AN ABSTRACT OF THE THESIS OF

John Batch for the degree of Honors Baccalaureate of Science in Electrical and Computer Engineering and Honors Baccalaureate of Science in Computer Science presented on May 20, 2008. Title: DiskGrapher: A Different Approach to Hard Drive Visualization for Mac OS X.

Abstract approved:

_____

Timothy A. Budd

DiskGrapher is a graphical visualization tool designed to help users better manage the space on their hard drives. The main goal of DiskGrapher is to provide a different visualization technique to display information, with the goal of providing a more intuitive understanding of the directory structure of the disk than the Treemap visualization technique, commonly found on hard drive visualization applications. By providing the user with better information about the directory structure of the disk, and presenting the information in a manner which the user is more likely to be familiar with, the program will help users to better manage used disk space. This thesis describes the features of many of the competing programs in hard drive visualization, as well as giving a detailed look at the structure and development of the DiskGrapher.

Key Words: DiskGrapher, Disk Management, Visualization, Computer Science, Computer Engineering

Email Address: jsbatch@lifetime.oregonstate.edu

DiskGrapher: A Different Approach to Hard Drive Visualization for Mac OS X

By

John Batch

A PROJECT

Submitted to

Oregon State University

University Honors College

In partial fulfillment of
the requirement for the
degree of

Honors Baccalaureate of Science in Electrical and Computer Engineering

Honors Baccalaureate of Science in Computer Science

Presented May 20, 2008

Commencement June 2008

Honors Baccalaureate of Science in Electrical and Computer Engineering and Honors Baccalaureate of Computer Science project of John Batch presented on May 20, 2008.

APPROVED:

_____

Timothy A. Budd, representing Computer Science

_____

Christine Wallace, representing Computer Science

_____

Mike Bailey, representing Computer Science

_____

Terri Fiez, Department of Electrical Engineering and Computer Science

_____

Daniel J. Arp, University Honors College

I understand that my project will become part of the permanent collection of Oregon State University, University Honors College. My signature below authorizes release of my project to any reader upon request

_____

John Batch, Author

Table of Contents

Table of Contents (cont)

Table of Figures

I. Introduction

The external hard disk storage available on computers has grown a great deal over the past 25 years, growing from a few kilobytes in size to hundreds of gigabytes. As hard drives grow, the amount of data that can be stored on a drive increases, but basic hard drive management has changed little.

Traditionally, operating systems provide a hierarchical structure that presents an increasing number of levels of files, nested within directories. This method of storing and representing files makes sense on a storage level, and provides an easy way for a user to organize files in a way that makes the most sense to them. However, when the disk becomes full, this traditional method of disk management gives the user little information to determine what is using the disk space.

Graphical visualizations of data are helpful to people when interpreting data. Because graphics can be such a powerful means of helping people understand a data set, offering a graphical representation of the data on a hard drive will help people to better understand where the majority of the disk space is being used. Representing files that take up a larger portion of the hard disk with larger shapes on the screen helps the user to quickly identify items that take up large amounts of space, allowing users to better manage the space available on the disk.

Project Motivations

There were several motivations behind choosing to pursue this project. The first of these motivations was the desire for a visualization tool for Mac OS X that did not use the Treemap style of visualization. While the Treemap style is useful for presenting a large

amount of information about a tree in a limited amount of space, it is difficult to understand at first, and does not help to find large directories.

The second major motivation behind this project was to learn Objective-C and the Cocoa API Frameworks. Having used an Apple computer for years, I was interested in learning how to program for the Mac OS X operating system, and using the Cocoa frameworks that is used by many of the native applications. DiskGrapher was an idea that would allow me to learn a new programming language and a set of APIs, while at the same time creating a tool that would be useful to others, as well as myself.

Project Goals

The major goal of this project is to create a tool for Mac OS X to help users visually see and understand the contents of their hard disk, and what is using the majority of the disk space. While many programs are available to visually display the contents of a hard disk, this tool provides a visualization technique different from the others available for Mac OS X. The goal of this visualization technique is to maintain the directory structure of the data, while still providing the user with a useful visual representation of disk space. By maintaining the basic structure of the disk, the tool offers the user a visualization that provides a great deal of information in a quick glance, without needing to dig through the visual display to discover what it means.

Outline

The rest of the paper describes DiskGrapher and the method in which it was developed. Section 2 of the paper provides an overview of the competing programs available for Mac OS X, and gives a general idea of their weaknesses and how Diskgrapher can

improve upon them. A description of the visualization technique used in DiskGrapher is presented in section 3 of the paper, giving the reader an idea of how the visualization is accomplished, as well as what the motivations are behind using this technique. The general requirements used while writing this program are described in section 4, with a description of the architecture used to implement the system in section 5. Finally, because this program, like many programs, is a continually evolving piece of work, section 6 outlines some goals for future improvements in the project.

II. Comparison with Existing Programs

There are currently several tools available that allow users to visualize the contents of their hard disk. Because of this, it is important that DiskGrapher, as a new tool in an existing market, offer some features and improvements that will set it apart from other tools. This section looks at some of the products currently available for Mac OS X, in order to help determine what users currently expect out of similar tools, and also to help determine some of the shortcomings of the existing tools. Analyzing competing tools can offer some idea of how to improve the user experience when creating this kind of disk utility, helping to make the tool more useful to the user.

Baseline

Overview

Baseline is a program that costs twenty dollars and allows the user to view the contents of the hard drive in several ways [1]. According to the developer, the program offers the following features:

- Superfast Disk Scanning
- View files and folders in a list, in columns, or graphically as a Treemap
- Compare your disk against saved Baselines
- View only the items that have changed
- Delete, Compress, or Archive items to save space

Observations

In testing, the program offers three views, two of which are similar to what is provided by the operating system. The third view, a Treemap view, provides a graphical representation of the hard drive, representing all the files with boxes of differing sizes. The program also provides the filename and size when you move the mouse over the corresponding box. For more information on the Treemap view, see Appendix A.

The speed of the program is one of the advertised features, and I found that the program was able to scan my entire drive in about two minutes. This is a reasonable speed, given the amount of data on a disk drive. The program provides a progress bar informing the user of how the progress is coming.

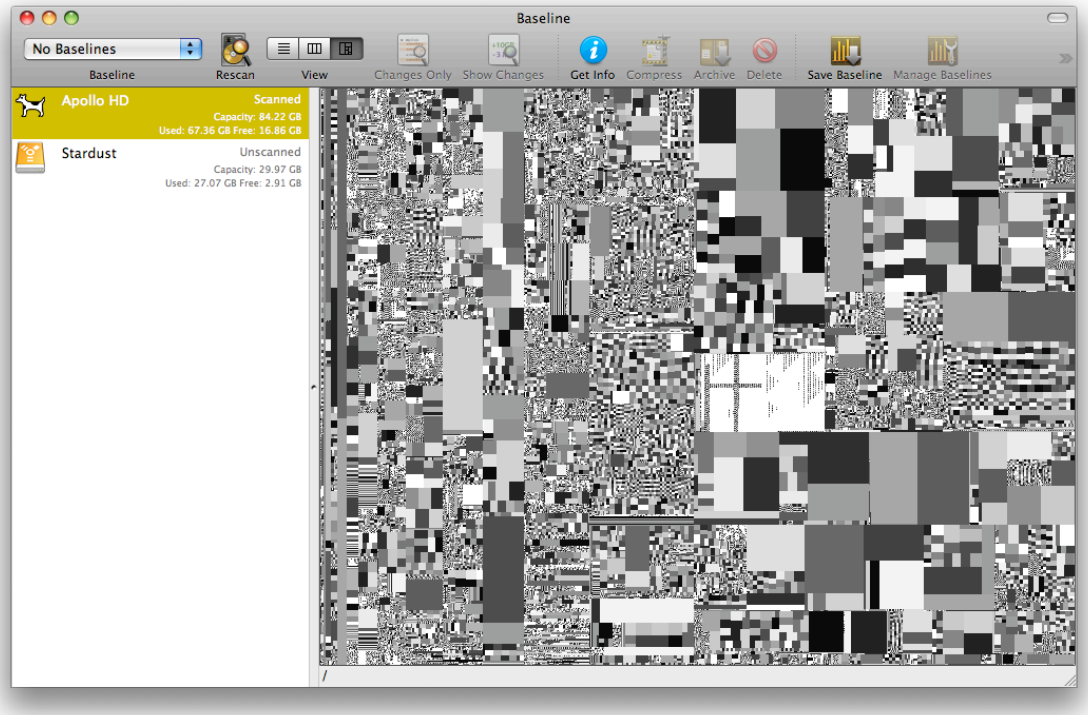

Figure 1: Baseline Main Windows with Completed Scan

Key Differences

While Baseline provides the user with many of the same features that DiskGrapher provides, it also offers a different set of features. The biggest difference between the programs is how the data is presented to the user. DiskGrapher presents the information in a structure similar to the layout of the hard disk. Baseline, like many other visualization programs, presents the information in a Treemap structure.

Baseline also provides the user with the ability to save a scan of the disk and open this later. This feature allows the user to scan a disk at a later time, and Baseline will compare the scan to the previous scans of the directory. The user can then look at the differences between the two scans to help determine what has changed on a disk since the previous scan. Seeing the difference between two scans is a feature that can be very helpful in figuring out what files have been added to a disk to take up disk space.

DiskGrapher also lacks the feature of Baseline that allows the user to see the hard drive in list and column view modes. Because Baseline allows the user to compare the current state of the hard drive against the previous state of the drive, these views offer more information than the regular file system view provided by the operating system. However, DiskGrapher does not have the ability to compare the disk to previous scans, so these added views would provide little additional information on top of what the operating system provides if they were implemented.

Filelight

Overview

Filelight is a graphical hard drive visualization program similar to DiskGrapher. Originating as a tool for Linux, Filelight is currently under development for Mac OS X [2]. Because the Mac OS X version of this tool is still in early development, and is available only as an unstable Alpha release at this time, many of the features of the tool do not work properly, and are sometimes missing entirely.

The visualization technique used by Filelight is similar in principle to that used by DiskGrapher. Both tools maintain the directory structure of the hard drive in their graphical representation of the hierarchy. Filelight, however, uses circles to represent the hard drive, creating an "interactive map of concentric segmented rings that help visualize disk usage on your computer" [3].

Observations

Filelight still needs a great deal of polishing, due largely to the early development phase of the project. The visualization technique itself, however, is very helpful for determining the location of large files and folders.

Unlike the Treemap visualization structure, Filelight's implementation not only allows the user to see what files are utilizing the largest portion of the disk or folder, but also provides the user with information on which directories are using the most space. This can be beneficial in finding directories using a large portion of the disk, but containing a large number of relatively small files.

Filelight has several major drawbacks, some of which will likely be fixed as development work on the program continues. The first major problem encountered was the inability to scan the entire boot drive. No indication of an error is provided; the program shows no progress indicator, and does not use the resources it typically uses during a scan.

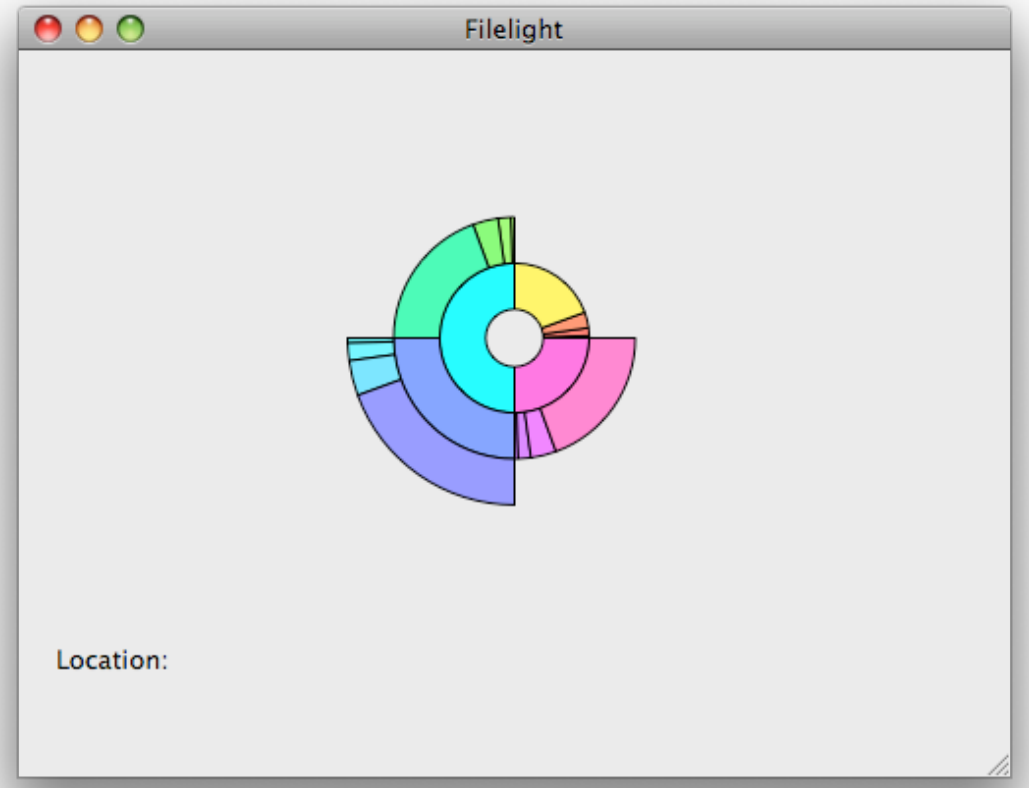

Figure 2: Filelight Main Window with a Folder Scan

The second problem with this visualization scheme is the limited number of directory levels that are presented to the user. When scanning a directory, the visualization only displays five levels to the user. This makes it difficult to determine if the outermost level of the display is representing a file or a directory.

The final problem that makes this program difficult to use is the color scheme used to represent files and folders. Filelight uses a variety of color, but it seems like these colors are assigned randomly, giving the user no indication of which blocks are files and which are folders. This, coupled with the five level limitation on the visual display, makes the visualization difficult to interpret.

Key Differences

Filelight uses a visualization technique that has some similarities to Diskgrapher, but also has key differences. Filelight presents the files and folders in the same hierarchy in which they are found on the hard drive. These files and folders are represented as circles expanding outwards from the center, which represents the root folder. The circular visualization resembles a pie graph, a style of visualization used often in scientific presentations. This technique is similar to DiskGrapher's approach, as the directory structure is maintained. However, the file system is often represented as a tree expanding to the right, and the circular representation of Filelight is somewhat counterintuitive when compared to this representation.

GrandPerspective

Overview

GrandPerspective is a small utility that provides the user with a Treemap visualization scheme similar to that provided by Baseline. GrandPerspective offers a smaller set of features, but it is also offered to the user free of charge. According to the developer, "each file is shown as a rectangle with an area proportional to the file's size" [4]. One feature that

sets it above Baseline is the ability to scan individual folders on a disk, rather than requiring the user to scan the entire drive.

Observations

GrandPerspective offers several features that make it very useful. In order to help users better visualize the disk directory, the program offers the user the ability to change the color scheme used, as well as how this color scheme is applied. The program offers several pre-defined color schemes, and different ways to apply the colors, which help the user to see different aspects of the visualization, such as how much of the disk is used by a certain type of file. One of the nice features is the ability to change how the color scheme is applied, whether it is the same color for each level in the structure or different colors for every file within a directory.

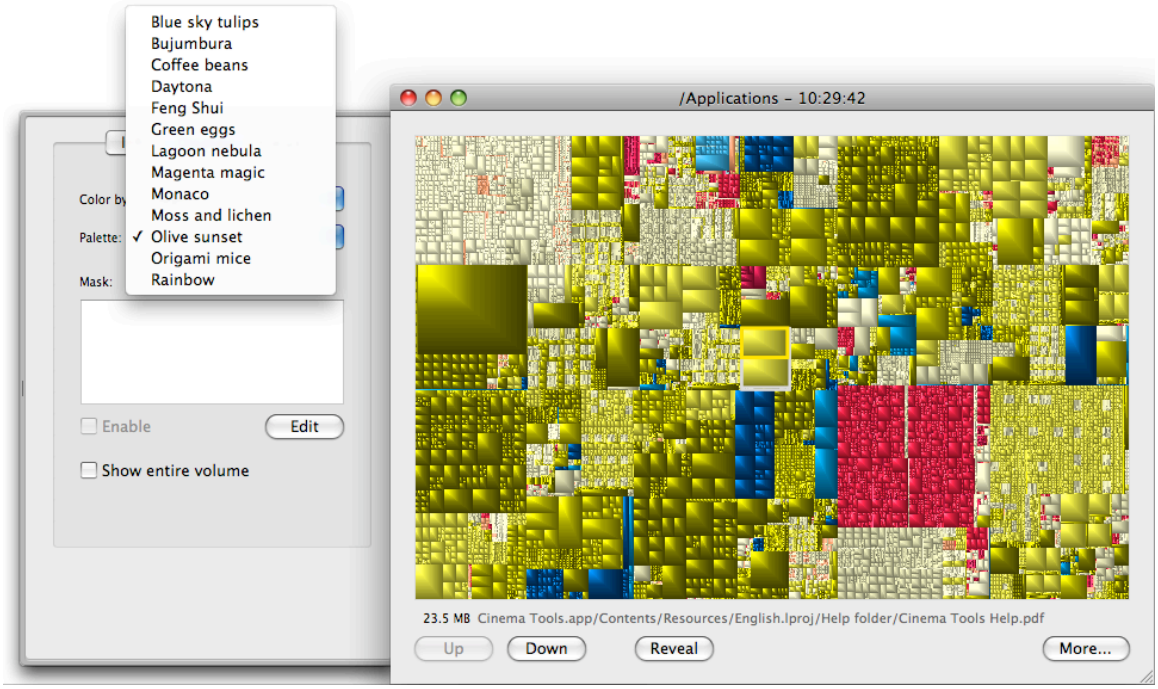

Figure 3: GrandPerspective Window showing available color schemes

Another feature which helps the user further visually investigate the disk utilization is the ability to move through the directory tree. By selecting a file or directory that is of interest to the user, a user can then select the "down" button to further investigate what is using the space in that directory. This allows users to investigate folders that use a large portion of the disk, but contain a large number of small files that are difficult to see in the full view.

Key Differences

GrandPerspective and DiskGrapher offer the user many of the same features. Both programs offer the user the ability to scan folders and disks to determine what is using space. The key difference that separates the two programs is the style in which they display the information. The Treemap visualization format used by GrandPerspective focuses the user's attention mainly on large files. The focus of DiskGrapher's visualization method is on finding areas of the disk that use large amounts of space.

III. DiskGrapher's Visualization Technique

DiskGrapher aims to provide the user with a visualization technique that helps them better manage their hard drive. What separates this program from the many other programs available is the method in which the data is displayed. This section provides a basic overview of the visualization technique used by DiskGrapher.

Users are offered a great deal of flexibility in how they arrange files within the directory structure of the hard drive, and many users take advantage of this to arrange files in a way which makes the most sense to them. Because of this, it does not make sense to completely ignore this organization and present the information in a manner organized completely different from what the user expects. DiskGrapher aims to maintain the directory structure of the disk drive or folder that is being scanned. Maintaining the directory structure of the hard drive in the visualization makes it easier for users to quickly tell what files are out of place, which increases the usefulness of the tool.

Visualization Details

The visualization layout is an important part of this project, as this is what distinguishes this program from other similar tools. The program uses the entire height of the window to represent the size of the root directory. The horizontal window space is divided to represent different levels of the directory structure.

As the disk is scanned, each file and folder is assigned a size, and this size is used to determine the percentage of the parent folder's size that this file or folder uses. This percentage is then used to represent how much vertical space this new directory item will use on the screen. For example, if a folder were to contain two items which took the same

amount of disk space, the program would show the parent folder as a rectangular block, and immediately to the right of this block, two blocks representing the files, equal in size, would be displayed. This example is shown below in Figure 4.

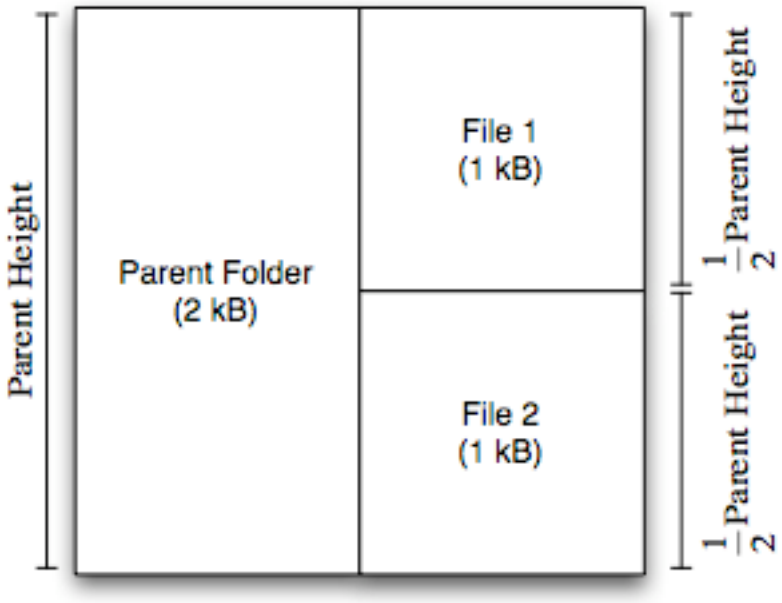

Figure 4: Simple Visual Example

As more subfolders are added to the main parent folder, they are displayed similarly. Each of the files and folders found in the subdirectory are also assigned a vertical size, and each is placed to the right of its parent directory. This process is continued until the entire depth of the tree has been displayed, with increasingly small file sizes being represented with increasingly small vertical sizes. An example of a more complicated directory structure is shown in Figure 5.

Visualization Advantages

Visualization is used in many industries to help the understanding of large sets of data. But what makes visualization valuable is the different ways in which data can be

constructed into a meaningful pattern. One method of visualization may be ideal for one person, but for another person, a different visualization technique may be more helpful. Because different visualization techniques exist, each with different strengths and weaknesses, a variety of programs exist to provide users with the graphical representation that appeals most to them.

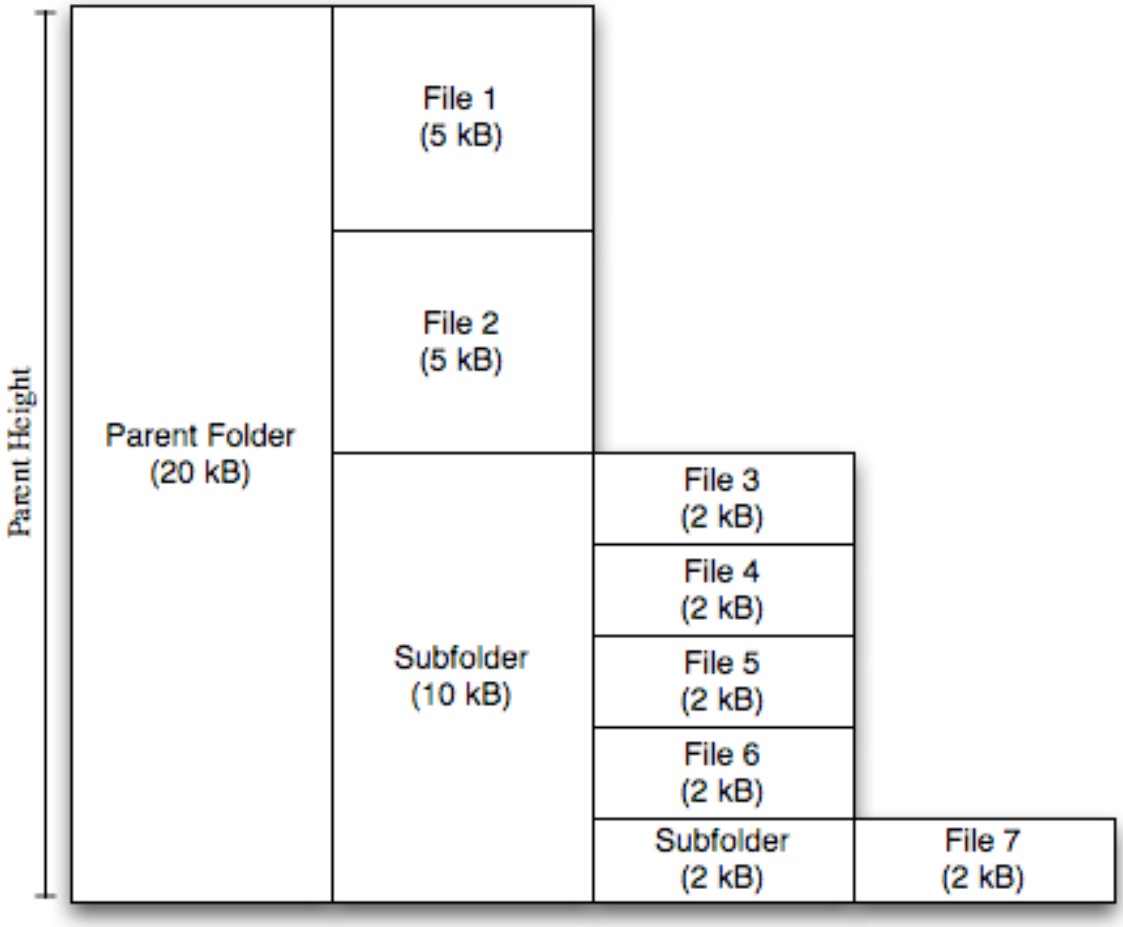

Figure 5: Example showing several directory levels

The visualization technique used in DiskGrapher helps to portray information about the contents of the hard drive and their size, while still maintaining the directory structure. Displaying information in this way is advantageous because it allows users to quickly

identify what folders are using the most disk space. Providing the user with a list of files and their sizes may allow them to find what is using disk space, but it is not as efficient as being able to see what is using the space. DiskGrapher's visualization technique gives the user a great deal of information about the contents of their hard drive in a structure similar to what they might expect by simply navigating through the file system.

IV. Program Requirements

When working on DiskGrapher, it was important to derive some basic requirements for what the program should accomplish. This helped with the design of the program, as it was clear what needed to be accomplished to make a functional program. The requirements of the program have been divided into system features and external interface requirements.

System Features

Hard Drive Scan

The program scans the contents of the hard drive, or a folder, and gather the tree structure starting at this location. It then calculates file sizes for every file and folder in the drive as the tree is being created. This forms the basis for the program, as the visualization is based on the tree that is created from this requirement.

Visualization of Directory Structure

The program converts a directory tree, which contains the location within the tree and size of every file on the drive or in the folder, into a graphical format that allows the user to quickly see what is using the most space. The visualization technique to be used maintains the directory structure of the disk, allowing users to quickly identify files using the most space, as well as the location of these files.

External Interface Requirements

User Interface

DiskGrapher utilizes a simple interface, but within this interface allows users to interact with the visualization. The main visualization window displays the visualization, and does not display anything else. Designing a main window in this fashion allows the user to quickly look at the visualization, without being distracted by a large number of interface elements. A second advantage to keeping the main window this simple is to allow the user the most available space for the actual visualization.

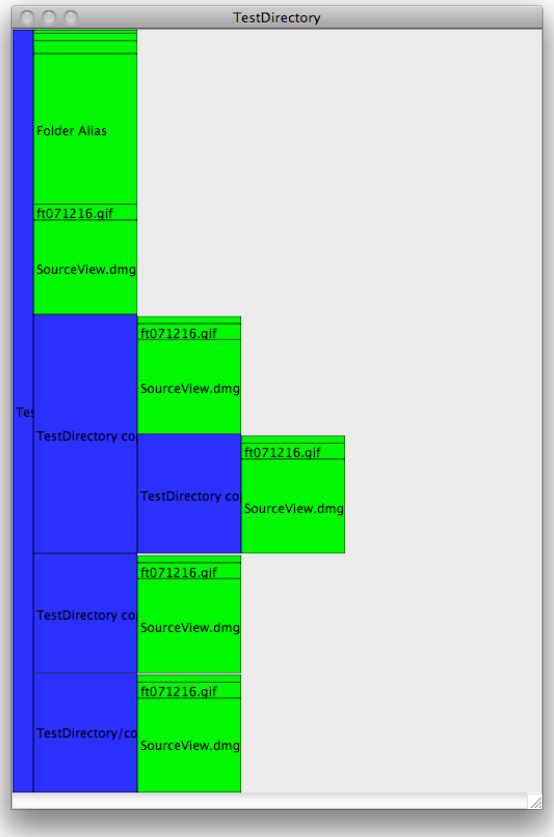

Figure 6: Main Window of DiskGrapher

A second, auxiliary window is used in this program to give the user more information about the file. The file info displayed in this window is determined by where the mouse is located on the screen. As the user moves the mouse over different files in the visualization, this window will change to reflect the current file. The window displays information on the file's name, size, modification date, and the type of the file, e.g. AVI Video file, etc.

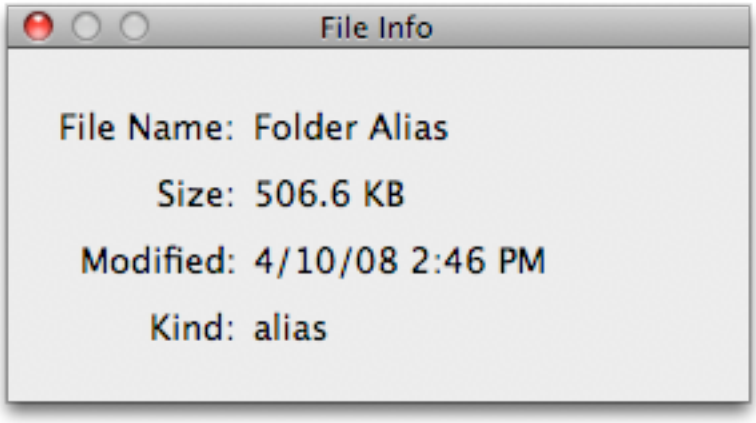

Figure 7: Window displaying file info

The user is prompted with a window, allowing them to choose a directory or disk to be scanned. This dialog allows the user to select only directories, preventing the users from scanning files. Once the user has made a selection, this dialog is then replaced with a window giving the user an indication that the scan is in progress.

In general, the GUI (Graphical User Interface) for DiskGrapher follows the Apple Human Interface Guidelines, found on the Apple Developer Connection website [5]. These guidelines are important for any application that runs on Mac OS X. They are designed to provide a unified interface design, making the program easier to learn for the user, as well as allowing easier localization of applications by reusing much of the code written by Apple.

Software Interfaces

DiskGrapher is written for Mac OS X, version 10.4 or newer. This allows the program to be used on machines several years old, while at the same time allowing the use of some of the newer technologies introduced in later versions of Mac OS X. The software utilizes the Cocoa API for the user interface and object tree storage, and most of the application is written in Objective-C. However, Cocoa is not well suited for low level file management, so the Carbon framework is used for gathering data on the files themselves.

V. System Architecture

System Model

Because this program is written in an Object Oriented Language, choosing a model is important to the structure of the program. Many different object models exist in Object Oriented Programming, but the model of choice for programming in Cocoa is the Model View Controller (MVC) architecture. This architecture was used as the basis for the Cocoa APIs, and many of the class methods work best when the application uses the same architecture. By using this model, DiskGrapher will fit well in the Cocoa frameworks, and make the program more efficient and responsive to user requests.

What is MVC?

Model View Controller is a programming architecture developed in 1978 for the Smalltalk programming language, developed at Xerox PARC [6]. According to [6], the MVC model was designed to "bridge the gap between the human user's mental model and the digital model that exists in the computer." The MVC architectural pattern has developed since the 1970s to be used in a number of GUI (Graphical User Interface) frameworks, including Cocoa, Microsoft Foundation Classes, and Java Swing.

The MVC architecture splits the program into 3 different parts, the model, the view, and the controller. The model portion of the program is used to represent the raw data. It separates this data from the rest of the program, allowing the view and the controller access to the data without concern for how the data is physically stored. This offers the advantage

that the data storage mechanism can change without affecting the controller and viewer portion of the program.

The view portion of the program is responsible for representing the information on the screen. This portion of the code generally renders the information to the screen, and typically uses user interface elements to do this. In Cocoa, this type of element will likely be a subclass of NSView, which has methods to draw to the screen.

Finally, a program contains at least one controller class. This class is generally located between the data model and the view classes in the program. By acting as the middleman between the view and the data model, the controller takes on the responsibilities of responding to events, such as user actions, as well as changing the model when necessary.

MVC in DiskGrapher

The use of the MVC architecture in DiskGrapher gives several advantages. First, it keeps the different functions of the program well separated. This is advantageous because it allows a large portion of the view code to be taken care of through the Cocoa APIs. Because this view code can be reused, more focus can be placed on writing the code for the controller and the data model, allowing better code to be written. The focus of the code being written is not on displaying data to the screen, but instead on describing how the data should look.

MVC also allows the scanning of the disk to be re-written without needing to change the view or controller code. By allowing this code to be replaced without having to rewrite controller or view code, new algorithms can be tested for scanning the disk, and quickly compared directly against the old code for efficiency testing. In fact, this has already been useful in switching between the Cocoa and Carbon APIs for scanning the disk.

Class Structure

DiskGrapher is based on 3 main types of classes. These class types correspond with the Model View Controller architecture. There are 3 model classes used in the project, DiskEntry, DiskDirectory, and DirectoryRoot. One class, DirectoryWindowController, represents the controller class. This class coordinates between the view and the model, and keeps the two views in sync when changes occur. Finally, the visualization and interface of DiskGrapher are created in the DirectoryViewController class. A class diagram of all the custom classes used in the program is shown below in Figure 8.

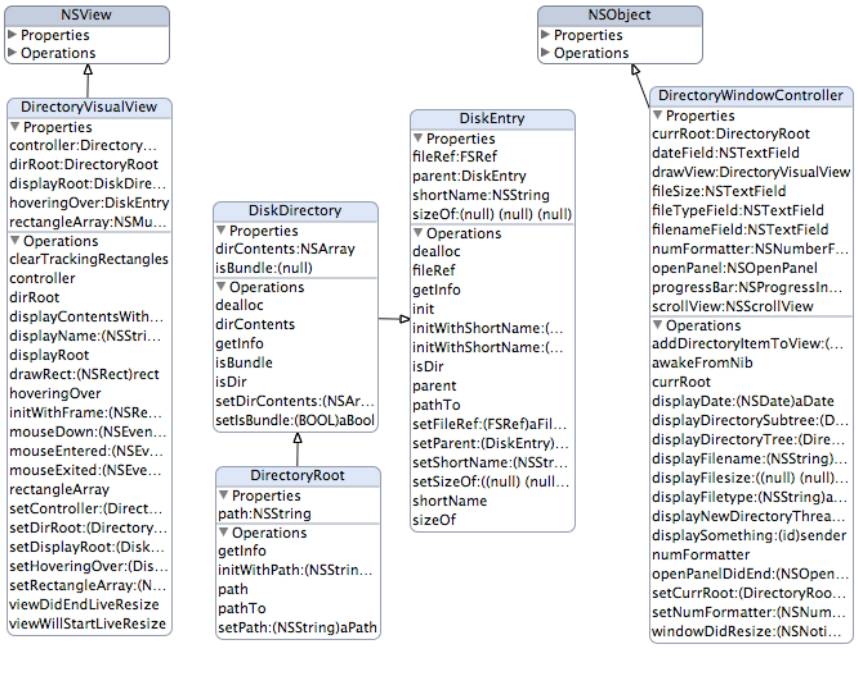

Figure 8: DiskGrapher Class Diagram

Model Classes

The 3 classes making up the model of the program are all based on the DiskEntry class. This class provides the foundation, providing a location to store the size of the file, as

well as the name. Storing the size of the file or folder is important, as it is the basis of the visualization. Getting the size of a folder is complicated, so storing the file size information during the scan is important so the visualization does not have the added overhead of finding this information out on it's own. Storing the name of the file or directory allows the path to the file to be recovered, important when constructing the visualization.

DiskDirectory is a subclass of DiskEntry, allowing the reuse of some of the code in the DiskEntry class. The use of polymorphism, instead of creating an entirely different class, offers some advantages. By reusing the structure of the DiskEntry class, DiskDirectory class members will respond to the same methods as DiskEntry, while allowing us to expand the implementation for functionality required only by directories. This functionality includes the ability to store the folder contents, as well as the ability to determine if the directory is an application bundle, which could be represented differently in the visualization.

The final class making up the model of the program is the DirectoryRoot class. This class inherits all the abilities of the DiskDirectory class, but is used to represent the head of the tree. This class includes a path member, which stores the path to the directory. This allows the user to scan any folder, rather than forcing them to scan the entire drive and navigate to the directory they are interested in.

View Class

The DirectoryVisualView class is responsible for visually displaying the data in the window. This class contains a reference to a DirectoryRoot member, which is the head of the scanned tree. The view then interprets the data, and presents the information visually on the screen. The view is created by drawing boxes on the view, sized proportionally to the size of the file or directory they represent.

In addition to taking care of the representation of the visual, the DirectoryVisualView class is also responsible for responding to user interactions. The view allows users to interact with the view by navigating into the hierarchy. In addition, the class tracks the mouse movements, and is responsible for providing the user with more information about the file.

Controller Class

DiskGrapher uses a single controller class to manage the window, and to connect it with the data model of the program. This class connects with several user interface objects, including the menu and the two windows used in the program, and coordinates actions taken in each area. When the user chooses a folder or disk to scan, this class is responsible for starting the scan of the drive, as well as giving the user feedback as to the status of the application.

VI. Future Project Plans

DiskGrapher is a useful project, but many different directions can be taken to improve the functionality of the software. These can help to improve the user's experience, as well as help to make the software more useful in general. The goal of this section is to outline some of the potential areas in which the program could be improved, as well as to give some ideas as to new concepts that could add functionality to this software.

Speed

One key area that could use improvement within DiskGrapher is speed. Currently, the algorithm takes several minutes to process the contents of a startup disk. This is due in part to the large number of files present on a hard drive. When the original scanning algorithm was written, the number of files that a hard drive may contain was not taken into account, and the efficiency of the scanning method could be improved.

The speed of the program could be improved considerably by improving the scanning algorithm. By reducing the number of function calls in the scanning function, the process of scanning the disk could be sped up considerably. Another option is to save previous scans and only update the information that has changed. By only scanning the changed files, scans could be performed much faster.

Visualization Update

Currently DiskGrapher, and the other programs it competes with, have one major weakness. Every time a file or folder on the disk is changed, even slightly, the entire disk or folder must be rescanned in the program to reflect that change. Because the scanning

process is such a huge task, this wastes time and computing resources. One of the major updates that would improve DiskGrapher is the ability to update the display without needing to rescan the entire drive.

One of the major goals of this program is to provide the user with a method to effectively manage their disk space. This will often include removing large files from a particular directory. By allowing the visualization to be updated to reflect these changes made by the user, the user would get an accurate idea of the disk space usage after the changes have been made. Two implementations are possible to allow for this to be updated. First, the user could initiate an update on a particular directory from the interface of the tool, and the display could then be updated after the portion of the disk specified is rescanned.

A second option to implement the ability to update the display is to use a new framework created by Apple for the Time Machine feature of Mac OS X, version 10.5. According to [7], the FSEvents framework was created to allow a program to track changes made on a hard drive, without needing to load something into the kernel of the Operating System. By keeping a record of each directory or group of directories that have changed since a certain time stamp, it is easy for a program to rescan only certain portions of a hard drive. Apple uses this to allow changes to the hard drive to be backed up quickly, without needing to scan the entire drive to determine what has changed. This same technology could also be used to allow DiskGrapher to update the visual display without requiring a complete scan of the drive.

Small File Visualization

With thousands of files on a hard drive, it is not possible to represent every file in a visual environment contained on the screen at one time. In order to quickly present

information to the user, DiskGrapher currently does not display files represented by less than 1 pixel on the screen. While this works well with individual files, when there are directories with large numbers of small files, gaps sometimes appear in the display.

DiskGrapher would be improved by creating a way of representing that there are small files present in these gaps. This could be accomplished by displaying a small subset of these files in a representation larger than they would appear if proportionally sized. Another option would be to create a block of a different color to represent the collection of these files, showing the user that there are files present, but they are too small to be displayed at the current directory level.

User Customization

DiskGrapher currently does not allow for the users to customize the display. Much of the visual display, such as the column width and the colors used for files and folders is fixed, and is not customizable by the user. Adding the ability for users to customize the display, as well as other aspects of the program, would make it more appealing to the user base, as they could then set the display to fit their preferences. Some of the customizations that could be added to the tool would be the ability to:

- Display bundles as a single file
- Change file color
- Change folder color
- Change column width

These changes would make the program more pleasant for the end user to use, and would make the program more useful to users in general.

VII. Conclusion

As media sizes continue to grow in size, the ability to effectively manage disk space is becoming more and more important. Many tools are available to help users visually manage hard disk space, using several different methods of displaying the data. Many of these programs are helpful for determining information about large files on the hard drive.

DiskGrapher attempts to present the disk usage in a different way. By creating a visual view of the hard drive that tries to maintain its structure, the display will allow the user to more quickly find the files that are using the most space and easily locate these files within the directory structure. Further, the display gives the user an indication of the percentage of the disk each directory is using. This is helpful in determining the location of the largest portion of the disk and to help the user deal not only with files that have grown to large sizes, but also to folders which have grown to include a large number of files that might otherwise have been overlooked.

For current information on DiskGrapher or to download the latest version, please visit: www.diskgrapher.com

Bibliography

[1] MildMannered Industries, “Baseline,” Mild Mannered Industries Homepage, March 2008. [Online]. Available: http://www.mildmanneredindustries.com/baseline/ [Accessed March 30, 2008].

[2] BerliOS, “BerliOS Developer: Project Info – MacFilelight,” Sept. 2006. [Online]. Available: http://developer.berlios.de/projects/macfilelight/ [Accessed March 31, 2008].

[3] MethylBlue, “Filelight | MethylBlue,” Feb. 2008. [Online]. Available: http://www.methylblue.com/filelight/ [Accessed March 31, 2008].

[4] Eriban, “GrandPerspective,” Sourceforge, April 2008. [Online]. Available: http://grandperspectiv.sourceforge.net/ [Accessed April 7, 2008].

[5] Apple Inc, “Apple Human Interface Guidelines: Introduction to Apple Human Interface Guidelines,” Jan. 2000. [Online]. Available: http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHIGuidelines/XHIGIntro/chapter_1_section_1.html#//apple_ref/doc/uid/TP30000894-TP1 [Accessed April 20, 2008].

[6] T. Reenskaug, “MVC: Xerox PARC 1978-79,” September 2003. [Online]. Available: http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html [Accessed April 21, 2008].

[7] J. Siracusa, “Mac OS X 10.5 Leopard: the Ars Technica Review,” Ars Technica, Oct. 2007. [Online]. Available: http://arstechnica.com/reviews/os/mac-os-x-10-5.ars [Accessed April 29, 2008].

[8] J. Porter, “Five decades of disk drive industry firsts,” DiskTrend. 2005. [Online]. Available: http://www.disktrend.com/5decades2.htm [Accessed May 6, 2008].

[9] B. Shneiderman, “Treemaps for space-constrained visualization of hierarchies,” University of Maryland, Dec. 1998. [Online]. Available: http://www.cs.umd.edu/hcil/Treemap-history/ [Accessed May 6, 2008].

Appendices

Appendix A: The Treemap Display

Ben Shneiderman developed the first disk visualization tool based upon the Treemap concept in 1990 [9]. Developed in response to the difficulties in managing disk space on an 80 MB hard disk drive used by fourteen users, the Treemap design was created with the goal of maximizing the amount of information that could be displayed in a small area. The first Treemap program, TreeViz, ran on a color Macintosh computer and was the basis of a paper published in the 1991 Conference on Visualization.

To create the Treemap display, the contents of directories are displayed in alternating horizontal and vertical directions. This means that the files and folders in the first directory are displayed with vertical rectangles, and directories or files in the second level are portrayed horizontally. A small example of a Treemap is shown in Figure 9.

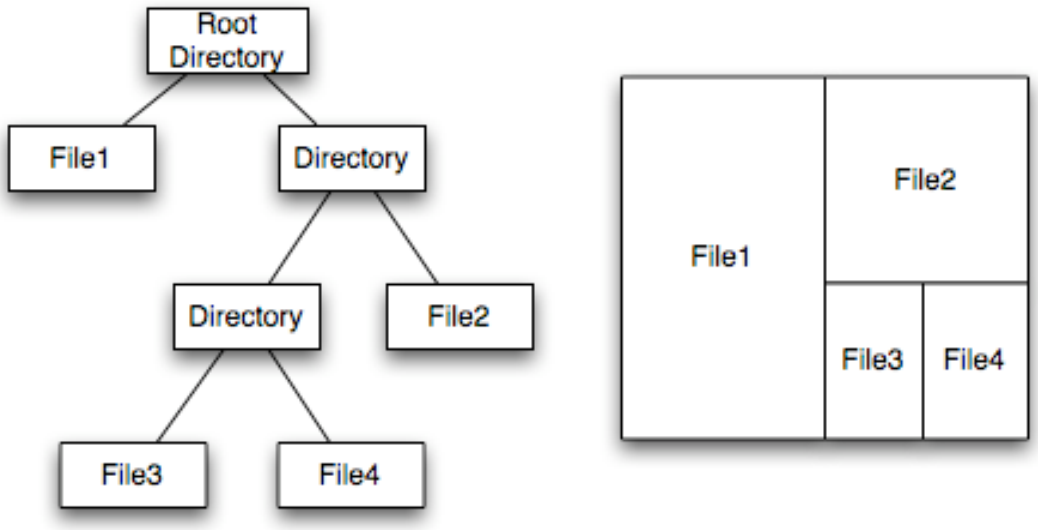

Figure 9: Example Directory Tree (Left) and corresponding Treemap (Right)

In this example, File1 takes up approximately half the scanned directory, and is portrayed by a rectangle that takes up approximately half of the display. A directory takes up the remaining half of the disk drive. The directory that takes up the remaining space in the scanned folder is split up in a similar fashion. File2 takes up a little over half the space in the directory, and the vertical area within the confines of the directory is split in the same method used to split the parent directory. The algorithm works recursively through the tree, using the same general method to divide up the area until the entire tree has been traversed.