An Antique Engineering Filing System For Personal Use and as a DBMS Case Study

Ronald I. Frank, DPS
rfrank@pace.edu
Information Systems Department, Pace University
Pleasantville, NY 10570, USA

Abstract

This is document serves four purposes. First: it is a documentation of a 0th generation engineering filing system. Second: it is a call for input from anyone familiar with comparable file systems. Third: it is an introduction to a very useful personal filing system. By documenting an historical file system and interpreting it in terms of modern technology, we derive a filing system that has proved robust and very useful for personal filing systems and for small business use. Fourth: this document describes a very engaging case study use of this simple system employed in a database course as a real database that the students can use.

Keywords: File System, Filing System, Database Case Study, ISAM Simulation, Pareto Principle, Information Retrieval.

1. THE PARETO PRINCIPLE

The Pareto Principle or "80-20 law" is a universal law appearing in economics, physics, sociology, and virtually all fields. The trick is to recognize it. It underlies a large part of information retrieval and filing systems architecture in particular. One it can be stated is that over 80% of the files will be accessed less than 20% of the time. [The exact percentages 80 and 20 vary from one specific case to another. The consequence of iterating the principle is that it is not hard to get close to 100 %].

Another filing system application of the principle is that over 80% of the filing activity (insertions / deletions) applies to less than 20 % of the files.

These assertions are personal opinions. I have not found them supported explicitly in the literature with the exception of the citation at the bottom of section 3 below. However, this does not mean that the Pareto Principle has not been applied in filing systems analysis, only that it may go unrecognized.

It is what underlies the workability of the filing system described below.

2. FILE SYSTEMS == FILING SYSTEMS

For the purposes of this document, the file systems we discuss are manual filing systems. We are focused on the relatively low activity type found in personal filing systems or small business filing systems. These are filing systems that may or may not have real-world manila folder files, or may have only soft documents in a directory structure on a hard drive. The size of the entire system is in the low hundreds of named files (directories) containing tens of documents. In one case discussed below (Nat Rochester's) it is just a sequence of documents. This could fill one or two 4-drawer file cabinets, although in Nat's case, the filing system took up a wall of about 8 to 10 5-drawer file cabinets.

3. A LITTLE FILING SYSTEM HISTORY

General History

We go back to the era just before computers, the 1930's, 40's and 50's. Personal and small business file systems were maintained using manila folders in file drawers.

The basic system was a set of file folders ordered alphabetically with titles written on the folder top, which stuck up for viewing. This is the typical stereotype used in suspense movies. The hero breaks into the of-

fice at night and rummages through the file drawers to find the key evidence – or whatever. We see his/her hands walking through the folder tops (folder tab indexes).

The file-labeling problem was to figure out in advance what categories of information were going to need to be filed. It was often necessary to insert new file folders when new categories of information came in.

The insertion problem was to figure out what existing category (folder) was appropriate for each new piece of paper. There are all kinds of jokes about misfiling papers. It was, and still is a real problem. Notice that I have two (actually three) first names. My information is often misfiled under my first name.

A Personal History

I began considering this topic when I realized one day recently, that two different engineers I knew used similar idiosyncratic filing systems for personal use. One was my father, Abraham Frank PE, who owned a civil engineering practice, and the other was Nat Rochester, the architect of the IBM 701, the IBM "Clam-Shell" PC Laptop, and an inventor of the Linked List data structure. Nat was an IBM Fellow working at the Cambridge Scientific Center (Massachusetts) when I knew him in the 1970s and early 1980s.

Both engineers used the filing system variants for their business and personal use. Abe died in 1951 so his use of the file system was essentially before computers were available while Nat, however, in the years I knew him, had access to IBM VM mainframe computers, so he was able to automate some of the functions I will describe below.

After I retired from IBM Research, I established a small software R & D firm. I used a variant of the file system for that business. Of course, I had available PCs (in fact a workstation) to automate the filing system.

A Short Filing System Technical History

Early books on basic business filing systems (Basset Agnew, and Goodman 1964), devoted much space to the hardware of the day, which was aimed at physical storage and retrieval of many file folders. They focused mostly on the use of alphabetically organized files, such as correspondence and card-based address files. This is still the way many filing systems are organized.

They employed color-coding for classification, which we still see in medical offices. The more advanced books (Place and Popham 1966) discuss geographical indexing and more sophisticated coding and indexing schemes. Their concern is the efficiency of insertion and retrieval of paper information. They still devote many pages to the products that contain and manipulate physical files. These books also discuss the physical set up of the filing room space. By 1966 punch card equipment appears in the books – at the end, but not computers per se.

In all fairness, and very relevant to the engineering filing system mentioned below, some basic books do cover pure numerical filing (Dickinson 1964) including more sophisticated digit manipulation schemes for increasing efficiency of insertion and deletion.

Not covered is the multiple-index problem. Say we have a preprinted order form filed by its number. How do we cross reference the ordering customer or the product ordered?

More modern books (Gold 1995) and (Diamond 1995) return to the numerical indexing scheme as being best for computer use. The idea is that a numerical file folder index allows cross-referencing other folders by their numerical index using modern relational data base systems (or in Nat's earlier use – text editors).

An example would be filing all of your patient data in your doctor's office by your patient number (which was your Social Security number before HIPPA). The file might be colored by gender. In a group practice, the file might also be colored according to your primary care physician.

Interestingly, Diamond (Diamond 1995), when discussing various sophisticated digit manipulation schemes for efficiency, mentions in passing (page 123): "the most recently created files are generally the most frequently referenced". This is of course the Pareto Principle in action!

The technical level of the system(s) covered here lies below the interest threshold of modern information retrieval (Baez-Yates

and Ribeiro-Neto 1999) but I am sure that it falls within their analysis. However, solving the efficiency problem in a small business or personal use filing system is not worth a large effort or a complicated implementation.

4. THE ANTIQUE ENGINEERING FILING SYSTEM

Abe Frank's Version

The file drawers contained manila file folders that were numbered sequentially from 1 up. The file drawers had numbers on the front indicating the range of file numbers contained inside. These were on paper inserts that could be easily changed as files grew and required reorganization.

Since this was in the 1930s-1950's, the cross referencing was carried out on 3x5 cards which contained what we would call today the key words describing the content of the folders. When a new category was needed, the next open numbered file was used and a card was created for it.

Searching required going through the 3x5 card file, but since the whole file system was small, that was not a major problem. This was used for Abe's personal file system at home, but it was modified in some way for the business, which involved about 20 employees and many customers.

Nat Rochester's Version

Nat's files covered decades of his engineering work at IBM. He was an IBM Fellow so was free to pursue his research interests. As I recall his system it was a variant of the one described above. The major differences were that there were NO file folders as such. There were dividers.

Every incoming DOCUMENT received the next sequential number and was filed. The dividers only indicated large divisions (every 50 items?) Being in IBM and having many file drawers to keep track of, Nat used a computer. However, this was mostly before personal computers and well before relational databases.

At his point I can not find anyone who remembers the details of how he cross-indexed his files. My best guess is either he had some special code for it, or most likely, being an IBM Fellow, his secretary/assistant used a text processing system such as IBM's Script system. It could search for key words and return every line containing them. I queried his principle colleague Frank Bequaert about this but Frank does not recall.

In addition, I must conjecture that items had to be purged from time to time as being wrong or irrelevant. I assume he had a method for reusing the sequential number. Clearly it was not much work to enter a marker (e.g., "OPEN") in the text for a filing position number, and search for "OPEN" when physically filing a new item, thus reusing open positions. A simple printout of the OPEN items would enable filing without computer lookup. However, this is conjecture. Alternately, in this kind of filing system, he could have just left the item number "empty" and never reused it again. The only loss would be in lines of the indexing database or text processor.

The point is that a simple text file with the sequential index followed by a list of descriptors can be searched very quickly by a text editor, which is all that is needed for efficient retrieval by cross indexing. Document insertion in this system is very easy. Put the new document behind the very last one (or in the first OPEN one), give it the available number, and type in a line of descriptors in the text file.

Ron Frank Version

The files

Immersive Systems Inc., my company, had two full time employees, five part-timers, and about five contractors at one time or another. The entire business was supported by one administrative workstation and one four-drawer file (plus many storage boxes for old files and documents)

The file system was a hybrid of the two mentioned above. There were manila files in the drawers numbered sequentially. The drawers had numbers on front indicating the range of files inside. The filing system was backed by a relational DBMS (Database Management System). A text editor with search capabilities could have been used.

The storage boxes were numbered sequentially and used as if they were file folders in a filing system but backed by separate tables in the DBMS.

All told there were less than 300 files and 40 storage boxes used in the 11-year history of the filing system.

The Cross-indexing

The relational DBMS (MS ACCESS) was used to maintain a table of file numbers with keywords describing the contents. Using forms input, a new file took just a few minutes to input. New inserts into a file folder might on rare occasion require a new descriptor. If there was a new item to be filed that did not fit the current descriptors, the next sequential file was started and a new table entry was made.

The descriptors were a primary name – the meaning of the file folder, a "Category", a "Location", and up to four other general descriptors for cross-reference. Rarely more than two of the latter were needed. All of these secondary descriptors were not unique. That is, any one could appear in any number of file entries. "Category" and "Location" were mandatory on every entry though not unique. Bills (from suppliers), and Subcontractors were two such categories. Locations had to do with where they file resided (which drawer, if in the filing cabinet, or where if outside of the cabinet).

Three practical "fudges"

Three practical manipulations made this work quite well. First, when filed information became obsolete, the file was emptied and the information put in storage. The file was returned to the drawer and its number was returned to an OPEN pool. The next needed file was taken first from the OPEN pool before any new number was generated. This controlled the growth of the total number of files.

Second, the table was printed out two ways: first, by numerical value with a primary descriptor, and second by sorted primary descriptor with the file number following (inverted file). These two printouts were posted near the file (see Appendix I below). This enabled almost all look-ups or filing without having to turn on the computer.

The storage boxes were treated as a separate virtual filing cabinet. They were numbered and their MS ACCESS table contained descriptors of their contents.

Third, when a folder became physically large, but the primary descriptor (folder label) still held, a second UNNUMBERED folder was put behind it to "expand" it. At rare times this caused a "drawer overflow". The last folder became the first of the drawer below. Only the numbers on the drawer fronts had to be reprinted. We never had to overflow into a second file cabinet.

Other files maintained

Since we were using a DBMS for the main physical files, we also used it to keep track of other business data such as inventories of supplies, book lists, phone lists, to do lists, etc. This was mostly before MS OUTLOOK or its equivalent (for address information and task to do lists).

Call for input

If you know of other instances of the historical use of this file system please email me the information at the email address in the title.

5. EDUCATIONAL USE: DBMS CASE STUDY

Data Base Course

I teach an introductory undergraduate (sophomore - senior) IS database course. I assign a very practical problem to my database classes: a personal job-search document system. The students use either MS ACCESS or SQL Server 2000.

Requirements

The students have to do the requirements for a case study. They understand the meta problem of a job search so they can generate the requirements fairly easily. They do the ERDs the tables and the implementation, which includes input forms and printouts.

The interesting part of the problem is that there MUST be an external REAL file system. Phone calls will be logged on paper to be filed in a log or log book. Letters of application must be filed. Responses will be filed. Brochures from prospective employers will be filed. Travel instructions and maps have to be kept. Records of interviews (notes) will be kept. A phone/address file is to be kept. Some of this might be PDA data, in which case the filing system can treat the PDA as a file or a set of named files.

The external file system can grow over time, so I have them use the sequential real file system I used in my business.

I found that this problem keeps students interested and focused. It brings home the meaning of the component processes in building a database.

6. CONCLUSIONS

The simple engineering filing system described has been used for over 50 years. With modern DBMSs or text editors for cross-indexing, it can be a very efficient scheme for maintaining real files in a personal file system or a small business file system. It also makes a very engaging case study for an introductory hands-on database course.

By now the database cognoscenti will have noticed that this paper is a description of a simulation of an Indexed Sequential file system (Hoffer, Prescott, and McFadden, 2005). The file system stores physical records (file folders, or in Nat's case just single documents) sequentially by number in the file drawers. The folder numbers and "names" are maintained in either a text editor or a database system. IBM calls it ISAM – Indexed Sequential Access Method, which predated VSAM (Virtual Sequential Access Method).

The primary key is the file name. The combination of the file name descriptor and number could be thought of as a composite key. Both fields are unique and uniquely paired. The other descriptors (cross-references) Category, Location, and 1 to 4 are non-unique secondary indices.

Rearranging the file folders in the drawers when large folders are purged is just a form of dynamic physical record reorganization (garbage collection).

It is important to note that this antique engineering file system was invented and used decades before ISAM was implemented.

The fact that this is a simulation is useful in the classroom case study use of the tool.

Of a somewhat related historical note, ISAM for the IBM 360 family of machines was implemented in 1963-64 on a 360 machine located on the third floor of the IBM Systems Research Institute (SRI) in New York City across First Avenue from the UN. SRI was IBM's internal Computer Science graduate school before there were such things. The 360 family had no operating system at that time. Each application ran by itself under a simple tape monitor system. We SPOOLed cards to tape.

ISAM was implemented on the third work shift, because first and second shifts were used for teaching and research – the primary purpose of the machine. Ascher Opler of Computer Usage Corporation (CUC) led the development under contract to IBM. As a member of the IBM Mathematics and Applications (M & A) department, which shared the site, I managed the computing facility for M & A and SRI and "rented" CUC the third shift for "funny money" i.e., internal funds transfer (for a profit!). I believe this third shift development project was the origin of ISAM, the first of the IBM access methods.

7. ACKNOWLEDGEMENTS

I thank my former R & D colleagues from the IBM Cambridge Scientific Center, Frank Bequaert and Dr. Coyt Tillman, for spending the time to answer my questions about Nat's system. Unfortunately neither Abe Frank nor Nat Rochester is still alive.

8. REFERENCES

Baez-Yates, Ricardo and Ribeiro-Neto, Berthier (1999). Modern Information Retrieval. ACM Press (Addison Wesley /Pearson) NY, NY.

Basset, Ernest D., Agnew, Peter L., Goosman, David G. (1964). Business Filing and Records Control. 3rd Ed. South-Western Publishing Co. Cincinnati, Ohio.

Diamond, Susan Z. (1995). Records Management. 3rd Ed. AMACOM. NY, NY.

Dickinson, A. Litchard (1964). Filing and Finding in the Office. The Business Press Elmhurst, Illinois.

Hoffer, J. A., Prescott, M. B., McFadden, F. R. (2005). Modern Database Management. 7th Ed. Pearson Prentice Hall, Upper Saddle River, NJ.

Gold, Gloria (1995). How to Set Up and Implement a Records Management System. AMACOM. NY, NY.

Place, Irene and Popham, Estelle (1966). Filing and Records Management. Prentice Hall Englewood Cliffs, NJ.

APPENDIX I Sample Output

(Usually put on the filing cabinet for easy reference).

FILE CONTENTS NAME SORT

File Description	File #	Do	File Description	File #	Do
3D STUDIO AUTODESK	72		FALKOFF, ADIN	106	
3M	79		FEDEX	254	
ACCOUNTANT VASSALLO, FRANK	85		FOODMAN, HAROLD	197	
ACM, ANSI, IEEE, ISO	37		FRANK, JUDITH M. (MEDSOFT) NLM	172	
ALBERTI, GLENDA & PETER	125		FRANK, LEE	191	
ALCOHOL EFFECTS	66		FRANK, RI - CORRESPONDENCE	137	
AMERICAN EXPRESS ADMIN	179		FRANK, RI - GATEWAY INFO	103	
AMERICAN EXPRESS BILLS	214		FRANK, RI - ISI TAX FORMS	151	
ANDERSON JIM	182		FRANK, RI - MICRON LAPTOP	96	
APL	126		FRANK, RI - MOBILE PHONE ATT	113	
ARONOFF, ALAN	206		FRANK, RI - OLD NOTEBOOKS	169	
AT&T PHONE	42		FRANK, RI - RESUME	70	
BELL ATLANTIC- 7600	190		FRANK, RI - SOFTWARE OWNED	193	
BUSINESS	15		FRANK, RI - SW TOGET / UPGRADE	180	
BUSINESS BIBLIOGRAPHY	216		FRANK, RI - WRITINGS	192	
BUSINESS MANAGEMENT	162		GRAHAM, ALAN	213	
BUSINESS PLAN	28		GRANTS GUIDE	47	
BUSINESS SERVICES	165		GRAPHICS HARDWARE	252	
C++ & JAVA	131		GRAPHICS SW NOT OGL	224	
C++ CLASS LIBRARIES	61		GREAVES, TOM	98	
CAVREL, IRA	54		GREENBERG, MICK & LEE	95	
CENTER OF ALCOHOL STUDIES RU	155		GROPPER, JOHN	51	
CHASE BANK ACCOUNT	141		GROSSMAN, DAVID D.	194	
CHASE CHECK REGISTER	163		GROSSMAN, FRED	223	
CLEMENT, JOHN	97		HANNES, RD - CV, PC, INFO	231	
CLIENT - 1 -	255		HARTFORD U.	196	
CLIENT / SERVER COMPUTING	237		HOLD	74	
CLIPS	40		HOLOGRAPHY	174	
COHEN, MALCOLM	245		HUMAN FACTORS	166	
CONNECTICUT STATE TAX	43		IMAGE PROCESSING	176	
CONSULTING RESOURCES	118		IMPEDIMENT	46	
COSMIC	44		INDEX	0	
CRAWBUCK, JOHN	199		INGRAM MICRO DISTRIBUTOR	5	
DATA ACQUISITION	7		INTERNET	130	
DIGITAL SIGNAL PROCESSORS	101		INTUIT - TURBOTAX, TTB, QB	87	
DISABLED	58		ISDG - CORRESPONDENCE	257	
DUN & BRADSTREET	138		ISDG - LABELS & LOGO	259	
DVORKEN, HENRY J.	227		ISDG - ORGANIZATION	64	
DVORKEN, LEO V.	57		ISDG - VILLAGE BANK	258	
E-COMMERCE	260		ISI - AUTH. ED. RESELLER	154	
EDUCATION FOR US	189		ISI - ACCOUNTANT	119	
EDUCATION TECH & SOLICITATION	203		ISI - APPLICANTS	164	
EMPLOYMENT & HEADHUNTERS	80		ISI - ARCHIVE INDEX	133	
EMULATOR ARTICLES	59		ISI - BIZPLAN	1	
EXPERT SYSTEMS	226		ISI - BURDEN RATE	29	

11/12/00 3:17:58 PM Page 1 File = ISIFILE Total File Count = 90

APPENDIX II SAMPLE DATA

This is not an ACCESS table in normal form. It is an example of the type of data used in the application of the file sy

- The file number is the sequential number of the real file.
- The File descriptor is the "file label" in a regular file system – the primary key.
- The Category or contents type is a secondary grouping found important because we often needed to find all f than a descriptor. For example, all bills regardless of who sent them. It is another, special, secondary index.
- The Location was the drawer in which it should be found (if we kept our book inventory up to date with our physi
- The file date start was when a file was first established versus the second file date that might be used for a majo
- The second file date was rarely used. It was intended for a major change in the file. Any file of bills was change
- The file descriptors 1 to 4 were used for cross-referencing. It turned out that we rarely needed more than two.
- Memoranda are large text fields, also not used very much.

Example: The printed sheets on the file cabinet (cf. Appendix I above) would show our accountant's file as being nur merical listing would show file number 2 was ACCOUNTANT. If we needed to know his name we could search the dat we had a searchable address and names file that we would use to search for his name, address, and phone number.

If we needed all employee folders, we could search all Descriptors for "Employee". A well-designed database with us makes this searching easy. Pulling the folders was very easy also since it was direct to find the numbered files and d unambiguously. *[Notice that File # is monotone increasing as a function of Location (Drawer #) == Physically Index*

File #	**1**	**2**	**3**	**4**	**5**	**6**
File Descriptor	BUSINESS PLAN	ACCOUNTANT	BANK ACCOUNT	C++ & JAVA	Smith, J	Kelly, R
Category	WORD DOC	BILLS	STATEMENTS	MANUALS	FORMS	FORMS
File DATE Start	1/1/1999	1/2/2001	1/3/2001	2/2/2003	5/4/2002	1/6/2001
Descriptor 1	ISI	Smith, John	Chase Bank		Full time	Part time
Descriptor 2					Employee	Employee
Descriptor 3						
Descriptor 4						
Location	**Drawer 1**	**Drawer1**	**Drawer 2**	**Drawer 2**	**Drawer2**	**Drawer3**
File DATE						
Memoranda						