A FILE TRANSFER PROTOCOL AND IMPLEMENTATION

B. Butscher, W. Heinze

Hahn-Meitner-Institut fuer Kernforschung Berlin GmbH
Department of Computer Science and Electronics

Glienicker Straße 100, D-1000 Berlin 39

1. Introduction

This paper presents a file transfer service in a heterogeneous computer network (HMINET). The basic architecture models are presented, the file transfer protocol is described and the implementation in the HMINET is shown with the functional capabilities and the command structure.

A transfer of files between different computer system can be regarded as a service, which is divided into four steps:

A access to a file, which is administrated in a File Management System (FMS), residing on one computer system

B transfer the file data by any transport medium

C creation of a file copy in another FMS

D additional operations on the transferred file, e.g. print out or conversion

The usual case of file transfer (FT) between different computer systems is an offline transport by means of any storage medium (disks, tapes, etc.). Some disadvantages arise by this method:

- offline transfer takes a long time
- incompatibility of peripheral devices, storage medium and data representation can make file transfer impossible or lead to an additional overhead
- for step A and C the user must be familiar with the control languages of every participated operating and file management system
- additional operations on the transferred file must be organised under the system control of the destination host

A lot of advantages offers a File Transfer Service, integrated in a network, which links together the above mentioned computer systems (hosts):

- with a reliable transport system in the network a fast online transfer of the file data can satisfy step B
- step A and C are controlled by the File Transfer Service with only one standard interface to the user
- additional operations on the copied file can be managed during the transfer of the data (conversion of file structure, file data and subsequent operations e.q. print)
- ɔeripheral device support by involved operating and file management system under File Transfer Service control

The HMINET is a private star-shaped, heterogeneous network, which connects 22 process control computers and 3 mainframes. One basic function in that computer network is such a file transfer service. Typical applications of this service are:

Copy of data files from and to mainframes and process control computers

Transfer of programms and runable modules

Remote Job Entry and remote spool services, etc.

2. File Transfer Architectures

The system, providing a file transfer service in a network, can be considered to be composed by two different kind of processes:

Service process providing access to files in the involved FMS

Control process supervising and synchronising the operations of the service processes to perform the file actions due to the users request

In a general File Transfer Model one control process and two service processes are required (Fig.1):

- one service process on the host where the source file resides, called producer
- one service process on the destination host to which the file has to be transferred, called consumer
- one control process on any host, which represents the interface to the user and supervise the service processes

The communication between these processes is done by different transactions. Command flows for the synchronisation and controlling of the service processes are represented by transactions C1 and C2, the data transfer between producer and consumer is performed by the transaction D.

To guarantee these transactions, the network must provide a transport system with additional Inter Process Communication facilities (flow control, error detection and recovery, process control).

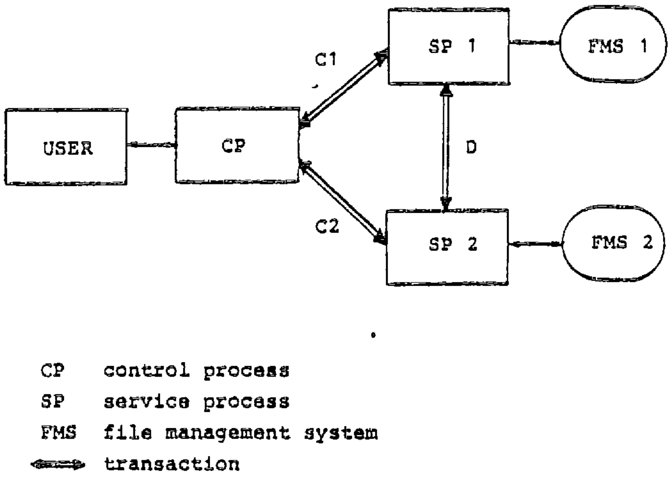


Fig. 1 General FT - model

The problem in this general file transfer model is the synchronisation of the three processes, which may run on three different hosts. This will lead to a considerable expense in file transfer protocol definition.

A simplified File Transfer Model can be derived from the general model by combining one service process with the control process. The combined process will be called master process, the remaining service process is the slave process. The user interface is now sited at the master process. Depending on the transfer direction given by the user, the master process will be either the source or the destination of the transferred file.

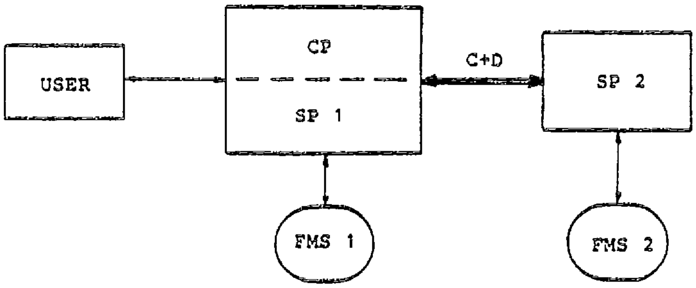
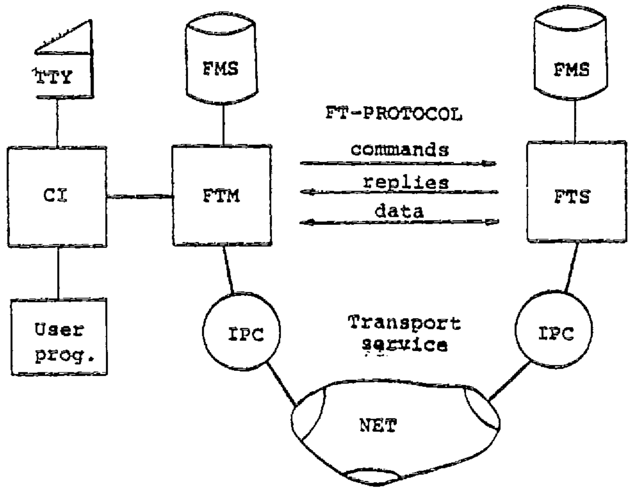


Fig. 2 Simplified FT - model

The three transactions of the general FT model can be multiplexed on one connection between master and slave process, on which the commands, replies and the data elements are exchanged. This approach simplifies synchronisation and restart problems between master and slave process and is the basis of a standard file transfer protocol.

Based on the simplified FT model, the architecture of the File Transfer Service in the HMINET was designed and is shown in the following figure.



FTM	File Transfer Master	FMS	File Management System
FTS	File Transfer Slave	IPC	Inter Process Communication
CI	Command Interpreter		

Fig. 3 File transfer architecture in HMINET

The initiator of a file transfer is always the FT master process, whereby the user decides the transfer direction from master to slave or vice versa.

3. File Transfer Protocol (FTP)

Basic requirements

A standard network file transfer protocol should accomplish the following basic requirements:

- setup and termination of a connection from master to slave process
- identification of the source or destination files

- agreements about file structure and data conversion and additional file manipulation
- data transfer control and error recovery
- extendable for future developments

FT protocol structure in the HMINET

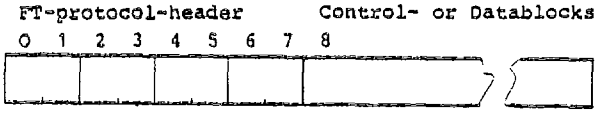
The protocol consists of three phases: association, data transfer and break association phase.

The association phase performs tasks like identification of the access rigths to the remote host, connection setup from master to slave process, establisment of the identity and properties of the file, definition of the transfer direction and the conversions of file structure and data and the specification of additional operations on the transferred file.

The data transfer phase includes the local file operations,the mapping into a virtual netfile format, the actual data transfer and the administrative exchanges required to regulate it. The conversions of the file data and structure are automatically done in this phase by the consumer process.

The break association phase finishes the local file operations, performs the additional functions on the transferred file (spool, job entry etc.), closes the connection and indicates the final state of the transfer.

The format of the protocol elements are shown in figure 4. A protocol element consists of a fixed length header part and a following control- or data part. The parameters in the control part depend on the type of protocol element.



Header definition:

Byte 0	TYPE (protocolelement type)
Byte 1	RCODE (returncode)
Byte 2-3	BCOUNT (blockcount)
Byte 4-5	DATA LENGTH
Byte 6-7	RCOUNT (record count in block)

Fig. 4 FT - PROTOCOL FORMAT

Typical types of protocol elements are:

INIT — initialisation of slave process (via FTDSP) with user access rights parameters

FFDB — function and file descriptor controlblock for identifying the file by filename, -properties and specifying the additional file operations

DATA — file data block in a virtual netfile format

EOF/EOT — end of file or end of transmission declaration

ACK — control element for synchronisation and error detection and recovery

Figure 5 shows a standard scenario of a file transfer from the masters host (initiator) to the slaves host.

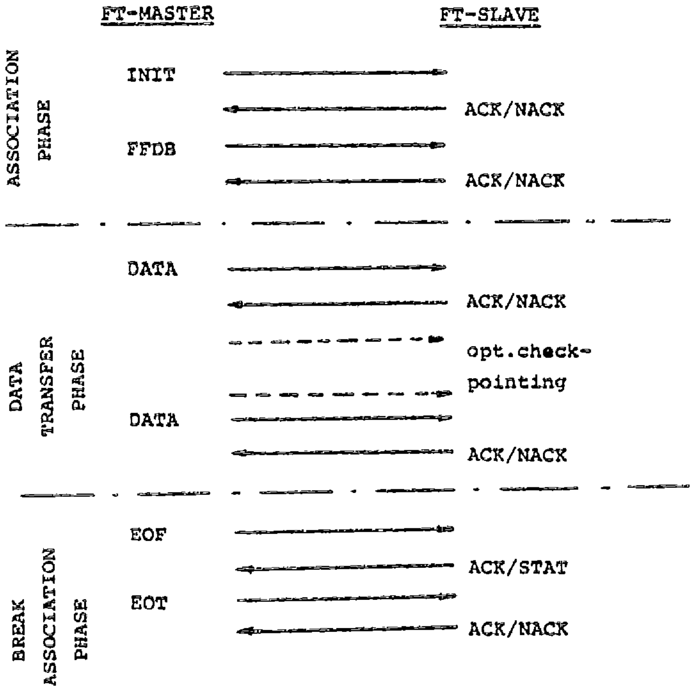


Fig.5 Protocol scenario (Transfer Master→Slave)

Error handling in the FTP

In each of the three protocol phases an error handling and recovery service must be provided. The following table gives an overview of the possible error handling and recovery attempts in each phase.

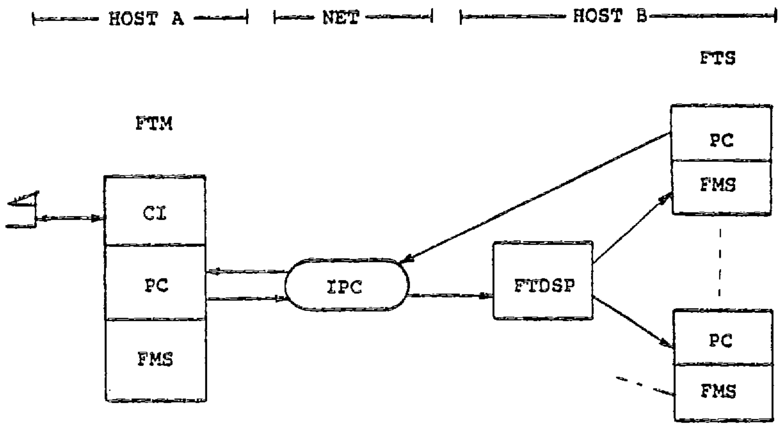
Error in:	Error handling
association phase	break attempt to establish a connection, or cancel connection at remote host errors
data transfer phase	different strategies are available due to the sophistication of the protocol design: 1) termination of session after finishing local and remote file actions 2) retransmission of data elements, if not sucessful see 1) 3) checkpointing and restart facility, if not sucessful see 1)
break assoc. phase	termination of session after finishing local and remote file actions without performing additional functions. (error handling within these additional file operations e.g. print file, are not included in the FT error handling)

4. File Transfer Implementation in the HMINET

On each host computer in the HMINET are three different FT processes running:

- a FT master, which is invoked by the user from a terminal and handels the FT commands, the protocol control and the local file operations
- a FT dispatcher, which is always runing and waiting for incoming FT requests from any FT master in the network. This process request a
- FT slave, which serves the FT protocol and handels the file operation on this host

The data transfer between the parcitipated processes is realised by the inter process communication system (IPC) of the HMINET.



FTM	File Transfer Master	FMS	File Management Service
FTS	File Transfer Slaves	IPC	Inter Process Communication
CI	Command Interpreter	FTDSP	FTS Dispatcher
PC	Protocol Control		

Fig. 6 FT Implementation in the HMINET

The described implementation considers some special FT problems:

Accounting and privacy

The FT-service demanded by a user must be accounted as far as possible to his credit. This problem is solved in the HMI-FT by the FTDSP. As mentioned above this process requests a slave process under the users account and access rigths. Therefore accounting is done by the standard account systems of the hosts on which the master and slave processes are running. The disadvantage of this procedure is that the user must have a seperate access permission to all machines he wants to use for FT.

An advantage of this approach is that the user can only get access to files marked with his own user identification. The problems of privacy and security are reduced to the level of the standard file management protection in the participated computers.

File name conventions

To access the files, different file name conventions must be considered and known by the user. A solution of this problem would be the definition of a netwide standard file name convention. But this approach brings some difficulties by mapping standard file names into local FMS conventions and the users must be familiar with the local and the netstandard file conventions. In the HMINET FT protocol the conventions of the accessed host FMS are used.

Handling of file duplicates

It may happen that on a computer to which a file will be transferred another file exists with an identical name. It depends on the host FMS whether the already existing file is superseded by the new one, or a new file version is created. In each case of file transfer the copied file is treated as a new file which is independent from the source file.

5. Functional Capabilities

The FT in HMINET offers a set of functions which are divided in three groups: data conversion, file structure conversion and additional file operations.

a) Data conversions

This function is automatically done in the data transfer phase if the user dont want to transfer the file in transparent mode.
Conversion is available for:

text files — characters are converted from or to ASCII/ EBCDIC code

binary files — data formats Integes*2, Integer*4, Real*4, Real*8 are converted due to the number representations of the involved hosts (no mixed data files can be converted)

graphic files — the standard graphic files of the HMI GRAFIX system can be automatically converted from and to each host representations

b) File structure conversion

This function relates to the transferred file and integrates it in the destination FMS. Due to the possibilities of the destination FMS, the following file structures and organisations can be selected: sequential-, indexsequential-, block access, contiguous organisation.

c) Additional file operations

These operations on the file are performed after the transfer and are available for both the source and the destination file:

PR	print the file
DE	delete the file
SP	spool the file (PR+DE)
PU	punch the file
RJ	enter the file as a job in the job entry queue (only for destination file)

6. Command-Structure

The user interface of the HMI FT is organised in the manner of a two level command language. In the first level, the remote host access identification is to declare. In the second level, the source and destination file identification and the additional functions are to be specified.

Level 1:

/KEYWORD   remote host access identification

Level 2:

outdescriptor/local function -- indescriptor/loc.function//global function

outdescriptor: destinationhost,-device,-catalog,-filename,-type
indescriptor: sourcehost,-device,-catalog,-filename,-type (some parameters are optional)
local function: file structure definition for the destination file and additional operations for destination and source file (see Chapter 5)
global function: file data conversion definition (see Chapter 5)

Example:

The FT master process prompts with the FIP symbol for command input.

```
FIP  /LOGON MHB,ACCTXYZ,PASSW'ABC'
FIP  DV55  ABC.LST.1/IS/PR = DK1: 1,1 ALFA.FTN/DE//TX
```

The first command line defines the user identification, the accounting, and the user password of the remote host system.
By the second command line, the file ALFA.FTN on device DK1 under catalog 1,1 will be transferred from local host to the remote host DV55 and stored as an indexsequential file on a public volume in the user catalog with the filename ABC:LST. During transfer, a character code conversion (ASCCI- EBCDIC) is done. After transfer the local file ALFA.FTN is deleted and the copied file is additionally printed.

Literature

1 Crocker S.D.,et al
Function Oriented Protocols for ARPA Network
AFIPS SJCC 1972

2 Neigus N.
File Transfer Protocol, NIC 17759
ARPA RFC 542, July 73

3 Heinze W.,Butscher B.
File Transfer in the HMI Computer Network
3. Europ.Network User's Workshop, IIASA April 77

4 Gien M.
Proposal for a Standard File Transfer Protocol
EIN/IRIA/77/4

5 A Network Independent File Transfer Protocol
by High Level Protocol Group, c/o University
of Cambridge, December 77

6 Schicker P.,Duenki A.,Baechi W.
Bulk Transfer Function
EIN/ZHR/75/20

7 Belloni A.,Bozetti M.,Le Moli G.
A Proposal for a Bulk Transfer Function
EIN/CREI/77/1+3

8 Butscher B.,L.-Bauerfeld W.,Popescu-Zeletin R.
Data Access in Heterogeneous Computer Networks
Proceedings GI Fachtagung: Datenbanken in Rechnernetzen mit Kleinrechnern, Karlsruhe April 78

9 Strack-Zimmermann H.,Schrödter H.D.
The Hahn-Meitner-Institut Computer Network
Proceedings GI Fachtagung: Rechnernetze,
Aachen März 76

10 Butscher B.,Irsigler R.,Vogt F.
Interprocess Communication in a Resource Sharing Network
Proceedings GI Fachtagung: Rechnernetze,
Aachen März 76