Community Filing with PC's

Author: Jon Baker
Version: 2.0
Printed: May 30, 1988

Nine Mile Ride
Wokingham
Berkshire
RG11 3LL

ENGLAND

TABLE OF CONTENTS

Abstract			ii
1	Introduction		1
	1.1	Community Filing	1
2	General Requirements		3
	2.1	User Scenarios	3
	2.2	Host Filing Services	4
3	The PC World		5
	3.1	Architecture	5
	3.2	Specific PC Needs	6
	3.3	DOS Environment	7
4	Solutions		8
	4.1	Additional Features For PC Integration	9
		4.1.1 Interface To Filing Services	9
		4.1.2 Item Transfer	9
		4.1.3 Integration	10
	4.2	Implementation Details	11
		4.2.1 Item Transfer	12
		4.2.2 PC Controller	12
Summary			13

ABSTRACT

HP File/Library is HP's first community filing product and is currently available as a "bolt-on" product to HP DeskManager. Since its introduction in 1986, the need to extend the service to PC users has become increasingly important. In addition, customers without HP DeskManager have also expressed an interest in allowing PC users to share information effectively.

The integration of PC's in a community filing environment presents a number of new and challenging problems, these include allowing both terminal and PC users to share a single community file store, integration with PC applications, coping with limitations set by the MS-DOS file store and finding a suitable transport architecture which offers a satisfactory level of performance.

This paper presents solutions we are devising to offer community filing services for PC users. It includes an overview of the product's architecture and a technical discussion of the main design areas including the PC component, the transport mechanism and the interface to File/Library.

In this paper I propose to examine how the PC fits into the world of community filing. It is based on my own experiences from working in this area. For those new to community filing I will present a short summary explaining what it is and highlight the main benefits to end users. Following that I will concentrate specifically on the integration of PC's into such an environment, I will discuss the problems found during design and present solutions we are currently devising.

1.1 Community Filing

Community Filing is quite simple the "*Sharing of large amounts of unstructured information amongst a group of office users*".

What are the commonest types of information that users want to share ? There are three main categories: textual information (the most dominant), spreadsheets and graphics. Note we are **not** infringing into the world of databases, the information we are storing is *unstructured* consisting of many *items* of varying size and format (typically these will be files created by applications).

What operations will users want to perform on this shared information ? Certainly they would want to be able to read it (or *browse*), they may want to use an application to *edit* an item or simply extract selected parts using *cut* and *paste* facilities. They will want to ensure that their information is secure, i.e. they will want to specify who can read or change their items. Users may not require all items to be on-line at the same time, this gives the system the opportunity to *archive* items onto an offline medium such as magnetic tape to save disk space. These archived items could of course be retrieved when needed at a later date. References to non-electronic items such as books or manuals are also becoming increasingly more important. We refer to these as *offline* items. The information shared amongst the group now becomes an index entry which will probably contain a reference to the physical location of the offline item itself.

The scope of the community sharing the information could be any group of people you wish to name. Here are a few examples: A small group of colleagues, typically in the same department possibly located within close proximity of each other. A product team could form a group consisting of a member from each department of a company. Inter-office groups might consist of people within the same company but who are resident in different offices, maybe a group of managers who communicate regularly with each other. A standards committee might have representatives from a number of different companies, the members of this group could potentially be spread world-wide. Remember also that any user may be a member of more than one group. It is common these days to hear these groups being referred to as *Work Groups*.

Before we look at the requirements for a community filing product we need to clarify the basic hardware architecture for the type of system we are discussing.

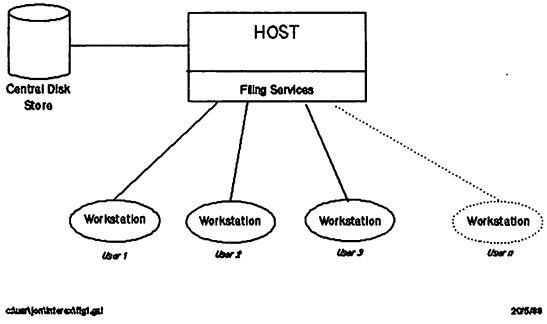


Fig-1.1: Elementary architecture

All the information for a work group is centralized on a host machine, every member of the group needs to be able to connect to this host. The filing services that manipulate this information can support many different groups on the same machine, note that all items belonging to a group are held on this one machine. The workstation could either be a terminal or PC, we will discuss the actual choice of host in chapter 3.

Having looked briefly at what community filing is let us look further at the requirements for a community filing product. To help us I have written three scenarios where community filing could be used.

2.1 User Scenarios

A team of six technical writers are preparing a manual for a new product, they are using a popular PC word processor to create it. Each writer has been assigned a certain area to concentrate on. The manual consists of a number of chapters, they decide that each of these will be held in a separate file. There are many chapters where each writer will want to make a contribution, therefore control is needed over the editing of files to prevent two or more of them attempting to update a chapter at the same time.

A team of lawyers are preparing the defence for a complex law case, they need to perform extensive research on similar cases from the past. Their firm holds electronic copies of transcripts from all the cases that they handle. They wish to perform generic searches on the transcripts to retrieve any information that might be useful. The store of information holding the transcripts is constantly being added to and is rapidly increasing in size.

The personnel department for a large company are in charge of administering employee evaluations which every employee has annually to discuss his/her progress over the year. Each evaluation is held electronically and stored for future reference. Security is of great importance, the company has a hierachical structure to its organization, the personnel department dictates that an individual may only *read* the evaluations of people working directly under him. He/she may not read the evaluations of colleagues who are at the same level in the organization or the evaluations of people working for them. One final security requirement is that the only people to have edit access to an employees evaluation are his/her manager and the head of the Personnel department.

2.2 Host Filing Services

Fast retrieval information from a community file store is the key to success. Todays modern searching techniques are based around the concept of *full-text-searching*, that is the ability to specify a search with respect to the actual content of the information. For example I might specify a search to retrieve all items in the file store that contain in them the words "Hewlett Packard". The specification of searches can be made quite complex by introducing logical operators and other notations. For example I could ask to search for all the items containing *either* the word "Hewlett" or the word "Packard" or both. A full-text-search mechanism works by creating sophisticated indexes that are built as items are added to the store. All the scenarios in 2.1 would need a good searching mechanism particularly the team of lawyers.

Full-text-searching really only makes sense for items containing text, which is probably the majority of information filed. However what mechanism do we use for retrieving non-textual items? One way is to use *fixed-attributes*, these are assigned to every item that is filed into the community store. Typical types of attribute may include Author, Subject, Comments and the most powerful attribute User-defined Keywords. Your searching mechanism now searches on the values of these attributes, for example you could retrieve all items whose author is "Jon Baker". You will of course want to be able to specify searches in terms of both the content of items and the values of attributes combined, for example retrieve all the items with "Hewlett Packard" in the content that were authored by "Jon Baker". Fixed-attributes are the way in which you can implement the indexes to the offline items mentioned earlier. Remember that offline items are simply a reference to a physical item, there is no associated electronic item just an entry with attributes ...

Once you have a retrieval mechanism your next requirement is to be able to describe exactly whom you wish to allow access to your information. This introduces the topic of *Security*. If we define an item as the unit of information within your file store (this would probably be a document, spreadsheet, chart etc.) a comprehensive security system should allow you to configure security at this level. How is security specified ? There are basically two parameters firstly the security attributes for an item i.e. Read/Write/Delete and secondly the scope of the attribute which is specified in terms of a pre-defined group, this group would be configured elsewhere and allocated a name for referencing. The personnel department in the third scenario placed great importance on security with a very complex structure

Editing items was an operation we identified in chapter 1, but how would this work? The actual editing itself is done by the application that originally created the item. The community filing product has to provide a mechanism which allows the application to be invoked and secondly lock the item so that it cannot be accessed by anyone else while changes are being made. It would also be useful if other users could see who was editing a particular item. The technical writers in the first scenario need a good editing mechanism to prevent loss of data through simultaneous updates of the same chapter.

Archiving is an essential feature of any community filing mechanism. With the amount of information that could be filed being virtually unlimited, it is

necessary to periodically archive off onto a secondary media (such as magnetic tape) items that are infrequently accessed. But note one important feature, the indexes created for the item (that are used for searching) remain intact, this means that archived items can still be referenced with searches. The user will also need to be able to issue a command to retrieve an item from its archived media back into the store if the item needs to be read or changed. Again all the scenarios could use archiving but the team of lawyers are the best example. They would probably have a very rapid growth in the amount of information being filed into their store.

Terminal users directly access the filing services that are running on the host, they use host based applications to create items. PC's distribute applications onto local machines. This chapter discusses the extra requirements that this places on our community filing product. Firstly let us refine further the type of hardware we intend to use.

3.1 Architecture

From previous experience in building terminal-based filing products we have learnt that 3000 hosts are well suited for locating the core filing services (i.e. storage of the information, searching mechanism, security, archiving, administration and maintenance). This is because of the amount of CPU resource necessary to support community filing together with the ability to connect large amounts of storage. The host itself maybe used for other activities such as mailing, databases etc.

The PC workstations will typically be XT, AT or PS/2 compatibles running MS-DOS or OS/2. They could be linked to the host computer with either an Asynchronus or LAN connection.

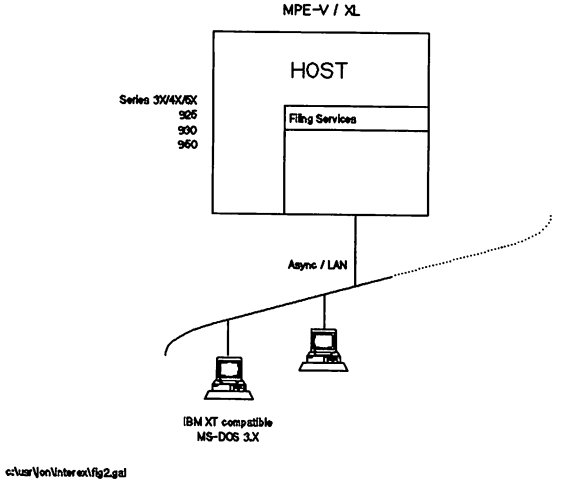


Fig-3.1: Hardware configuration

3.2 Specific PC Needs

PCs off-load host resource onto individual machines which are better suited to dealing with the intensive screen I/O demands of word processors, spread sheets and graphics packages. But they do introduce a number of new problems:

- Applications are now resident on a users local PC rather than being centralized on a host.
- Users now have their own local disk storage to hold files. These cannot easily accessed by other people (unless you use a LAN).
- With all the PC software available there are now many more file formats to deal with.

What additional features do we need to integrate PC users with our host based community filing services?

Interface

We will need an interface on the PC to the host based filing services. This will allow users to access the features we have mentioned already (searching, archiving, security etc.) as well as the item transfer mechanism identified above.

Item Transfer

PC users create items that are held in local storage (i.e. DOS files on floppy/hard disc). With the requirement that the filing services are resident on a host machine we need a transfer mechanism to move items between this and a users PC. This process is likely to be time consuming and should therefore be minimized.

Integration

To work effectively we need a good level of integration between the community filing product and the users PC applications, users maybe retrieving information from the community store to change an item or use parts of it in creating another. This integration typically means using MS-DOS and its file store (most PC applications are invoked from DOS and themselves use DOS files as their unit of store). The integration problem is easier to solve for terminal users since their applications and the filing services can be located on the same machine.

A further requirement is the ability to support *mixed environments* of terminal and PC users. Many companies are in the process of moving terminal users to PC's and therefore may have work groups consisting of both. Conversion facilities will be needed for Terminal and PC users to be able to edit each others items.

3.3 DOS Environment

MS-DOS is the operating system installed on the majority of PC's. Unfortunately it is not well suited to integrating with host based filing services. Firstly there is limited control over DOS files, users can copy them, move them, delete them, or rename them as they wish. There are no unique references to keep track of a file throughout its life (DOS handles are not good enough). To support the checkout mechanism that we described in 2.1 we will need to create a link between the PC and the community file store on the host. When we have no way of tracking a DOS file during its life the link becomes difficult to maintain.

MS-DOS has no support for the coding of files to distinguish their type e.g. Lotus, AdvanceWrite, HP Word/PC etc. MPE for example supports file codes which can be allocated to represent different types of files. Often with DOS the only way to identify a files type is by examining the extension in the file name.

Another feature that would be useful for integrating community filing would be the ability to store some special user defined information in a file somewhere without disturbing its actual content. They could be areas of a file which you can allocate and use as you wish which are ignored by applications when they use the file. This feature would be useful for holding the link information described above.

Many users find difficulties in using MS-DOS, they find some commands difficult to understand and remember (especially from the DOS prompt). Although we are not trying to offer a new shell for PC users we must be careful in the way in which we use MS-DOS in our integration.

In 1986 we released a terminal-based community filing product called **HP File/Library**. HP File/Library runs on a 3000 series host under both MPE-V and MPE-XL and the current version is available as an optional bolt-on to HP DeskManager.

HP File/Library consists of a number of flat *catalogs*, created as necessary by users. On entering the Library the user is presented with a list of these catalogs. Each catalog contains the items themselves, they have no hierachical structure (like DOS directories for example). This has the distinct advantage that users do not need to know where to file an item other than the appropriate catalog. The division and retrieval of information within the catalog is achieved through the search mechanism.

HP File/Library can be used to index *any item* held inside or outside of HP DeskManager, including MPE files and *paper* items.

When an item is filed into a catalog it is indexed by 8 fixed attributes which allow you to uniquely identify each item. For items created in or imported into HP DeskManager the user is prompted for three of the fixed *attributes*. These are common to every item indexed in the library : Author, Keywords and Comments. HP File/Library automatically provides a further 5 attributes for each item indexed : Subject, Creator, File Type, Status and Create Date.

Items are *retrieved* from a catalog by searching on one or more of the attributes. Each catalog has an automatically maintained keyword dictionary to help find suitable keywords for searching. Searching on the "Comments" attribute will perform a "full-text-search" on the Comments on every item indexed in the catalog.

HP File/Library offers *security* at three levels (access to the Library, to the catalogs and to the items within them). This allows you to determine what a user sees and how they can manipulate the items that they have access to.

To prevent two or more users trying to simultaneously update the same item there is a *checkout* mechanism. This lets a user copy a document to his work area, edit it and check it back in. While a document is checked out no one else can change it.

Browsers and *conversion* facilities allow users to read and edit documents stored in HP document formats and DCA document formats. A plug in mechanism in HP DeskManager allows users to write their own converters for other document formats which HP File/Library can access.

HP File/Library has an *archiving* facility to allow items to be stored off to magnetic tape. When an item has been archived, references to it are still maintained in the catalog indexes for searching and the item can be retrieved if necessary.

The *next release* of HPFile/Library has several major enhancements including a *full text searching* mechanism on the content of items, in addition to the current searching facilities. There are also plans to provide both a *standalone* version of HP File/Library and a version integrated with HP DeskManager The

third major enhancement in the next release of HP File/Library is that it will provide the host filing services for *PC users*. The rest of this chapter discusses how the PC user will be given access to these host services.

4.1 Additional Features For PC Integration

How do we implement the features described in chapter 3 to integrate the PC user with File/ Library?

4.1.1 Interface To Filing Services

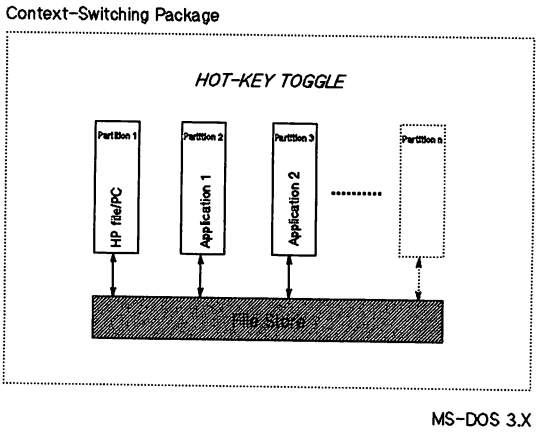
We have decided to use the current terminal-based HP file/library interface via AdvanceLink terminal emulation as a simple and effective solution. A full native PC interface would have been considerably more complex, needing a mechanism that could cope with the frequent transfer of information between the PC and the host.

4.1.2 Item Transfer

To initiate the transfer of items between the PC and host we have extended two existing HP File/Library commands. Firstly we have a command to *copy* a PC file into the Library. We have extended the *checkout* command to checkout an item in the Library to a file on the users PC (he supplies the DOS filename). We have a new command invoked from the PC to *checkin* a PC file that was previously checked out from file/library.

4.1.3 Integration

Quick access to the host filing services from applications is essential. The neatest solution is context-switching. There are a number of context-switching products already available on the market (e.g. MS-Windows), typically they allow the user to set up a number of partitions, each partition appears to an applications as a virtual PC with a full complement of RAM. The user can switch from partition to partition by using a pre-defined hot-key. The underlying context-switching mechanism saves the state of a partition when it is switched out to load another. They can use a number of different types of storage to save these states: normal unused RAM, Extended Memory, Expanded Memory and even the hard disc.



c:\usr\jon\interex\fig3.gal 12/5/88

Fig-4.1: Context-switching

We considered the use of a Terminate-Stay-Resident component which could sit under all applications. These however can be unreliable and cause problems. It would also present a large memory overhead to the user.

4.2 Implementation Details

The product has four major components:

Control menu:	User is presented with this menu on startup of the product.
Filing services:	This component is made up of the AdvanceLink terminal emulator accessing the HPfile/library interface.
Item transfer:	There are a number of AdvanceLink command files and some enhancements to the HPfile/library interface to perform the transfer of items between the Library and the PC.
PC controller:	PC software that manages the checked-out items while they are on the PC.

There is an interface to this component which allows the user to list all the items he has checked-out on his PC and the DOS filename given to each.

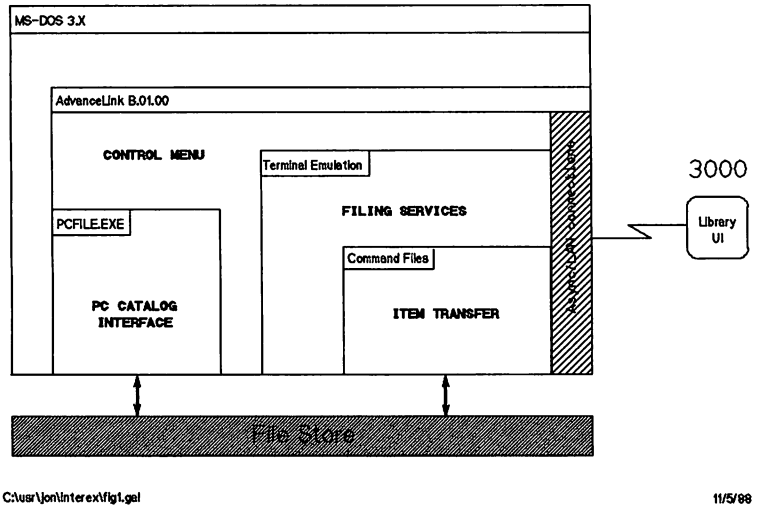


Fig-4.2: Component Integration

4.2.1 Item Transfer

With AdvanceLink being used to provide terminal emulation we decided that we could also make good use of its file transfer and command language capabilities to implement the item transfer mechanism. It can do this across both asynchronus and LAN connections. The performance of the file transfer mechanism is satisfactory, we recognized in our initial investigation that this was always going to be the major performance overhead in the product.

4.2.2 PC controller

This controller maintains a table of entries, one entry for each item checked-out to the PC. Each entry contains information which links the file on the PC with the host store. It contains information to allow us to check the file back into File/Library to where it came from.

This table is also a list of all the items that are currently checked-out on the users PC. We will provide an interface so that the user can view this list and issue checkin directives against selected items.

To minimize the number of file transfers we have included a mechanism inside the controller called *Minimum Data Transfer* (MDT). It works as follows, suppose a user checks-out a large report from the Library, he makes a number of changes and then checks it back in. The copy transferred to his PC when he checked the item out is not deleted after the check-in operation. He then decides he wishes to make further changes to the report. The MDT mechanism discovers that this user has checked the item out before and that a copy still resides on his PC, by using version information that is stored both in each catalog entry in the Library and also a special table created on the PC, it can decide whether the item has been altered by anyone else since this user last checked it in. If no changes have been made no transfer is necessary, the user can use the copy he still has on his PC.

SUMMARY

What is Community Filing ?

- Sharing large amounts of unstructured information amongst a group of users.

What are the key features ?

- centrally accessed store.
- searching mechanism.
- security.
- editing control.
- archiving.

What are the key benefits ?

- fast and effective information retrieval.
- reduce disk storage requirements through archiving and minimizing data duplication.
- restrict access to authorized users only.
- increase the availability of information.

What do we need to integrate PC users ?

- PC <-> host item transfer.
- interface to host filing services.
- application integration.

What solution are we devising ?

- use HPfile/library for host services (searching, security, archiving etc.).
- use AdvanceLink terminal emulation to access these services.
- AdvanceLink command files and new HPfile/library commands for item transfer.
- controller on PC to manage the checked-out items.