, Figure 3.38 shows the results for Windows Vista, Figure 3.39 shows results for FS using Windows 7, and Figure 3.40 shows results for Mac OSX FSs. All Windows figures have in common that the majority of events is detected in the Program files folder, in the Windows folder, and in the Documents and Settings folder. These regions in the FS hold most of the detected events. Figure 3.37 further shows that two HDD partitions, C:\ and D:\, are used and events are detected in both of them. The Mac OSX tree-map shows that a majority of the events is detected in the user folders (USERS*Username*\ or PRIVATE\) and in application related folders (APPLICATION\).

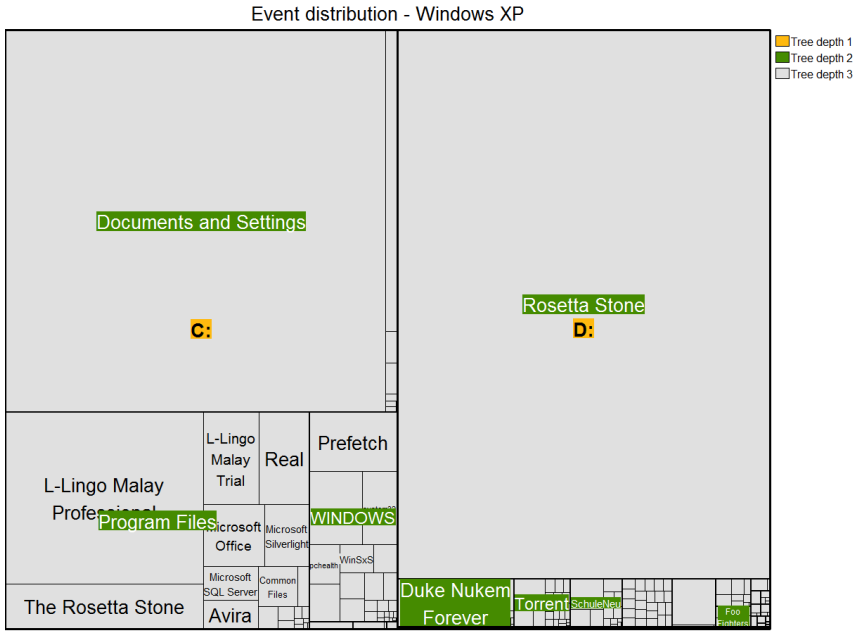


Figure 3.37: Tree-map of event frequency in observed Windows XP FSs.

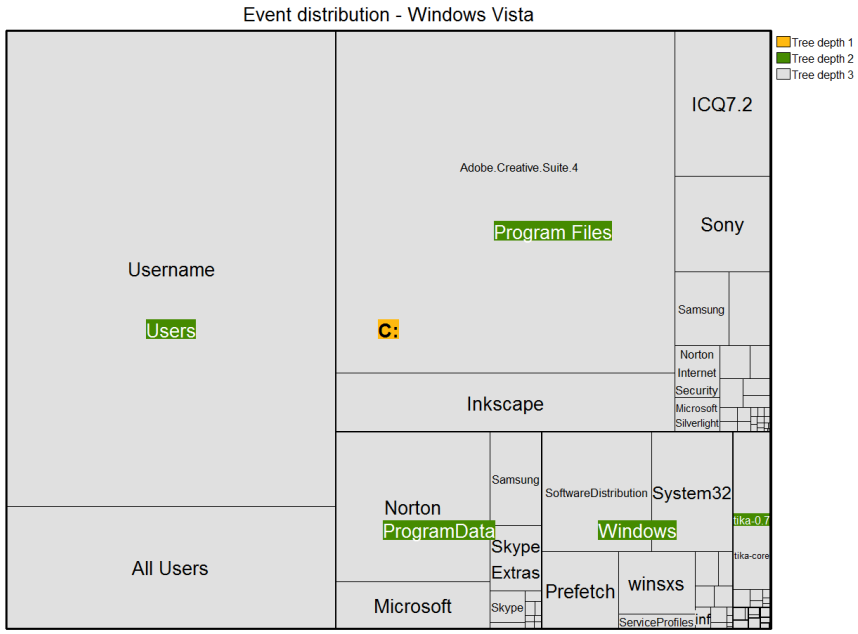


Figure 3.38: Tree-map of event frequency in observed Windows Vista FSs.

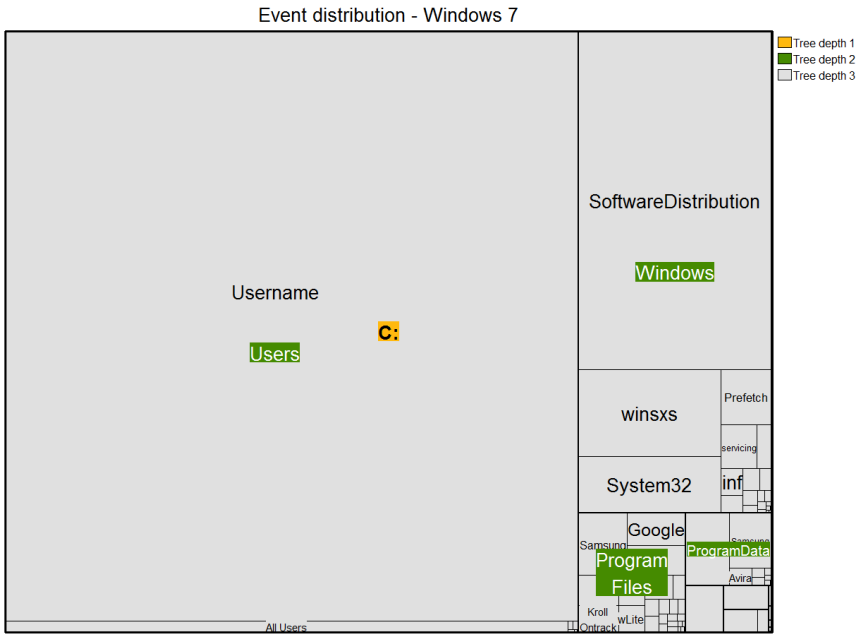


Figure 3.39: Tree-map of event frequency in observed Windows 7 FSs.

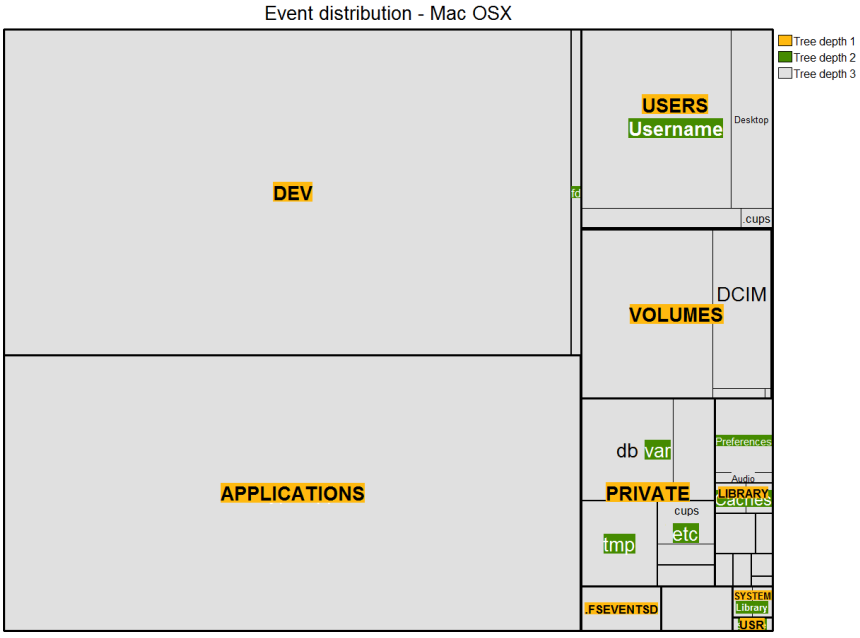


Figure 3.40: Tree-map of event frequency in observed Mac OSX FSs.

3.4.2 File name extension distribution

As described in Chapter 2.1.4 the result file containing all detected events also contains the path of the FSOs an event was detected for. This information is used to extract the file name extension and calculate the percentage distribution of file name extensions as described in Chapter 3.1.6. Figure 3.41 plots the percentage distribution for the ten most popular file name extensions in terms of file count. The typical usage of the most popular file name extensions found can be looked up in Table 3.4. The results are similar to the ones from Figure 3.13, which shows the ten most popular file name extensions in terms of file count for all found FSOs in the collected FS snapshots. The event distribution results show that a high amount of Windows events, around 30% of *create* and *remove* and more than 50% of *move* and *update* events, are performed for files with a file name extension falling under the category Æ, which means "no extension". Far less Mac OSX events fall under this category. File categories that are often changed are image files such as *jpg*, *png*, *svgz*, or *gif* files as well as *htm(l)* files. Furthermore, text files such as *txt*, *tmp*, *xml*, and *log* files are in the top ten list. The figure shows that between 40% to 60% of all events are performed on nine file name extensions only.

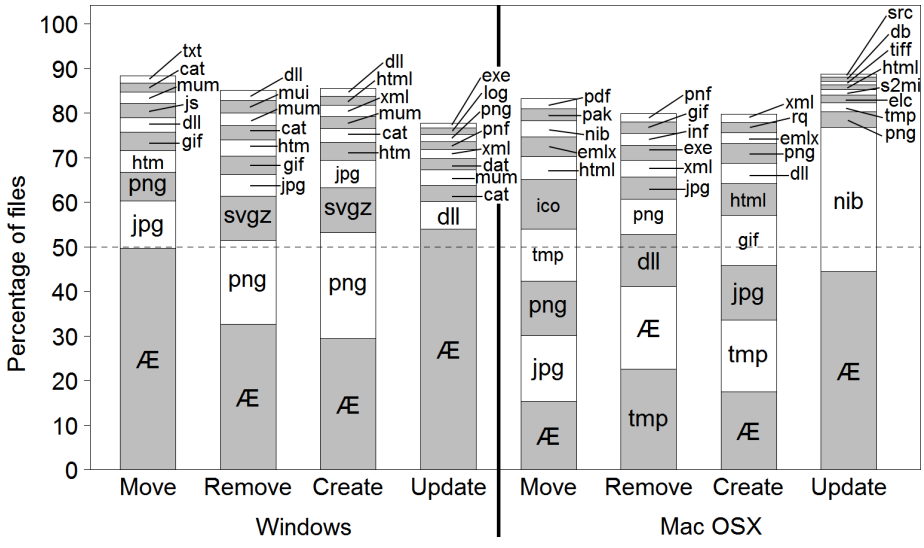


Figure 3.41: Percentage distribution of top ten file name extensions found for detected events.

3.4.3 File size distribution

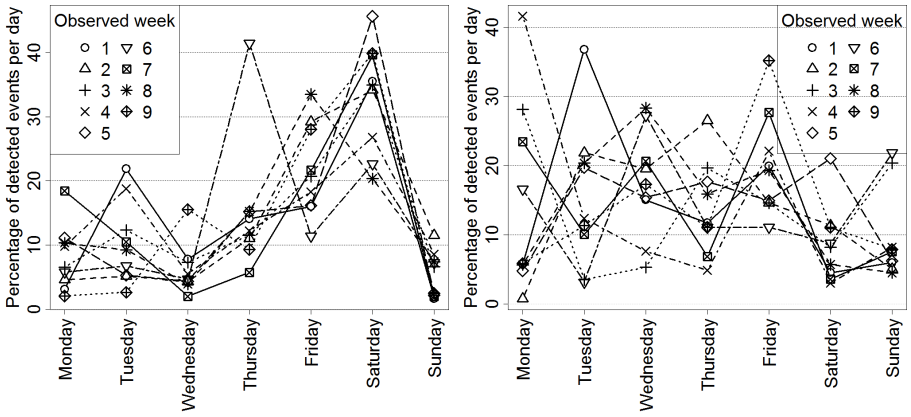
This section describes the file size distribution of changed FSOs. Table 3.9 presents the calculated mean and median file size of all modified FSOs per event type and observed operating system. According to the table, mainly small FSOs with a median size below 7.54 KByte in Windows FSs and below 6.48 KByte in Mac OSX FSs are created, removed, moved, or updated. The mean file size for events lies around 755.68 KByte for Windows FSOs and 1,783.16 KByte for Mac OSX FSOs.

Table 3.9: Mean and median file size per event type.

		All (KByte)	**Windows** (KByte)	**Mac OSX** (KByte)
Mean	All events	844.48	755.68	1,783.16
	create	580.98	507.53	1,458.45
	remove	566.47	585.45	286.31
	move	465.93	251.95	3,059.32
	update	3,703.2	4,039.66	2,470.91
Median	All events	7.52	7.54	6.48
	create	8	8	8
	remove	8	8.34	5.06
	move	6.46	6.18	18.57
	update	4	4.31	3.58

3.4.4 Modification date distribution

The following results show how many events were detected per weekday. The number of events per weekday is calculated for each participant. Figure 3.42 shows the distribution for one Windows (using Windows Vista) and one Mac OSX participant. The results of the Windows participant indicate a usage pattern with less detected modifications from Sunday to Wednesday and increasing modifications from Wednesday to Saturday. This pattern can be observed in each week in a very similar way. The results of the Mac OSX participant do not show such a clear pattern. Results look different for every observed week. Figure 3.43 plots the detected events per weekday for all observed FSs, Figure 3.44 for all observed Windows FSs, and Figure 3.45 for all observed Mac OSX FSs. The results for all and Windows FSs show that the fewest modifications are performed on Sundays. The Mac OSX results show a peak of events, mainly *update* events, on Fridays. As already discussed, events can get lost for example when an FSO is modified several times between two FS



(a) Windows participant. (b) Mac OSX participant.

Figure 3.42: Percentage of detected events per weekday for one Windows and one Mac OSX participant.

scans. Therefore the data of this study cannot deliver precise information about daily patterns. In order to find out if patterns exist, FS snapshots must be taken more frequently (e.g. hourly or daily) or real-time logging of FS regions must be used. This impreciseness of data is a potential topic for further analysis and studies, were FS data is collected more frequently and more precise analysis can be performed.

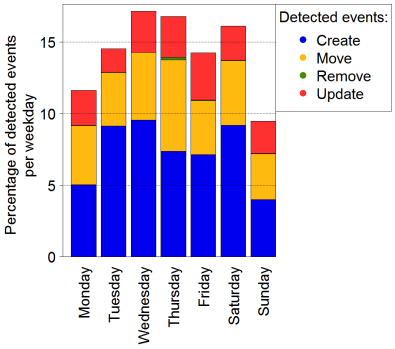


Figure 3.43: Percentage of detected events per weekday for all FSs.

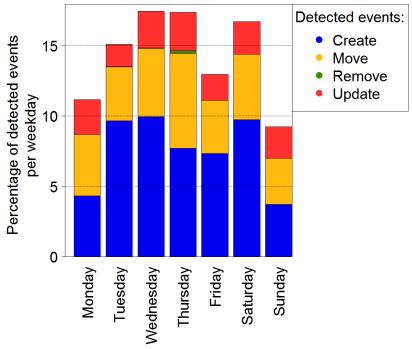


Figure 3.44: Percentage of detected events per weekday for Windows FSs.

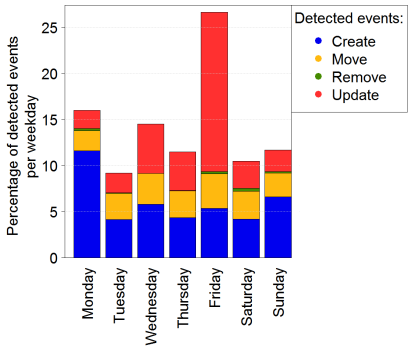


Figure 3.45: Percentage of detected events per weekday for Mac OSX FSs.

Chapter 4

Related work

This chapter presents articles and studies related to FS information gathering and analysis in order to give background information about the area of research.

In 1981, a study about file sizes and lifetimes was conducted and published by Satyanarayanan [26]. FS data from computers used by the Computer Science Department of the Carnegie-Mellon University was gathered and about 36,000 files were captured. File size, time-stamps, and file type according to the file-name extension were recorded. The main conclusions were that most files are very small, most files have a short functional lifetime (time between the most recent access and the most recent modification), and older files are used less frequently than younger files, with larger files tending to have a shorter functional lifetime than smaller files.

In 1984, Mullender and Tanenbaum [21] came to similar results as Satyanarayanan. In the study file size information from a UNIX system was captured. The distribution of file sizes closely matched that of Satyanarayanan's study.

In 2005, Mullender and Tanenbaum [31] repeated study [21] from 1984 and showed that the median file size had increased from 1 KByte to 2.4 KByte. Furthermore, 4 KByte was considered as the optimal block size for FSs as with this size almost 60% of all files fit into a single disk block.

In 1991, Bennett, Bauer, and Kinchlea [7] gathered FS data of three file servers of the Department of Computer Science at the University of Western Ontario. A working day long file information of more than 304,000 files were captured by making snapshots every 15 minutes. File size, time-stamps, and file type according to attribute flags were recorded. Compared to Satyanarayanan's

study of 1981, the size of mean text files has grown, the mean executable file size has decreased, and the mean number of files per user has increased. The study confirms that most files are small. A conclusion was that the overall number of files increases over time and that over 80% of files were last updated more than one month before the data was gathered.

In 1994, Smith and Seltzer [28] collected daily snapshots from 48 FSs on four file servers over a period of ten months. The study focused on file fragmentation and did not report any distributions for the collected file information.

Also in 1994, Sieknecht et al. [27] developed an application that collects static FS data from a commercial UNIX system at the Hewlett-Packard Company. Information from about 267 FSs and about 2,300,000 files was gathered. This study also came to the conclusion that most of the files were small files ranging from 10 KByte to 40 KByte.

There are studies from Douceur and Bolosky [9] and Agrawal, Bolosky, Douceur, and Lorch [6] where snapshots of FS meta data from Windows PCs were collected in the commercial environment of Microsoft Corporation.

In 1999, Douceur and Bolosky [9] took snapshots from more than 10,000 Windows FSs. This study focused on lateral variations among FSs at a specific moment of time.

Study [6] published in 2007 is a longitudinal extension of [9]. Over five years (from 2000 to 2004) snapshots of FS meta data from more than 60,000 Windows company PCs were collected on a voluntary basis. For only 18 FSs snapshots were created in each of the five years. The snapshots were used to study temporal changes in file size, file age, file-type frequency, directory size, namespace structure, FS population, storage capacity and consumption, and degree of file modification. Results of the longitudinal study showed that the space used in FSs had increased over the course of the study. This study also confirms the assertion that most of the files are small, with 4 KByte or smaller. The mean files size as well as the number of files had increased. Over 35% of files used one of eight file name extensions. The portion of FS content created or modified locally has decreased over time. The study showed that the directory size distribution has not notably changed over the years of the study. Most of the directories had two or fewer sub-directories. Moreover, the study showed that the FS capacity had increased dramatically in the course of five years. Limitations of the study are that the scans were anonymised due to privacy reasons. No access to the real file and directory names and paths were available. Furthermore, the data set is biased as only FSs from the Windows

company were analysed. The snapshots were taken one year apart and can therefore give only little information about change frequencies of FSOs.

In 2008, Hicks et al. [13] developed an application that collected the entire contents of 40 personal FSs. The study focused on FSs from mechanical engineers with an industrial and academic background to analyse how they manage their personal electronic files. This data collection was performed only once and the tool gathered different file and directory information such as name, size, level in the hierarchical file space, creation and last modification date, etc. Results of this study show that almost 95% of all analysed files used by engineers can be classified as text, spreadsheet, presentation, database, project, drawing, cad, code, simulation, application, image, audio, video, internet, data, and compressed files. The FS hierarchy includes an average of nine hierarchical levels with 42 sub-directories and 495 files per level. Whereas the third, fourth, and fifth levels of the hierarchy hold the majority of all files (53.5%). Common criteria used to name files are document title (75%), purpose or function (60%), project title (50%), and date (45%). Common criteria for naming directories are the purpose or function (85%), the name of the project (55%), and the date (20%).

In 2010, Rowe and Garfinkel [25] used a corpus of more than 1,000 drive images and over five million files to explore time-stamps associated with files in order to detect atypical time patterns. Significant daily and weekly patterns were found. Fourteen subsets of files based on their times, bursts of activity (one-time, periodic in the day, and periodic in the week), and those having specific equalities or inequalities between any two of creation, modification, and access times were identified. By using overall statistics fourteen kinds of drive usage such as a business operating primarily in the evening were identified.

In 2010, Xu et al. [41] used relational database management and tree-map visualization to analyse personal digital collections by focusing on showing their structure and properties. A group of staff members was studied and information about how they organised their FSs and their motivations to organise their collections over time were gathered. Results showed that personal information management practices change according to changes in projects and in the organization, and that the value of documents to the user can also change accordingly.

In 2004, Teerlink [32] proposed a method that uses visualization techniques to represent file statistics such as file size, creation date, last access date, last modification date, owner, and file type. Visual representations of information

are easier for humans to process than represented as text. The developed software helps computer forensics to increase the chance of locating criminal evidence. Enhanced tree-maps, square block diagrams, and flexible colouring schemes were used to produce custom representations of file attributes.

In 2011, Popitsch [24] identified and discussed several building blocks for the implementation of semantic data organization methods on the desktop. As part of this work a study of the real-world distributions of low-level file features is presented. 15 Windows and Mac OSX FSs were scanned and a feature vector was created for each found FSO, containing file and directory information such as name, size, last modification date, level in the FS hierarchy, and various others. This study also confirms the assertion that a majority of files are small with a mean file size of 351 KByte and a median file size of 2.3 KByte. As with other studies, the vast majority of detected FSOs was last modified longer than a month before the FS scan.

Chapter 5

Conclusion and outlook

5.1 Summary and review

In this work, techniques for observing and analysing FSs, and the data stored in those FSs are presented. Two methods for FS monitoring, (1) real-time logging of the FS changes and (2) the creation of FS snapshot files are described. The technique of creating FS snapshots is used to collect snapshots of 13 Windows and three Mac OSX FSs over a span of ten weeks. This results in 160 FS snapshots, containing more than 7×10^7 FSOs which are used for further research.

This study is motivated by the question of how FSs and the stored data change over time and if there are file types or regions in the FS that are potentially more dynamic than others. Results of FS events and last modification date distributions show that there are regions in the FS that are more dynamic than others, but in general a majority of the detected FSOs is rather static. The analysis of file name extensions found several file types which tend to be more dynamic than others. The following list summarises results answering this research question.

- Chapter 3.4 presents results regarding the dynamics of the observed FSs showing that around 96% of the FSOs remain unchanged between two FS scans. Only a small fraction of FSOs is created (around 2%), removed (less than 2%), moved (less than 2%), or updated (less than 1%). The visualization of event frequencies shows that many events detected in the observed Windows operating systems are located in the Windows

special directories (Program files, Windows, and Documents and Settings). In Mac OSX FSs the majority of events is detected in the user folders USERS\Username\ or PRIVATE\ and inside the APPLICATION\ folder. According to these measurements it can be assumed that there are regions in the Windows and Mac OSX FSs tending to be more dynamic than other regions.

- Further results show that a large fraction of FSOs stored in an FS stay untouched for longer time periods (weeks to months). Especially operating system related files in the Windows directory remain untouched for a long time. The mean interval between scan date and last modification date is 2.5 years for Windows FSs, 2 years for Mac OSX, and 2.3 years for all observed FSs. A median interval of 1.9 years is calculated for all FSs.

- The analysis of file name extensions of detected events shows that image files (*jpg*, *png*, *svgz*, and *gif* files), *html* files, and text files (*txt*, *tmp*, *xml*, and *log* files) are the most often modified file categories. Between 40% to 60% of the detected events are performed on these nine file name extensions only. These file types seem to be more dynamic than other file types such as audio or video files.

- Analysing the size of altered FSOs shows that mainly small FSOs are created, removed, moved, or updated. A median size below 10.3 KByte in Windows FSs and below 7.2 KByte in Mac OSX FSs is measured. The mean file size is below 1 MByte for detected events in observed Windows FSs and below 2 MByte for events detected in observed Mac OSX FSs.

- The distribution of altered FSOs detected during the observation period shows that on average more than 70% of the altered FSOs are detected only once, which means that they were modified in only one of the observed weeks. Around 12% of modified FSOs are detected two times and around 4% of modified FSOs on Windows and nearly 7% of modified FSOs on Mac OSX are detected in all nine compare result files, meaning they were modified at least once in each of the observed weeks.

Furthermore, this thesis investigates which fractions of multimedia files are stored in an FS and if these multimedia files are somehow clustered inside the FS. According to Chapter 3.1.1 one fourth of all detected files can be categorized as image (21.9%), audio (2.7%), or video (0.4%) files. Visualizations show that

image files are scattered all over the FS, while audio and video files are clustered in a few regions. In Windows FSs the special directory Documents and Settings contains an accumulation of image and audio files which indicates that users store their image and audio collections inside, using the Windows recommended folders "My Documents", "My Music", or "My Pictures". The multimedia file distribution for the observed Mac OSX FSs shows that a majority of audio and video files is stored in the USERS\ and LIBRARY\ sections. These results show that around one fourth of stored files are multimedia files and that at least the audio and video files tend to be clustered in the FS. Chapter 3 contains some further FS evaluations and results summarised in the following list.

- Results show that the amount of files and directories has increased over the last years as more storage capacity is available for storing data.
- Most of the files have a small file size, for Windows FSs a mean file size of 464.6 KByte and 196.7 KByte for Mac OSX FSs is measured. The median file size is also small with 5.2 KByte for Windows and 1.6 KByte for Mac OSX FSs.
- Several results show that there are differences between the observed Windows operating systems Window XP, Windows Vista, and Windows 7. File and directory count as well as file age are different in FSs using the different Windows operating systems. These differences arise not only through the use of various Windows operating systems but also by using different hardware. As older operating systems tend to use older hardware such as HDDs, less space is available for storing data. This results in smaller file and directory counts than on newer operating systems using newer HDDs with more available space. The file age is related to the time the operating system was installed. Files on newer operating systems are newer because in general they were set up more recently than older operating systems.
- The analysis of file and directory names shows that mainly short names are used. Around 40% of all FSO names use between zero and five characters. Moreover, the results show that more than 50% of the Windows FSO names contain letters or special characters.
- Tree depth distribution analysis shows that the tree depth of the observed FSs depends on the used operating system. Windows FSs tend to have a

flatter FS hierarchy, with a peak of FSOs at depth three and four, than Mac OSX FSs, with a peak of FSOs at depth seven and eight. The analysis of file size distributions in tree depths shows that files deeper in the FS hierarchy tend to have a smaller file size than files inside lower tree depths.

- Space capacity and usage statistics show that the available storage capacity has increased by around 770% compared to the results of Agrawal et al. [6] from the year 2004. The measured mean fullness of the observed FSs is below 50% which is equal to the results of Agrawal et al. [6], where 45% were measured. These results show that the available storage capacity has increased throughout the last years but people still keep about half of their available space free.

5.2 Future work

This study analysed a small data set of 16 FSs (13 Windows and three Mac OSX FSs) over a rather short observation period of ten weeks using the technique of creating FS snapshots. In order to get real-time and real-world data, a higher amount of various FSs using different operating systems must be observed over a longer period. In addition, more frequent observations would deliver more accurate data about FSs, their contents, and how they change. By creating FS snapshots more frequently the number of not detectable or lost events can be reduced, but using the technique of performing real-time logging of FS regions would provide most accurate data about FS dynamics. Since real-time logging is a very resource demanding technique it can currently be used for small FS regions only, but never for the whole FS. Furthermore, this study shows that it is not easy to detect by whom an FS event is triggered. This limitation could also be eliminated by using the real-time logging technique including an auditing function. Performing a study with event logs observing small FS regions containing operating system or user related content could help analyse how different entities modify, use, and organise data and could give a more detailed insight into FS dynamics.

Another limitation is that the scanning programme delivered only little statistical information about the scanned FSs. More information such as, when the operating system was installed the last time, how old the computer is, how many users use the computer, what is the principal purpose the computer is used for, etc., would help to get more detailed insights into FS usage patterns.

This study mainly focuses on the observed Windows FSs. Observing and analysing Mac OSX and Linux FSs in a more detailed manner would deliver more detailed insight into diverging FS usage behaviour. The results of this study showed some discrepancies between Windows and Mac OSX results such as file and directory counts, which can be analysed in future research.

Furthermore the study at hand analysed desktop computers and notebooks only. Due to the increasing popularity of mobile devices such as tablets and smartphones future observations should also cover these new devices. It would be interesting to find out the differences between FSs on desktop computers or notebooks and FSs on other mobile devices. Since tablets or smartphones generally have less storage capacity than desktop computers and are used for different purposes it is likely that the organisation and distribution of stored data will differ.

The usage of online storage services such as *dropbox*, *Google Drive*, *Microsoft Skydrive*, etc. is becoming more and more popular. Many data on mobile devices and local desktop environments is synchronised using these cloud storage services [33]. Future studies can focus on what kind of data is stored there, to what devices they are synchronised, and how often this data changes. In addition, it would be interesting to see if the usage of online storage services changes the data and usage of local FSs. It is, for example, possible that the music or image collections are moved to online storage services in order to be available and synchronised on different devices. Results from Chapter 3.1.1 and 3.2.1 show that the file and directory count is increasing constantly. Further analysis can try to find out if this trend still continues with the increasing usage of online storage services, or if the amount of files and directories stored in local FSs decreases.

Appendix A

Curriculum Vitae

CHRISTINA OCHSENHOFER

Student at University of Vienna, Faculty of Computer Science
Email: christina.ochsenhofer@hotmail.com

EDUCATION

2010 - 2013
Master studies at the University of Vienna in Media Informatics

2006 - 2010
Bachelor studies at the University of Vienna in Media Informatics

2001 - 2006
BORG Birkfeld, focusing on computer science

Bibliography

[1] Internet Assigned Numbers Authority (IANA) - mime media types. Available online at `http://www.iana.org/assignments/media-types`; visited on February 23rd 2013.

[2] *Dropbox* (file hosting service). Available online at `https://www.dropbox.com/`; visited on February 27th 2012.

[3] *File extenstion definitions* (from whatis.com). Available online at `http://whatis.techtarget.com/file-extension-list/A`; visited on February 27th 2012.

[4] *The Central File Extensions Registry.* Available online at `http://www.fileinfo.com/`; visited on February 27th 2012.

[5] *The R Project for Statistical Computing.* Available online at `http://www.r-project.org/`; visited on April 20th 2012.

[6] Nitin Agrawal, William J. Bolosky, John R. Douceur, and Jacob R. Lorch. A five-year study of file-system metadata. *Trans. Storage*, 3, October 2007.

[7] J. Michael Bennett, Michael A. Bauer, and David Kinchlea. Characteristics of files in NFS environments. *SIGSMALL/PC Notes*, 18(3-4):18–25, December 1992.

[8] L.F. Cabrera, B. Andrew, K. Peltonen, and N. Kusters. Advances in Windows NT storage management. *Computer*, 31(10):48–54, October 1998.

[9] John R. Douceur and William J. Bolosky. A large-scale study of file-system contents. *SIGMETRICS Perform. Eval. Rev.*, 27(1):59 – 70, May 1999.

[10] Kylie M. Evans and Geoffrey H. Kuenning. A study of irregularities in file-size distributions. In *International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 02)*, 2002.

[11] Tim Golden. *Tim Golden's Python Stuff: Watch a Directory for Changes.* Available online at `http://timgolden.me.uk/python/win32_how_do_i/watch_directory_for_changes.html`; visited on April 28th 2012.

[12] Greyware Automation Products, Inc. *System Change Log.* Available online at `http://www.greyware.com/software/systemchangelog/3x/index.asp`; visited on April 28th 2012.

[13] B. J. Hicks, A. Dong, R. Palmer, and H. C. Mcalpine. Organizing and managing personal electronic files: A mechanical engineer's perspective. *ACM Trans. Inf. Syst.*, 26(4):23:1–23:40, October 2008.

[14] VI Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet Physics-Doklady*, volume 10, 1966.

[15] Linux. *Linux man page - inotify.* Available online at `http://linux.die.net/man/7/inotify`; visited on April 28th 2012.

[16] M. Mesnier, E. Thereska, G.R. Ganger, D. Ellard, and M. Seltzer. Attribute-based prediction of file properties. *Harvard Computer Science Group Technical Report TR-14-03*, 2003.

[17] M. Mesnier, E. Thereska, G.R. Ganger, D. Ellard, and M. Seltzer. File classification in self-storage systems. In *Autonomic Computing, 2004. Proceedings. International Conference on*, pages 44 – 51, May 2004.

[18] Microsoft MSDN. *Auditing Security Events.* Available online at `http://msdn.microsoft.com/en-us/library/ms731669.aspx`; visited on March 6th 2013.

[19] Microsoft MSDN. *FileSystemWatcher Class.* Available online at `http://msdn.microsoft.com/en-us/library/x7t1d0ky.aspx`; visited on April 29th 2012.

[20] Microsoft MSDN. *Keeping an Eye on Your NTFS Drives: The Windows 2000 Change Journal Explained, September 1999.* Available online at `http://www.microsoft.com/msj/0999/journal/journal.aspx`; visited on March 6th 2013.

[21] Sape J. Mullender and Andrew S. Tanenbaum. Immediate files. *Software: Practice and Experience*, 14(4):365–368, 1984.

[22] Oracle. *Watching a Directory for Changes - The Java Tutorials.* Available online at `http://docs.oracle.com/javase/tutorial/essential/io/notification.html`; visited on April 28th 2012.

[23] Uwe Pachler. *Java library - jpathwatch.* Available online at `http://jpathwatch.wordpress.com/`; visited on April 28th 2012.

[24] Niko Popitsch. *Building Blocks for Semantic Data Organization on the Desktop.* PhD thesis, October 2011.

[25] N.C. Rowe and S.L. Garfinkel. Global analysis of drive file times. In *Systematic Approaches to Digital Forensic Engineering (SADFE), 2010 Fifth IEEE International Workshop on*, pages 97 –108, May 2010.

[26] M. Satyanarayanan. A study of file sizes and functional lifetimes. *SIGOPS Oper. Syst. Rev.*, 15(5):96–108, December 1981.

[27] Tracy F. Sienknecht, Richard J. Friedrich, Joseph J. Martinka, and Peter M. Friedenbach. The implications of distributed data in a commercial environment on the design of hierarchical storage management. *Performance Evaluation*, 20(1-3):3–25, 1994.

[28] Keith Smith, Margo Seltzer, et al. File layout and file system performance. Technical report, 1994.

[29] W.R. Stanek. *Microsoft Windows 2000: Administrator's Pocket Consultant.* It-Administrators Pocket Consultant. Microsoft GmbH, 2000.

[30] Microsoft Support. *How to view and manage event logs in Event Viewer in Windows XP.* Available online at `http://support.microsoft.com/kb/308427/en-us`; visited on March 6th 2013.

[31] Andrew S. Tanenbaum, Jorrit N. Herder, and Herbert Bos. File size distribution on UNIX systems: then and now. *SIGOPS Oper. Syst. Rev.*, 40(1):100–104, January 2006.

[32] Sheldon B. Teerlink. A graphical representation of file statistics for computer forensics. 2004.

[33] Wikipedia. *Cloud storage*. Available online at `http://en.wikipedia.org/wiki/Cloud_storage`; visited on March 6th 2013.

[34] Wikipedia. *Disk access time*. Available online at `http://en.wikipedia.org/wiki/Disk_access_time`; visited on December 26th 2012.

[35] Wikipedia. *Levenshtein distance*. Available online at `http://en.wikipedia.org/wiki/Levenshtein_distance`; visited on December 11th 2012.

[36] Wikipedia. *Multilingual User Interface*. Available online at `http://en.wikipedia.org/w/index.php?title=Multilingual_User_Interface`; visited on January 27th 2012.

[37] Wikipedia. *My Documents*. Available online at `http://en.wikipedia.org/wiki/Documents_and_Settings`; visited on February 23rd 2013.

[38] Wikipedia. *Portable Network Graphics*. Available online at `http://en.wikipedia.org/wiki/Portable_Network_Graphics`; visited on June 16th 2013.

[39] Wikipedia. *POSIX time*. Available online at `http://en.wikipedia.org/wiki/POSIX_time`; visited on February 26th 2013.

[40] Wikipedia. *Special Folders*. Available online at `http://en.wikipedia.org/wiki/Special_Folders`; visited on February 23rd 2013.

[41] Weijia Xu, Maria Esteva, and Suyog Dott Jain. Visualizing personal digital collections. In *Proceedings of the 10th annual joint conference on Digital libraries*, JCDL '10, pages 169–172, New York, NY, USA, 2010. ACM.