Tool Kit PRO/DATATRIEVE Documentation Supplement

AA-CK37A-TH

September, 1984

This manual provides information on using components of PRO/DATATRIEVE to develop PRO/DATATRIEVE applications for the Professional hard disk computer system.

OPERATING SYSTEM AND VERSION:	P/OS	V2.0 or later
	VMS	V3.2 or later
	RSX-11M	V4.1 or later
	RSX-11M-PLUS	V2.1 or later
SOFTWARE VERSION:	Professional Host Tool Kit	V2.0
	PRO/Tool Kit	V2.0
	DATATRIEVE-11	V3.0
	PRO/DATATRIEVE	V2.0

digital equipment corporation, maynard, massachusetts

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The information and drawings herein are the property of Digital Equipment Corporation and shall not be reproduced or copied or used in whole or in part as the basis for the manufacture or sale of items without written permission.

The following are trademarks of Digital Equipment Corporation:

CTI BUS	MASSBUS	PROSE PLUS
DEC	PDP	Rainbow
DECmate	P/OS	RSTS
DECsystem-10	PRO/BASIC	RSX
DECSYSTEM-20	PRO/Communications	Tool Kit
DECUS	Professional	UNIBUS
DECwriter	PRO/FMS	VAX
DIBOL	PRO/RDT	VMS
digital™	PRO/RMS	VT
	PROSE	Work Processor

Contents

How to Use This Manual

This manual explains how to use Tool Kit PRO/DATATRIEVE components to create PRO/DATATRIEVE applications for the Professional hard disk computer system. It is one of seven manuals included in the Tool Kit PRO/DATATRIEVE documentation set.

Intended Audience

This manual assumes that:

- You have experience with PRO/DATATRIEVE or DATATRIEVE-11
- You have experience using the PRO/Tool Kit or the Professional Host Tool Kit
- You are familiar with your Professional P/OS operating system and user interface

If you intend to use the callable components of PRO/DATATRIEVE, you should also be familiar with a Tool Kit programming language such as BASIC-PLUS-2, PASCAL, COBOL, or FORTRAN.

Structure

This document contains three chapters, three appendixes, an index, and a master index for all manuals in the Tool Kit PRO/DATATRIEVE documentation set:

Chapter 1	Describes the components of Tool Kit PRO/DATATRIEVE and differences between Tool Kit PRO/DATATRIEVE and DATATRIEVE-11.
Chapter 2	Explains the Tool Kit PRO/DATATRIEVE development process.
Chapter 3	Explains how to use the callable components of Tool Kit PRO/DATATRIEVE.
Appendix A	Contains a documentation directory.
Appendix B	Contains a sample PRO/DATATRIEVE application.
Appendix C	Contains a sample definition of the DATATRIEVE Access Block in PASCAL.

Related Manuals

PRO/DATATRIEVE is derived from DATATRIEVE-11, so the following manuals are included in the Tool Kit PRO/DATATRIEVE documentation set to supply usage and reference information:

- *Introduction to DATATRIEVE-11*
- *DATATRIEVE-11 User's Guide*
- *DATATRIEVE-11 Guide to Writing Reports*
- *DATATRIEVE-11 Call Interface Manual*
- *DATATRIEVE-11 Reference Manual*
- *DATATRIEVE-11 Pocket Guide*

In the course of developing PRO/DATATRIEVE applications, you may need to refer to other documentation:

- For information about the implementation of function keys in PRO/DATATRIEVE, refer to Chapter 2 of *PRO/DATATRIEVE for Beginners*.
- For information about the Professional and Tool Kit environments, refer to Tool Kit documentation. Your Tool Kit user's guide is the primary resource for all application development activities and is an excellent starting point.
- For information about programming languages, refer to the language-specific documentation.

Conventions

- Programming examples appear in this manual in a dot matrix typeface. PRO/DATATRIEVE or program output lines displayed on your Professional are printed in black. Commands and statements you type on your keyboard are printed in red.
- Brackets ([ ]) are required syntax in directory specifications.
- The term "Professional" refers to the Professional 300 series computer equipped with a hard disk.
- The term "PRO/Tool Kit" refers to the Professional Developer's Tool Kit running on a Professional.
- The term "Host Tool Kit" refers to the Professional Developer's Tool Kit running on a VMS, RSX-11M, or RSX-11M-PLUS system.
- The term "Tool Kit" refers to both the PRO/Tool Kit and the Host tool Kit.
- Language names like "BASIC-PLUS-2," "COBOL," and "FORTRAN" refer to both the native and host versions of these Tool Kit languages.

Components and Features of Tool Kit PRO/DATATRIEVE 1

Tool Kit PRO/DATATRIEVE is a documentation-only option that explains how to:

- Build end-user applications that use interactive PRO/DATATRIEVE
- Use the PRO/DATATRIEVE V2.0 Remote Call Interface, Local Call Interface, and callable PRO/DATATRIEVE to create distributed data management applications

The components of Tool Kit PRO/DATATRIEVE applications are:

- Interactive PRO/DATATRIEVE

  The task image (PRODTR.TSK) that you access from your Professional when you select PRO/DATATRIEVE from a menu. You can customize interactive PRO/DATATRIEVE by using the PRO/DATATRIEVE logical name capability and building an application end users can install on a menu.

- Callable PRO/DATATRIEVE

  The callable PRO/DATATRIEVE task image (PROCALDTR.TSK) that the Local Call Interface accesses. Interactive PRO/DATATRIEVE and callable PRO/DATATRIEVE are both part of PRO/DATATRIEVE Version 2.0.

- The PRO/DATATRIEVE Local and Remote Call Interfaces

  Routines in the PRO/DATATRIEVE Call Interface Object Module Library, DTCLIB.OLB, that let programs written in Tool Kit languages call PRO/DATATRIEVE, DATATRIEVE–11, VAX DATATRIEVE, or DATATRIEVE-20. DTCLIB.OLB is distributed with the PRO/Tool Kit Version 2.0 and the Host Tool Kit Version 2.0.

- PRO/RDT (Remote Data Transfer)

  An information management tool that works with PRO/DATATRIEVE Version 2.0 to let you extract selected data from DATATRIEVE domains on other computers. PRO/RDT menus prompt you for the information needed to access and retrieve the DATATRIEVE data you want.

- PRO/DDMF (the Distributed Data Manipulation Facility)

  The PRO/DATATRIEVE Distributed Server that lets users on other DECnet nodes access PRO/DATATRIEVE data files and dictionaries on your Professional. PRO/DDMF substitutes an interface to DECnet for the interactive PRO/DATATRIEVE interface. PRO/DDMF is distributed with PRO/RDT Version 1.0. See the *PRO/RDT Handbook* for more information on PRO/RDT and PRO/DDMF.

By using these components in different combinations, you can perform many different data management activities·

- Using interactive PRO/DATATRIEVE, you can perform ad hoc data management functions such as organizing and reporting on data, or you can develop customized applications as described in Chapter 2 of this manual.
- Using PRO/RDT and PRO/DATATRIEVE, you can extract DATATRIEVE information from a remote host and manipulate it on your Professional to create the reports you need for your daily activities.
- Using PRO/DECnet, the Remote Call Interface, and a remote DDMF, your program can call PRO/DATATRIEVE on another Professional, DATATRIEVE-11 on a PDP-11 system, DATATRIEVE-20 on a DECSYSTEM-20, or VAX DATATRIEVE on a VMS system.
- Using the Local Call Interface and callable PRO/DATATRIEVE, your program can call PRO/DATATRIEVE on your own Professional.

Interactive PRO/DATATRIEVE

When you select PRO/DATATRIEVE from a P/OS menu, you are running PRODTR.TSK, the interactive PRO/DATATRIEVE task image. This program accepts PRO/DATATRIEVE commands as you type them and uses your Professional video screen as the default output device.

By using PRO/DATATRIEVE's logical name capability to identify application files and devices, you can customize the PRO/DATATRIEVE interface for end-user applications and integrate your application with the PRO/DATATRIEVE task image and message file. End users can then install your layered application and select it from a P/OS menu.

Chapter 2 describes how to create layered PRO/DATATRIEVE applications.

PRO/DDMF

PRO/DDMF, the Distributed Data Manipulation Facility, is distributed with PRO/RDT Version 1.0 (Remote Data Transfer). It is a server program, DDMF.TSK, that substitutes for PRO/DATATRIEVE when a program on a remote node calls PRO/DATATRIEVE. PRO/DDMF executes commands passed by the calling program and passes results back to the calling program.

Note

If your program calls PRO/DATATRIEVE on a remote Professional, that Professional must have PRO/DECnet software and PRO/RDT installed.

The PRO/DATATRIEVE Call Interfaces

The PRO/DATATRIEVE call interfaces let you write Tool Kit language programs that call PRO/DATATRIEVE on your Professional (the Local Call Interface) or PRO/DATATRIEVE, DATATRIEVE-11, VAX DATATRIEVE, or DATATRIEVE-20 on a remote node (the Remote Call Interface).

Note

You must have PRO/DECnet software installed on your Professional before you can use the Remote Call Interface. PRO/DECnet is not required to use the Local Call Interface.

The DTCLIB.OLB Object Module Library shipped with the PRO/Tool Kit and Host Tool Kit contains routines necessary for using the Remote and Local Call Interfaces. To use either call interface, you include calls in your program to subroutines contained in DTCLIB. When you task build the program, you link your program with DTCLIB and specifically with either the Remote Call Interface module, NCPOS, or the Local Call Interface module, LCPOS:

- NCPOS sends commands to and receives information from the DDMF task on the remote node.
- LCPOS sends commands to and receives information from the callable PRO/DATATRIEVE task on your Professional.

Chapter 3 in this manual describes how to link your program with DTCLIB routines.

When you run a program that calls DATATRIEVE on a remote node, the following tasks are active:

- The program linked to DTCLIB.OLB routines
- A DDMF task on the remote node

When you run a program that calls PRO/DATATRIEVE on your own Professional, the following tasks are active:

- The program linked to DTCLIB.OLB routines
- Callable PRO/DATATRIEVE (PROCALDTR.TSK) on your Professional

A DDMF task on a remote node and callable PRO/DATATRIEVE on your Professional serve the same functions but for different call interfaces:

- DDMF services calls to DATATRIEVE from programs running on other DECnet nodes.
- PROCALDTR.TSK services calls to PRO/DATATRIEVE from programs running on the local Professional.

The Remote Call Interface uses PRO/DECnet software to access a remote node. It then uses the DDMF task on the remote node to access PRO/DATATRIEVE data files and dictionaries. Figure 1-1 shows how calling programs access data files and DATATRIEVE files and dictionaries on remote systems.

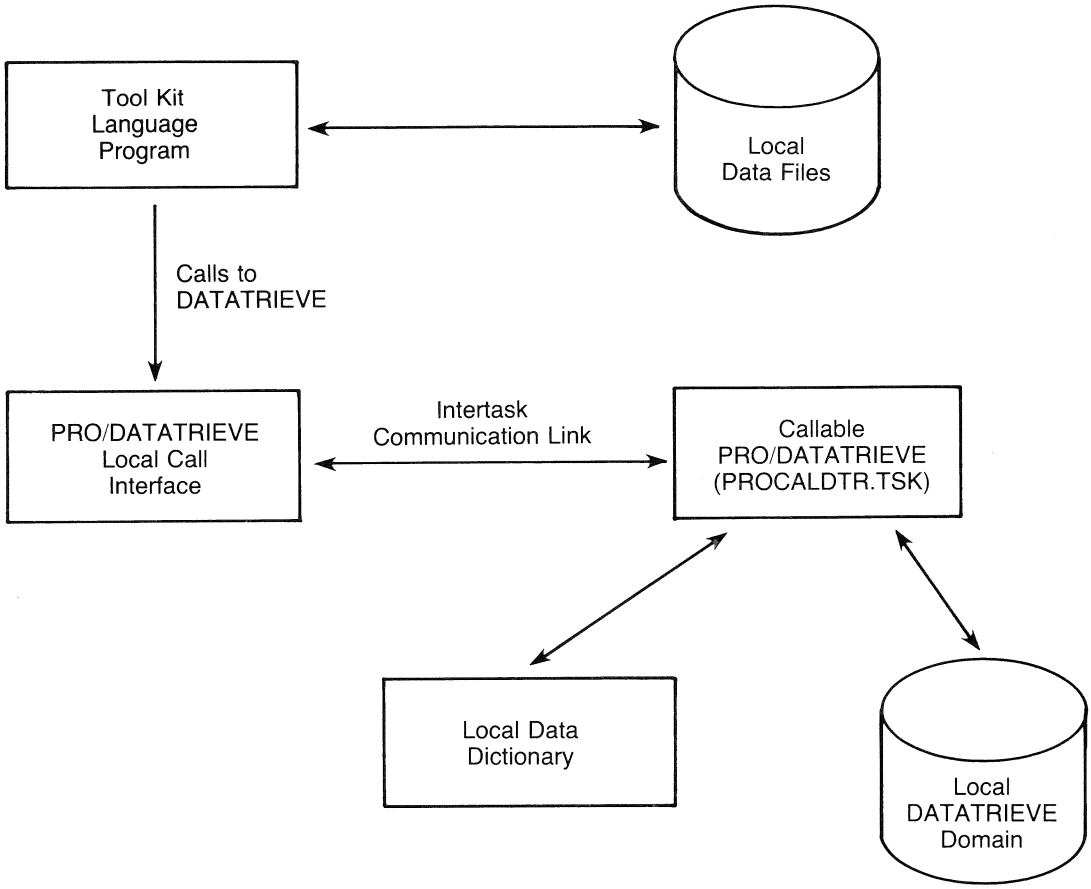

Figure 1-1: PRO/DATATRIEVE Remote Call Interface

The Local Call Interface uses PROCALDTR.TSK to access PRO/DATATRIEVE data files and dictionaries on the local Professional system. Figure 1-2 shows how calling programs access data files and DATATRIEVE files and dictionaries on local systems.

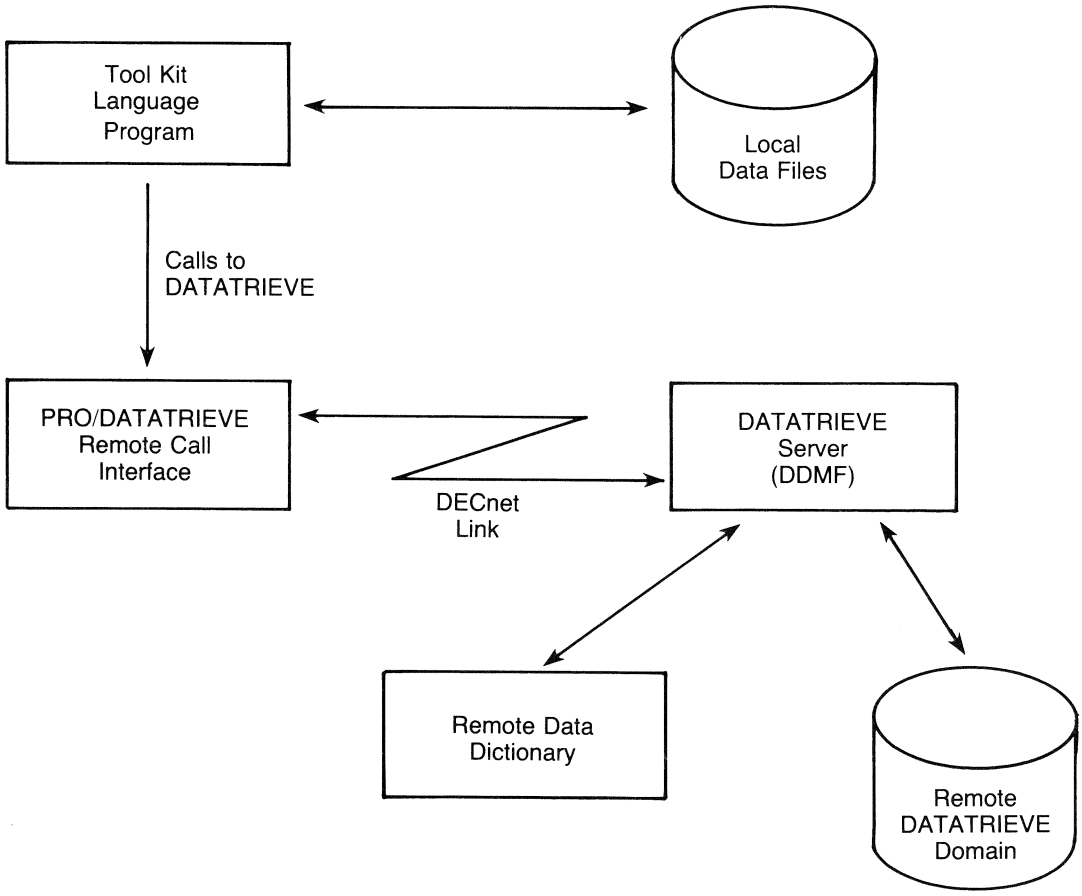

Figure 1-2: PRO/DATATRIEVE Local Call Interface

PRO/DATATRIEVE Features

PRO/DATATRIEVE and DATATRIEVE-11 are essentially the same, but the following differences do exist:

- PRO/DATATRIEVE has an extended logical name capability; DATATRIEVE-11 does not.
- PRO/DATATRIEVE recognizes function keys; DATATRIEVE-11 does not.
- The PRO/DATATRIEVE Remote Call Interface and Local Call Interface are separate and are activated by different Call Interface Object Module Library routines; the DATATRIEVE-11 Remote Call Interface functions as the local call interface through a DECnet loopback link.

Because of language differences, applications developed with VAX DATATRIEVE may require modification to work with PRO/DATATRIEVE.

For example, the default data alignment for VAX DATATRIEVE is MAJOR-MINOR, while the default for PRO/DATATRIEVE and DATATRIEVE-11 is LEFT-RIGHT. Consequently, the record definition and the actual records may not be the same size. To ensure that PRO/DATATRIEVE record definitions map to the actual records, add an ALLOCATION IS MAJOR-MINOR clause to the record definition.

For best results, use PRO/DATATRIEVE or DATATRIEVE to develop PRO/DATATRIEVE applications.

PRO/DATATRIEVE Logical Name Capability

With PRO/DATATRIEVE's logical name capability, you can assign logical names to devices and files in the application installation file. You can also use logical names in PRO/DATATRIEVE statements and commands wherever you can use a file specification or device name. See the section on logical names in Chapter 2 for more information.

PRO/DATATRIEVE Function Keys

You can use the MAIN SCREEN, DO, CANCEL, HELP, and EXIT function keys and the INTERRUPT/DO key sequence when using PRO/DATATRIEVE much as you can when using Professional menus. If you press other function keys on the Professional keyboard while using PRO/DATATRIEVE, you hear a beeping noise. This indicates that the key does not perform a specific function in PRO/DATATRIEVE.

Function keys perform in the following ways:

- DO – Executes a PRO/DATATRIEVE command or statement or ends an input line. Using the DO key is identical to using the RETURN key.
- MAIN SCREEN – When pressed in response to the DTR> prompt, leaves PRO/DATATRIEVE and returns to the Main Menu. Pressing MAIN SCREEN while PRO/DATATRIEVE is performing a calculation or executing a command or statement returns you to the Main Menu after PRO/DATATRIEVE finishes its current operation. In Guide Mode, pressing the MAIN SCREEN key returns you to the DTR> prompt.
- CANCEL – Cancels a command or statement. PRO/DATATRIEVE ignores the command or statement and prompts again for input. Using the CANCEL key is identical to using CTRL/U.
- HELP – When pressed in response to the DTR> prompt or after you type HELP topic, displays the introductory HELP frame. To get HELP on a topic, you must use the HELP command and press DO or RETURN. In Guide Mode, the HELP key displays a list of possible responses. In ADT, the HELP key provides information on ADT questions.

- EXIT – The effect of the EXIT key depends on what PRO/DATATRIEVE is doing:
  - Pressing EXIT in response to the DTR> prompt returns you to the Main Menu. In this instance, pressing EXIT is the same as pressing MAIN SCREEN or typing EXIT and pressing DO.
  - Pressing EXIT in response to any other PRO/DATATRIEVE prompt, including Guide Mode, returns you to the DTR> prompt.
  - Pressing EXIT while PRO/DATATRIEVE is calculating or executing a command or statement returns you to the Main Menu *after* PRO/DATATRIEVE finishes its current operation.

  Using the EXIT key is identical to using CTRL/Z.
- INTERRUPT/DO – The effect of an INTERRUPT/DO sequence also depends on what PRO/DATATRIEVE is doing:
  - Pressing INTERRUPT/DO in response to *any* PRO/DATATRIEVE prompt returns you to the Main Menu.
  - Pressing INTERRUPT/DO while PRO/DATATRIEVE is calculating or executing a command returns you to the DTR> prompt.

    Using INTERRUPT/DO is identical to using CTRL/C.

See Chapter 2 in *PRO/DATATRIEVE for Beginners* for more information on PRO/DATATRIEVE function keys.

The PRO/DATATRIEVE Call Interfaces

Tool Kit PRO/DATATRIEVE has separate Remote and Local Call Interfaces. The DATATRIEVE-11 Remote Call Interface, as described in the *DATATRIEVE-11 Call Interface Manual*, includes the local call interface.

The modules NC11M and NCRSTS in the DATATRIEVE-11 Call Interface Object Module Library are replaced in the PRO/DATATRIEVE Call Interface Object Module Library by NCPOS and LCPOS. NCPOS activates the Remote Call Interface and LCPOS activates the Local Call Interface.

Developing PRO/DATATRIEVE Applications 2

This chapter supplements the *DATATRIEVE-11 User's Guide* and is intended for experienced users of PRO/DATATRIEVE or DATATRIEVE-11. For introductory information, see *Introduction to DATATRIEVE-11* or *PRO/DATATRIEVE for Beginners*. For syntax and usage rules for PRO/DATATRIEVE statements and commands, refer to the *DATATRIEVE-11 Reference Manual.*

This chapter describes the development process for creating PRO/DATATRIEVE applications that an end user can install on a Professional and select from a menu. Chapter 3 describes the additional steps involved in developing Tool Kit language applications that use the PRO/DATATRIEVE Remote or Local Call Interface.

Developing a PRO/DATATRIEVE application consists of the following steps:

1. Writing the PRO/DATATRIEVE application files. The next section describes the types of application files a PRO/DATATRIEVE application uses.
2. Using logical names to identify application files and devices. The section entitled "Using Logical Names" describes PRO/DATATRIEVE's logical name capability.
3. Writing the application installation file. This file identifies your application files and installs the PRO/DATATRIEVE task image. It also assigns logical names to your application dictionary and may assign logical names to other application files and devices. See the section in this chapter entitled "Writing the Application Installation File."
4. Installing, executing, and debugging the application on a Professional. Your Tool Kit user's guide explains how to install applications. After you install an application on your Professional, select it from a menu to identify problems. If you find errors in the application, edit the application files and retry the application. Continue this process until the application works the way you want.
5. Compressing the application dictionary. The section entitled "Compressing the Application Dictionary" describes this step.
6. Creating distribution kits with the Application Diskette Builder. This step is described in your Tool Kit user's guide.

Creating Application Files

The minimum PRO/DATATRIEVE application consists of a dictionary file that contains the application definitions. It may also include other dictionary files, command files, data files, and an initialization file that sets dictionary and environment defaults. The PRO/DATATRIEVE V2.0 application, for example, consists of the following files:

PRODTR.TSK	The PRO/DATATRIEVE task image
PROCALDTR.TSK	The callable PRO/DATATRIEVE task image
PRODTR.MSG	The PRO/DATATRIEVE message file
PRODTR.DIC	The default data dictionary
PRODTR.INS	The PRO/DATATRIEVE installation file
EMPLOYEES.DAT	A sample data file

Unless you are developing a Tool Kit language application that calls PRO/DATATRIEVE (as described in the next chapter), your application does not need its own task (.TSK) or message (.MSG) file, because it uses the PRO/DATATRIEVE task image and message file.

See the *DATATRIEVE-11 User's Guide* and *DATATRIEVE-11 Reference Manual* included in this documentation set for additional information on developing applications.

All application file names must conform to the following P/OS format for naming files:

```
volumename:[directory]filename.typ
```

Volume Name (volumename:)

A volume name can consist of from 1 to 12 alphanumeric characters and should not duplicate any of the following names reserved for P/OS:

BIGDISK	DZ2	TT
BIGVOLUME	LB	TT1
DISKETTE1	LP	TT2
DISKETTE2	SY	XK
DW1	SYSDISK	XT
DZ1	TI	

If you omit a volume name, P/OS uses the current default.

Directory ([directory])

A directory name can consist of from one to nine alphanumeric characters. It must be enclosed in square brackets ([]) or angle brackets (<>). If you omit a directory name, P/OS uses the current default.

File Name (filename)

A file name can consist of from one to nine alphanumeric characters. P/OS converts lowercase letters in file names to uppercase letters. Spaces, punctuation, and other special characters are not valid characters in file names.

File Type (.typ)

You can explicitly specify any three characters you like as a file type. However, if you omit the file type PRO/DATATRIEVE uses the following default file types that describe the internal organization of the file:

Dictionary files	.DIC
Command files	.CMD
Data files	.DAT
Initialization file	.INI

The default for PRO/DATATRIEVE command files created with the EXTRACT command is .CMD. You can override this default when you use the EXTRACT command. The default for data files identified in the DEFINE DOMAIN command and created with the DEFINE FILE command is .DAT. You can override this default by specifying a file type.

See your Tool Kit user's guide for more details on naming P/OS files.

Using Logical Names

Most applications that run on the Professional let you assign logical names only to devices. The PRO/DATATRIEVE logical name facility lets you assign logical names to entire or partial file specifications, with only one restriction: a logical name, if used, must be the first field in the file specification. You cannot, for example, specify a physical device name, like DZ1:, and a logical directory name, like MAIL$DIR in the same file specification.

A PRO/DATATRIEVE logical name may replace any of the following:

- A device name (BIGVOLUME:)
- A device and directory name (BIGVOLUME:[ZZDTR])
- A device, directory, and file name ((BIGVOLUME:[ZZDTR]PRODTR)
- A full file specification (BIGVOLUME:[ZZDTR]PRODTR.TSK)

You assign logical names in the application installation file with the ASSIGN LOGICAL command, as described in the section entitled "Writing the Application Installation File."

In addition to logical names you assign, the PRO/Dispatcher assigns the logical name APPL$DIR to the directory on the hard disk where your application files reside. The P/OS Install program creates application directories in the form [ZZAPnnnnn], where an integer value in the form nnnnn is assigned. This integer value varies from one Professional to another, so you can use the logical name APPL$DIR to identify files that reside in your application's ZZAP directory.

Because application files reside in one directory and the PRO/DATATRIEVE task image and message file reside in another directory ([ZZDTR]), the PRO/DATATRIEVE application uses logical names to locate the files it needs to operate:

- DTR$DIC Identifies a PRO/DATATRIEVE application dictionary file. You assign this logical name to your application's dictionary file.
- DTR$INI Identifies an optional initialization file that contains PRO/DATATRIEVE commands and statements. PRO/DATATRIEVE accesses the file identified by this logical name when it starts, before processing any terminal input.

You can assign logical names to application files and to the Professional video screen in the application installation file in this way:

```
Assign Logical DTR$DIC "APPL$DIR:INVENTORY.DIC"
Assign Logical DTR$INI "APPL$DIR:INVENTORY.CMD"
Assign Logical SCREEN "TI:"
```

Notice that the first two lines in the example assign the logical names DTR$DIC and DTR$INI in terms of the logical name assigned to the application directory, APPL$DIR. The first line assigns the logical name DTR$DIC to an application dictionary, INVENTORY.DIC. This causes PRO/DATATRIEVE to use that dictionary instead of the standard dictionary, PRODTR.DIC. The second line in the example assigns the logical name DTR$INI to an application command file, INVENTORY.CMD. PRO/DATATRIEVE executes this command file before displaying the first DTR> prompt.

The third line in the example assigns the logical name SCREEN to TI:, the device name of the Professional video screen. Your application could use this logical name in statements like the following:

```
REPORT CURRENT SORTED BY SALESMAN ON *,"File name or SCREEN"
```

Because SCREEN is defined as TI:, PRO/DATATRIEVE displays the report on the video screen when the user types SCREEN in response to the prompting value expression.

In addition to using logical names in the application installation file, you can also use APPL$DIR when defining application data files:

```
DEFINE DOMAIN WIDGETS USING WIDGETS-REC ON APPL$DIR:WIDGETS.DAT;
DEFINE DOMAIN SALESMEN USING SALESMEN-REC ON APPL$DIR:SALESMEN.DAT;
```

These domain definitions make it possible for PRO/DATATRIEVE to access the application data files, WIDGETS.DAT and SALESMEN.DAT, from the directory where the end user installed your application. Note, however, that the domains can be accessed only from the application.

Writing the Application Installation File

The application installation file:

- Identifies all your application files
- Identifies the PRO/DATATRIEVE dictionary
- Installs the PRO/DATATRIEVE task image
- Runs PRO/DATATRIEVE

If you plan to use the Fast Install utility to install your application without first building application diskettes, the installation file must have the same name as the directory in which it resides. It must also have a file type of .INS. See your Tool Kit user's guide for more information about Fast Install requirements.

A PRO/DATATRIEVE application installation file needs only the following elements:

- **NAME "menu name"**

  You can specify only one NAME command.

- **FILE filename/option**

  You must identify each application file your application uses in a FILE command and specify an option (KEEP or DELETE) that determines whether the file is kept or deleted when the application is removed. See your Tool Kit documentation for information on KEEP and DELETE.

  Do not include a FILE command for the PRO/DATATRIEVE task image, the PRO/DATATRIEVE message file, or the application installation file.

- **INSTALL [ZZDTR]PRODTR.TSK/TASK**

  This line installs the PRO/DATATRIEVE task image. You must include the /TASK value to indicate that the file is an executable task image.

- **ASSIGN LOGICAL logical name "string"**

  You must include ASSIGN LOGICAL commands to assign the logical name DTR$DIC to your application dictionary or the default PRO/DATATRIEVE dictionary, [ZZDTR]PRODTR.DIC. You may include as many more ASSIGN LOGICAL commands as your application needs. See the section on logical names for information on assigning and using logical names.

- **RUN DTR**

  You can specify only one RUN command. For applications layered on PRO/DATATRIEVE, RUN DTR tells the PRO/Dispatcher to execute PRODTR.TSK when the user selects the application from a P/OS menu.

All lines must appear in the order shown in the previous list and in the following sample installation file:

```
! This is INVENTORY.INS, the installation
! file for the INVENTORY application
Name "INVENTORY"
File INVENTORY.DIC/DELETE
File INVENTORY.CMD/DELETE
File WIDGETS.DAT/DELETE
File SALESMEN.DAT/DELETE
Install [ZZDTR]PRODTR.TSK/TASK
Assign Logical DTR$DIC "APPL$DIR:INVENTORY.DIC"
Assign Logical DTR$INI "APPL$DIR:INVENTORY.CMD"
Assign Logical SCREEN "TI:"
RUN DTR
```

This application installation file lists four application files and marks all four for deletion upon removal of the application. The INSTALL line installs the PRO/DATATRIEVE task image. The ASSIGN LOGICAL lines assign logical names to the application dictionary, an initialization command file, and the video screen. The RUN DTR line tells the PRO/Dispatcher to execute the PRO/DATATRIEVE task image.

Compressing the Application Dictionary

After you have thoroughly debugged an application, you should compress the application dictionary so that it takes up as little disk space as possible. Compressing a dictionary is particularly important if you have made extensive changes while refining the application.

To compress a PRO/DATATRIEVE dictionary, take the following steps:

1. Select the application or PRO/DATATRIEVE from the Main Menu.
2. Set your current dictionary to the application dictionary.
3. Use the SHOW ALL command to see the names of all definitions stored in the application dictionary.
4. Use the EXTRACT command to write all dictionary definitions to a command file:

```
DTR> SET DICTIONARY INVENTORY
DTR> SHOW ALL
Domains:
        CUSTOMERS       CUST_VIEW       EMPLOYEES       INVENTORY
        SALES_VIEW      SUPPLIERS       TRANSACTIONS    TRANS_BY_ITEM
Records:
        CUSTOMERS_REC   EMPLOYEES_REC   INVENTORY_REC   SUPPLIERS_REC
        TRANS_REC
Procedures:
        CUR_REPORT      CUST_REP        INV_STORE       MAIL_FORMAT
        MAIL_REPORT     TRANS_WRITE     UPDATE_MASTER   WRITE_PRICE
Tables:
        AREA_TABLE      COMPANIES
The current dictionary is DW1:[INVENTORY]INVENTORY.DIC;1
No established collections
No ready domains
```

(continued on next page)

```
DTR> EXTRACT ON NEWDIC.CMD CUSTOMERS, CUST_VIEW, EMPLOYEES,
CON> INVENTORY, SALES_VIEW, SUPPLIERS, TRANSACTIONS, TRANS_BY_ITEM,
CON> CUSTOMERS_REC, EMPLOYEES_REC, INVENTORY_REC, SUPPLIERS_REC,
CON> TRANS_REC, CUR_REPORT, CUST_REP, INV_STORE, MAIL_FORMAT,
CON> MAIL_REPORT, TRANS_WRITE, UPDATE_MASTER, WRITE_PRICE,
CON> AREA_TABLE, COMPANIES
DTR>
```

5. Use the DEFINE DICTIONARY command to define the application dictionary again:

```
DTR> DEFINE DICTIONARY [INVENTORY]INVENTORY.DIC
DTR> SHOW DICTIONARY
The current dictionary is DW1:[INVENTORY]INVENTORY.DIC;2
DTR>
```

6. Invoke the command file NEWDIC, created with the EXTRACT command in Step 3, to store all your application definitions in the new dictionary:

```
DTR> @NEWDIC
```

7. Type SHOW ALL to confirm the successful execution of NEWDIC. Then exit from PRO/DATATRIEVE and delete the old version of INVENTORY.DIC.

Using the PRO/DATATRIEVE Call Interfaces 3

This chapter supplements Chapter 3, "Running Programs that Call DATATRIEVE," in the *DATATRIEVE-11 Call Interface Manual* and is intended for programmers experienced in writing Tool Kit language programs. It describes the steps involved in creating applications that use the callable components of Tool Kit PRO/DATATRIEVE.

Creating an application that calls PRO/DATATRIEVE consists of the following steps:

1. Writing the source language code
2. Compiling the source program
3. Task building the P/OS task image
4. Writing the application installation file
5. Running and debugging the application on a Professional
6. Creating distribution kits

This chapter explains aspects of this development process that are unique to applications that call PRO/DATATRIEVE. For a complete description of each step in the application development process, refer to your Tool Kit user's guide and the appropriate language documentation.

Writing the Source Program

As explained in Chapter 4 of the *DATATRIEVE-11 Call Interface Manual*, you call PRO/DATATRIEVE from a Tool Kit language by taking the following actions:

- Declaring a PRO/DATATRIEVE Access Block (DAB)
- Initializing the PRO/DATATRIEVE interface
- Checking the PRO/DATATRIEVE state to see which routine PRO/DATATRIEVE expects you to call next
- Calling PRO/DATATRIEVE routines

- Handling errors and displaying messages
- Terminating the call interface

See Chapters 4 through 8 and Appendix A in the *DATATRIEVE-11 Call Interface Manual* for information on writing programs that call PRO/DATATRIEVE and for sample BASIC, COBOL, and FORTRAN programs and DAB inclusion files. Appendix C in this manual contains a sample PASCAL DAB inclusion file and a sample PASCAL program.

Compiling the Source Program

Compile your program using the appropriate Tool Kit language compiler or assembler, as described in your language documentation.

Task Building the Application

Before you can task build your application with the Professional Application Builder (PAB), you must:

1. Create a Task Builder command file (.CMD) and overlay descriptor file (.ODL). See the appropriate language documentation and your Tool Kit user's guide for information on creating .CMD and .ODL files.
2. Edit the Task Builder command file to specify the logical unit numbers (LUNs) your task image will use. The next section in this chapter explains how to customize .CMD files for applications that call PRO/DATATRIEVE. See Chapter 3 in the *DATATRIEVE-11 Call Interface Manual* and your Tool Kit user's guide for more general information on logical unit numbers.
3. Edit the overlay descriptor file to include references to the Call Interface library, DTCLIB, so that your program can be linked with Call Interface routines. Another section in this chapter describes the Call Interface Library and routines you must reference in the .ODL file.
4. Task build your application using the Professional Application Builder. See your Tool Kit user's guide for complete information on Professional Application Builder (PAB) commands.

Editing the Task Builder Command File

The Remote Call Interface uses logical units to establish links between DECnet and the calling program. For this reason, the .CMD file for a task that calls DATATRIEVE on a remote node must specify how many logical units the Remote Call Interface needs and exactly which logical unit numbers (LUNs) it can use. Otherwise, the Remote Call Interface may try to use the same LUNs used by the language processor and P/OS user interface services.

The number of logical units the Remote Call Interface requires depends on the number of logical links your program will have active at the same time and is calculated by adding one to the number of DATATRIEVE Access Blocks (DABs) your program uses. Thus, if your program has only one DAB, you need to make sure two logical units are available and to identify which two LUNs the Remote Call Interface will use. See Chapter 4 in the *DATATRIEVE-11 Call Interface Manual* for more information on the DATATRIEVE Access Block.

The Remote Call Interface has a two-word global storage area called LUNMAP. By adding a GBLPAT line to the CMD file, you identify the LUNs available to the Remote Call Interface via LUNMAP. LUNs allocated to P/OS user interface services and the programming language are not available to the Remote Call Interface and should be specified as zeros in the LUNMAP.

For example, this is a .CMD file for a program named SAMPLE that was generated by the BASIC-PLUS-2 BUILD command:

```
SY:SAMPLE/CP/FP=SY:SAMPLE/MP
TASK = SAMPLE
UNITS = 18                      ; decimal          ①
ASG = TI:13:15                  ; decimal          ②
ASG = SY:5:6:7:8:9:10:11:12     ; decimal          ②
EXTTSK =  952
CLSTR = PBFSML,POSRES,RMSRES:RO
EXTSCT = DM$BUF:4540            ; octal
EXTSCT = FL$BUF:4310            ; octal
EXTSCT = HL$BUF:3410            ; octal
EXTSCT = MM$BUF:1000            ; octal
EXTSCT = MN$BUF:4540            ; octal
GBLDEF = HL$LUN:21              ; octal
GBLDEF = MN$LUN:20              ; octal            ②
GBLDEF = MS$LUN:16              ; octal
GBLDEF = TT$EFN:1               ; octal
GBLDEF = TT$LUN:15              ; octal
GBLDEF = WC$LUN:22              ; octal
//
```

Comments have been added to the .CMD file to clarify the lines.

The UNITS line (1) allocates a total of 18 (decimal) logical units to the program.

The ASG and GBLDEF lines (2) indicate that P/OS services and BASIC-PLUS-2 use all but three of these logical units, leaving LUNs 2, 3, and 4, available for the Remote Call Interface.

If your program requires more than three LUNs for the Remote Call Interface, you will have to allocate more logical units for the program by changing the UNITS line. If three LUNs are sufficient, you can simply add the following GBLPAT line to the CMD file:

```
GBLPAT = MAIN:LUNMAP:000016:0; LUNs 2, 3, 4 for the CI
```

This line specifies LUNs 2, 3, and 4 for use by the Remote Call Interface and maps LUNMAP as shown in Figure 3-1.

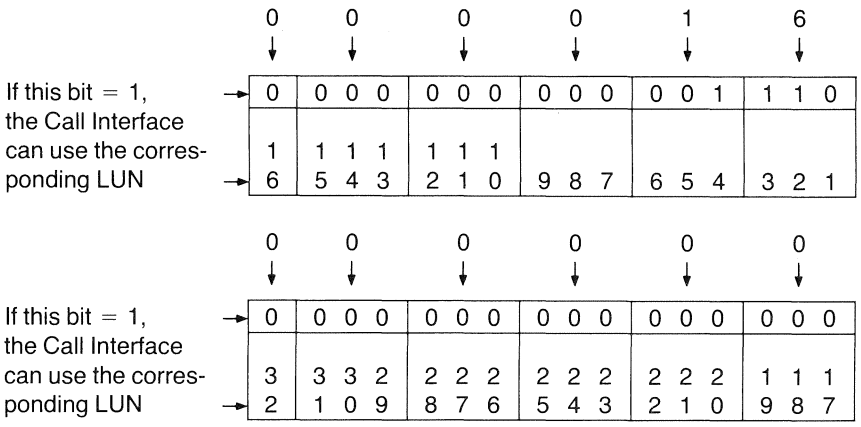

Figure 3-1: LUNMAP Bit Settings

Note

Note that ASG numbers are interpreted as decimal values, while GBLDEF and GBLPAT numbers are interpreted as octal values.

See Chapter 3 in the *DATATRIEVE-11 Call Interface Manual* for more information on LUNMAP.

Editing the Overlay Descriptor (.ODL) File

The .ODL file must reference the Call Interface object module library, DTCLIB.OLB. DTCLIB.OLB is shipped with your Tool Kit software and contains the modules needed to use the Remote and Local Call Interfaces.

The .ODL file must include the following references to DTCLIB.OLB:

- A reference to a particular language module:

  CIBAS – For programs written in BASIC-PLUS-2
  CICOB – For programs written in COBOL
  CIFOR – For programs written in FORTRAN or any other Tool Kit language, such as PASCAL, that uses the FORTRAN method of passing arguments

- A reference to either the remote or local interface module:

  LCPOS – For programs that call PRO/DATATRIEVE on your Professional
  NCPOS – For programs that call PRO/DATATRIEVE, VAX DATATRIEVE, DATATRIEVE-11, or DATATRIEVE-20 on a remote system

- A reference to DTCLIB.OLB as a whole

To link your program with DTCLIB routines, you need to reference DTCLIB in the ROOT line and define it in a .FCTR line in the .ODL file. The general format for a .FCTR line that identifies DTCLIB routines is:

.FCTR LB:[1,5]DTCLIB/LB: { CIBAS / CICOB / CIFOR } : { LCPOS / NCPOS } -LB:[1,5]DTCLIB/LB

An .ODL file generated by the BASIC-PLUS-2 BUILD command, for example, looks like this:

```
        ,ROOT BASIC2-RMSROT-USER,RMSALL
USER:   ,FCTR SY:SAMPLE-LIBR
LIBR:   ,FCTR LB:[1,5]PBFOTS/LB
@LB:[1,5]PBFIC1
@LB:[1,5]RMSRLX
        ,END
```

If the program uses the Remote Call Interface, you would edit the .ODL file as follows:

```
USER:   ,FCTR SY:SAMPLE-LIBR-DTRRCI
DTRRCI: ,FCTR LB:[1,5]DTCLIB/LB:CIBAS:NCPOS-LB:[1,5]DTCLIB/LB
```

CIBAS identifies the language-specific module for BASIC, NCPOS identifies the Remote Call Interface routine, and DTCLIB/LB causes the Task Builder to search DTCLIB for any other modules that are needed.

Creating the Application Installation File

As with all Professional applications, you must write an installation file (filename.INS) before you can install the application. The installation file for a COBOL application that uses the Remote Call Interface looks like this:

```
! Installation file for REMSAMP
Name "PRO/REMSAMP"
File REMSAMP.TSK/Delete
File REMSAMP.MNU/Delete
File REMSAMP.HLP/Delete
Install REMSAMP.TSK/Task
Install [ZZSYS]C81LIB.TSK/Library
Assign MENU REMSAMP.MNU
Assign HELP REMSAMP.HLP RH01
Run REMSAMP
```

Installation files for applications that use the Local Call Interface require an additional line to install the callable PRO/DATATRIEVE task image, PROCALDTR.TSK:

```
Install [ZZDTR]PROCALDTR.TSK/Task
```

Installation files for applications using the Local Call Interface may also assign the logical name CALDTR$INI to an initialization file for the application:

```
Assign Logical CALDTR$INI "REMSAMP.INI"
```

An initialization file for calling PRO/DATATRIEVE locally is optional. There is no default for an initialization file.

Some P/OS services require additional lines in the .INS file. See your Tool Kit user's guide for information specific to the service your program uses and for a complete description of .INS file lines.

Running and Debugging the Application

Use the P/OS menu structure and Fast Install to install your application on a Professional. Fast Install reports the success or failure of the installation process. If the installation process fails, check your .INS file for errors. Note that the .INS file name must be the same as the directory name you give to Fast Install. Correct any errors in the .INS file, rename the file if necessary, and run Fast Install again. Refer to your Tool Kit user's guide for more information on Fast Install.

Once the application has been installed successfully, run and debug it as you would any other Tool Kit language application. Your language documentation describes debugging commands and their use. For information on DATATRIEVE errors, see the *DATATRIEVE-11 Call Interface Manual.*

To help in the debugging process, you can define a log file for PROCALDTR with the DCL ASSIGN command or with the Assign Logical command in the installation file:

```
$ ASSIGN filespec CALDTR$LOG
```

or

```
Assign Logical CALDTR$LOG "filespec"
```

Each time the PROCALDTR task image is invoked, a new version of the log file is created, and you can consult it for information about how the program works.

If you are creating and debugging programs that use the Remote Call Interface to access another Professional, you can define DDMF$LOG on the remote Professional. The log file is created on the remote Professional, and you can consult that file for information about the program. You can also define DDMF$LOG on your own Professional to track remote access by calling programs and PRO/RDT.

Creating Distribution Kits

When your application is error-free, run the Application Diskette Builder (ADB) to create diskettes for distribution. See your Tool Kit user's guide for complete information on ADB commands.

Documentation Directory A

This appendix describes the manuals in the Tool Kit PRO/DATATRIEVE, PRO/RDT, and PRO/DATATRIEVE documentation sets.

PRO/DATATRIEVE Documentation

The PRO/DATATRIEVE documentation set consists of the following manuals and cards:

- *PRO/DATATRIEVE for Beginners*

  Objective

  To introduce PRO/DATATRIEVE and data management concepts to new users.

  Audience

  End users who are not familiar with PRO/DATATRIEVE or with data managemant.

  Summary

  The manual provides an overview of information management terms and concepts, and then takes the user step-by-step through the process of using PRO/DATATRIEVE to manage customer information. Tasks introduced in this manual include defining, storing, retrieving, and modifying data, using procedures and command files, and creating reports.

- *PRO/DATATRIEVE Handbook*

  Objective

  To show how to use PRO/DATATRIEVE to perform advanced data management tasks and to provide practical, task-oriented usage and reference information about PRO/DATATRIEVE.

  Audience

  People familiar with PRO/DATATRIEVE, the basic concepts of data management, the Professional hard disk computer system, and the PROSE editor.

Summary

The first part of this manual contains information on how you can use PRO/DATATRIEVE to perform advanced data management tasks such as establishing context for PRO/DATATRIEVE statements, retrieving information from list fields, and restructuring a data base. The second part contains reference information on PRO/DATATRIEVE commands, statements, and field definition clauses and many explanatory examples. Language elements are grouped in chapters by function.

- *Distributed PRO/DATATRIEVE Reference Card*

Objective

To describe the distributed components of PRO/DATATRIEVE Version 2.0 and related products.

Audience

People who need to know about PRO/DATATRIEVE's distributed capabilities.

Summary

This card summarizes the distributed components of PRO/DATATRIEVE and provides information about the related products and documentation known collectively as Tool Kit PRO/DATATRIEVE.

- *PRO/DATATRIEVE Summary Card*

Objective

To provide quick reference information on the format of PRO/DATATRIEVE commands, statements, clauses, operators, and RSEs.

Audience

Users familiar with PRO/DATATRIEVE who need information on syntax formats.

Summary

This card shows the formats of PRO/DATATRIEVE commands, statements, and clauses, lists and shows examples of PRO/DATATRIEVE operators, and shows the format of an RSE.

PRO/RDT Documentation

PRO/RDT documentation consists of the *PRO/RDT Handbook* and extensive online help.

Objective

To introduce PRO/RDT concepts to new users.

Audience

End users who want to retrieve data from DATATRIEVE domains on a remote computer system.

Summary

The manual explains how to to use PRO/RDT to extract DATATRIEVE data from a Professional hard disk computer system, a VAX/VMS system, a PDP-11 system, or a DECSYSTEM-20. Online help provides information on all PRO/RDT menus, menu options, and forms.

Tool Kit PRO/DATATRIEVE Documentation

The Tool Kit PRO/DATATRIEVE documentation set consists of the following manuals:

- *Tool Kit PRO/DATATRIEVE Documentation Supplement*

  Objective

  To explain how to use Tool Kit PRO/DATATRIEVE components to create PRO/DATATRIEVE applications for the Professional hard disk system.

  Audience

  Programmers familiar with PRO/DATATRIEVE or DATATRIEVE-11, the Professional or Host Tool Kit, a Tool Kit programming language, and the Professional P/OS operating system and user interface.

  Summary

  This manual describes the components of Tool Kit PRO/DATATRIEVE, differences between Tool Kit PRO/DATATRIEVE and DATATRIEVE-11, and the Tool Kit PRO/DATATRIEVE development process, and explains how to use the callable components of Tool Kit PRO/DATATRIEVE.

- *Introduction to DATATRIEVE-11*

  Objective

  To introduce DATATRIEVE-11 to new users.

  Audience

  People who are unfamiliar with DATATRIEVE-11 or PRO/DATATRIEVE.

  Summary

  The manual begins with a brief overview of information management and then, using examples, guides you through basic DATATRIEVE-11 tasks. These include defining, storing, retrieving, and modifying data and using ADT and Guide Mode.

- *DATATRIEVE-11 User's Guide*

  Objective

  To describe the interactive use of DATATRIEVE-11.

  Audience

  People who:

  – Are familiar with the material covered in *PRO/DATATRIEVE for Beginners* or the *Introduction to DATATRIEVE-11*

  – Have previous experience using DATATRIEVE-11 or PRO/DATATRIEVE

  – Have experience in application programming but are unfamiliar with DATATRIEVE-11 or PRO/DATATRIEVE

  Summary

  The first part of the manual explains how to set up DATATRIEVE-11 and is not germane to the use of PRO/DATATRIEVE or the Layered Application Facility. The manual then explains how to use DATATRIEVE-11's data dictionaries and how to use DATATRIEVE-11 to define, retrieve, modify, and restructure data.

- *DATATRIEVE-11 Guide to Writing Reports*

  Objective

  To explain how to produce reports with either DATATRIEVE-11 or PRO/DATATRIEVE.

  Audience

  Users who want to produce reports based on data managed by either DATATRIEVE-11 or PRO/DATATRIEVE. The reader should be familiar with the material covered in *PRO/DATATRIEVE for Beginners* or the *Introduction to DATATRIEVE* and with some topics discussed in the *PRO/DATATRIEVE Handbook* or *DATATRIEVE-11 User's Guide.*

  Summary

  The manual begins with examples of DATATRIEVE-11 reports, and then describes how to write the instructions that specify the format and content of a report. Subsequent chapters discuss more advanced techniques of report writing.

- *DATATRIEVE-11 Reference Manual*

  Objective

  To provide comprehensive reference information about DATATRIEVE-11.

Audience

Users who have a working knowledge of DATATRIEVE-11 or PRO/DATATRIEVE and who understand the basic concepts of data processing.

Summary

The manual explains in detail the rules governing the use of all the features of DATATRIEVE-11. It describes value expressions, Boolean expressions, functions, record selection expressions, and record and field definition clauses. It presents full descriptions of the commands and statements that comprise the DATATRIEVE-11 language.

- *DATATRIEVE-11 Call Interface Manual*

Objective

To show how to write high-level language programs that call DATATRIEVE.

Audience

Programmers familiar with DATATRIEVE.

Summary

The manual explains how to call DATATRIEVE-11 from within programs written in high-level programming languages such as FORTRAN, COBOL, and BASIC. It also explains how to use DATATRIEVE to access data on remote PDP-11 and VAX-11 systems.

- *DATATRIEVE-11 Pocket Guide*

Objective

To provide quick reference information on the syntax and elements of the DATATRIEVE-11 language.

Audience

Experienced users of DATATRIEVE.

Summary

The manual lists the formats of DATATRIEVE-11 commands, statements, clauses, and value expressions and contains basic information about their use.

PRO/DATATRIEVE Sample Application B

This sample application, NORMAL MAILER, uses procedures to create a mailing list and print address labels. The NORMAL MAILER application installation file, MAILLIST.INS, looks like this:

```
! Installation file for NORMAL MAILER application
Name "Normal Mailer"
File MAILLIST.DIC/Keep
File MAILLIST.INI/Keep
File MAILLIST.DAT/Keep
Install [ZZDTR]PRODTR.TSK/Task
Assign Logical DTR$DIC "APPL$DIR:MAILLIST.DIC"
Assign Logical DTR$INI "APPL$DIR:MAILLIST.INI"
Assign Logical SCREEN "TI:"
Assign Logical PRINTER "LP:"
RUN DTR
```

The initialization file contains only one line:

```
:INITIALIZE
```

The INITIALIZE procedure contains the following lines:

```
PROCEDURE INITIALIZE
PRINT "Type :INFO for instructions"
DECLARE X_LIST   PIC X(20).
DECLARE X_LAST   PIC X(15).
DECLARE X_FIRST  PIC X(12).
DECLARE X_ANSWER PIC XXX.
READY MAIL_LIST WRITE
END_PROCEDURE
```

When the user selects NORMAL MAILER from a menu, PRO/DATATRIEVE executes the INITIALIZE procedure before displaying the DTR> prompt, so the user sees the following display:

```
PRO/Datatrieve, DEC Query and Report System
Version: V2.0 28-Sep-84
Copyright (c) 1984 Digital Equipment Corporation
Type HELP for help
Type :INFO for instructions

DTR>
```

The user sees the following when PRO/DATATRIEVE executes the INFO procedure:

```
         ***  Normal Mailer  ***

The following procedures make up the Normal Mailer application:

  :ADD-NAMES      Lets you add new people to a list.

  :PICK-NAMES     Selects a set of people for subsequent procedures.

  :PRINT-LABELS   Prints out names and addresses in a form suitable
                  for mailing labels.  You must run the PICK-NAMES
                  procedure first.

Start a procedure by typing a colon and the procedure name in
response to the DTR> prompt.  For example:

     DTR> :ADD-NAMES

Some procedures (like ADD-NAMES) will cycle indefinitely.  To
end the procedure, press the <EXIT> key.  For example, when
you have finished adding names to your list, press <EXIT> when you
are asked for another name.

DTR>
```

The NORMAL MAILER application dictionary, MAILLIST.DIC, contains the procedures identified to the user. It also contains other definitions that the user does not need to know about:

```
DTR> SHOW ALL
Domains:
        MAIL_LIST
Records:
        MAIL_LIST_RECORD
Procedures:
        ADD_NAMES       INFO            INITIALIZE      PICK_NAMES
        PRINT_LABELS
Tables:
The current dictionary is DW1:[ZZAP00012]MAILLIST.DIC;1
No established collections
Ready domains:
        MAIL_LIST: RMS INDEXED, PROTECTED WRITE
DTR>
```

These definitions look like this:

```
DTR> SHOW MAIL-LIST
DOMAIN MAIL_LIST
 USING MAIL_LIST_RECORD
ON APPL$DIR:MAILLIST.DAT;
DTR> SHOW MAIL-LIST-RECORD
RECORD MAIL_LIST_RECORD
 USING
01 MAIL_LIST_RECORD.
   03 LIST-NAME  PIC X(20)       QUERY-NAME IS LIST.
   03 NAME-FIELDS.
      05 LAST_NAME       PIC X(15).
      05 FIRST_NAME      PIC X(12).
   03 NAME_FL    COMPUTED BY FIRST_NAME|" "|LAST_NAME.
   03 NAME_LF    COMPUTED BY LAST_NAME|", "|FIRST_NAME.
   03 STREET     PIC X(25).
   03 CITY       PIC X(15).
   03 STATE      PIC XX.
   03 ZIP        PIC X(9).
   03 LOCALE     COMPUTED BY CITY|", "|STATE|"   "|ZIP.
   03 PHONE_NO   PIC X(10)       EDIT_STRING IS XXX-XXX-XXXX.
   03 BIRTHDAY   USAGE IS DATE   EDIT_STRING IS DD-MMM-YY.
   03 FILLER     PIC X(50).
;
```

(continued on next page)

```
DTR> SHOW ADD-NAMES
PROCEDURE ADD_NAMES
REPEAT 1000
    BEGIN
    PRINT " "
    X_LIST = **,"Which list do you want to add to"
    PRINT " "
    X_LAST = *,"Last Name"
    X_FIRST = *,"First Name"
    IF ANY MAIL-LIST with NAME-FIELDS = X_LAST!X_FIRST THEN
         PRINT "   That person is already on a list"  ELSE
         STORE MAIL-LIST USING
             BEGIN
             LIST = X_LIST
             LAST-NAME = X_LAST
             FIRST-NAME = X_FIRST
             STREET = *,"Street"
             CITY = *,"City or Town"
             STATE = *,"State (abbreviation)"
             ZIP = *,"Zip Code"
             END
   END
END_PROCEDURE
DTR> SHOW INFO
PROCEDURE INFO
BEGIN
PRINT "          ***  Normal Mailer  ***"
PRINT " "
PRINT "The following procedures make up the Normal Mailer application:"
PRINT " "
PRINT "  :ADD-NAMES      Lets you add new people to a list."
PRINT " "
PRINT "  :PICK-NAMES     Selects a set of people for subsequent procedures."
PRINT " "
PRINT "  :PRINT-LABELS   Prints out names and addresses in a form suitable"
PRINT "                  for mailing labels.  You must run the PICK-NAMES"
PRINT "                  procedure first."
PRINT " "
PRINT "Start a procedure by typing a colon and the procedure name in"
PRINT "response to the DTR> prompt.  For example:"
PRINT " "
PRINT "     DTR> :ADD-NAMES"
PRINT " "
PRINT "Some procedures (like ADD-NAMES) will cycle indefinitely.  To"
PRINT "end the procedure, press the <EXIT> key.  For example, when"
PRINT "you have finished adding names to your list, press <EXIT> when you"
PRINT "are asked for another name."
END
END_PROCEDURE
DTR> SHOW PICK-NAMES
PROCEDURE PICK_NAMES
X_LIST = *,"LIST name or blank for all lists"
FIND MAIL_LIST WITH X_LIST = LIST OR X_LIST = " "
END_PROCEDURE
DTR> SHOW PRINT-LABELS
PROCEDURE PRINT_LABELS
FOR CURRENT SORTED BY ZIP, NAME-FIELDS
         BEGIN
         PRINT SKIP 3,
                 COL 10, NAME_FL(-),
                 COL 10, STREET(-),
                 COL 10, LOCALE(-) ON *,"SCREEN or PRINTER"
         END
END_PROCEDURE
DTR>
```

Sample PASCAL DAB C

This appendix contains a DATATRIEVE Access Block (DAB) definition for PASCAL programs.

```
const LONG_STRING_LEN = 80;
      SHORT_STRING_LEN = 20;

      STATE_INIT = 0;              { values returned in DAB.STATE }
      STATE_CMD  = 1;
      STATE_PVAL = 2;
      STATE_LINE = 3;
      STATE_MSG  = 4;
      STATE_GETP = 5;
      STATE_PUTP = 6;
      STATE_FREQ = 7;

      ES_WARNING       = 0;       { values returned in DAB.ERR_SEV }
      ES_SUCCESS       = 1;
      ES_ERROR         = 2;
      ES_INFORMATION   = 3;
      ES_SEVERE_ERROR = 4;

type
      LONG_STRING  = packed array[1..LONG_STRING_LEN] of char;
      SHORT_STRING = packed array[1..SHORT_STRING_LEN] of char;

      DAB_TYPE = record
          IDI,
          STATE,
          ERR_CODE,
          ERR_SEV  : integer;
          FLAGS    : packed record
             DAB_ACTIVE,
             DTR11,
             PW_PROMPT,
             STR_OVERFLOW,
             BUF_OVERFLOW  : boolean;
          end;
          STR_LEN  : integer;
          STRING   : packed array[1..DTR$STRING_LEN] of char;
          BUFFER   : packed array[1..DTR$BUFFER_LEN] of char;
      end;

var   DAB : DAB_TYPE;
      DTR_MSG_BUFFER : LONG_STRING;
      DTR_ACTUAL_LEN : integer;
```

(continued on next page)

```
[external(DTCMD)]  procedure DTR$COMMAND (var DAB : DAB_TYPE;
                         var CMD_STR : [unsafe, readonly] LONG_STRING;
                         var CSTR_LEN : [readonly] integer);
   seq11;

[external(DTCMD)]  procedure DTR$COMMAND_1ARG (var DAB : DAB_TYPE;
                         var CMD_STR : [unsafe, readonly] LONG_STRING;
                         var CSTR_LEN : [readonly] integer;
                         var ARG1 : [unsafe, readonly] LONG_STRING;
                         var ARG1_LEN : [readonly] integer);
   seq11;

[external(DTCMD)]  procedure DTR$COMMAND_2ARG (var DAB : DAB_TYPE;
                         var CMD_STR : [unsafe, readonly] LONG_STRING;
                         var CSTR_LEN : [readonly] integer;
                         var ARG1 : [unsafe, readonly] LONG_STRING;
                         var ARG1_LEN : [readonly] integer;
                         var ARG2 : [unsafe, readonly] LONG_STRING;
                         var ARG2_LEN : [readonly] integer);
   seq11;

[external(DTCMD)]  procedure DTR$COMMAND_3ARG (var DAB : DAB_TYPE;
                         var CMD_STR : [unsafe, readonly] LONG_STRING;
                         var CSTR_LEN : [readonly] integer;
                         var ARG1 : [unsafe, readonly] LONG_STRING;
                         var ARG1_LEN : [readonly] integer;
                         var ARG2 : [unsafe, readonly] LONG_STRING;
                         var ARG2_LEN : [readonly] integer;
                         var ARG3 : [unsafe, readonly] LONG_STRING;
                         var ARG3_LEN : [readonly] integer);
   seq11;

[external(DTCONT)] procedure DTR$CONTINUE (var DAB : DAB_TYPE);
   seq11;

[external(DTFINI)] procedure DTR$FINISH (var DAB : DAB_TYPE);
   seq11;

[external(DTFREQ)] procedure DTR$FIELD_REQUEST (var DAB : DAB_TYPE;
                         var DOMAIN_NAME : [unsafe, readonly] LONG_STRING;
                         var DOM_NAME_LEN : [readonly] integer;
                         var INFO_BUFFER : [unsafe] LONG_STRING;
                         var BUFFER_LEN : integer);
   seq11;

[external(DTGETP)] procedure DTR$GET_PORT (var DAB : DAB_TYPE;
                         var RECORD_BUF : [unsafe] LONG_STRING;
                         var RBUF_LEN : [readonly] integer;
                         var RECORD_LEN : integer);
   seq11;

[external(DTINIT)] procedure DTR$INITIALIZE (var DAB : DAB_TYPE;
                         var STR_LEN,
                             BUFF_LEN : [readonly] integer;
                         var NODE : [unsafe, readonly] LONG_STRING;
                         var N_LEN,
                             OPTIONS : [readonly] integer);
   seq11;

[external(DTLINE)] procedure DTR$GET_PRINT_LINE (var DAB : DAB_TYPE;
                         var RECORD_BUF : [unsafe, readonly] LONG_STRING;
                         var RBUF_LEN : [readonly] integer;
                         var RECORD_LEN : integer);
   seq11;
```

(continued on next page)

```
[external(DTMSG)]  procedure DTR$GET_MSG (var DAB : DAB_TYPE;
                          var RECORD_BUF : [unsafe, readonly] LONG_STRING;
                          var RBUF_LEN : [readonly] integer;
                          var RECORD_LEN : integer);
   seq11;

[external(DTPEOF)] procedure DTR$PUT_EOF (var DAB : DAB_TYPE);
   seq11;

[external(DTPUTP)] procedure DTR$PUT_PORT (var DAB : DAB_TYPE;
                          var RECORD_BUF : [unsafe, readonly] LONG_STRING;
                          var RBUF_LEN : [readonly] integer);
   seq11;

[external(DTPVAL)] procedure DTR$PUT_VALUE (var DAB : DAB_TYPE;
                          var RECORD_BUF : [unsafe, readonly] LONG_STRING;
                          var RBUF_LEN : [readonly] integer);
   seq11;

[external(DTUNWD)] procedure DTR$UNWIND (var DAB : DAB_TYPE);
   seq11;
```

Index

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Z

Master Index

This index contains entries for the entire Tool Kit PRO/DATATRIEVE documentation set. Manuals are indicated by the following abbreviations:

CALL	Call Interface Manual
DOCSUP	Tool Kit PRO/DATATRIEVE Documentation Supplement
INTRO	Introduction to DATATRIEVE-11
REF	Reference Manual
REPORT	Guide to Writing Reports
UG	User's Guide

Page numbers followed by a "t" contain tables; those followed by "f" contain figures.

Master Index

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA and Puerto Rico call **800–258–1710**

In Canada call **800–267–6146**

In New Hampshire, Alaska or Hawaii call **603–884–6660**

DIRECT MAIL ORDERS (U.S. and Puerto Rico*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532

*Any prepaid order from Puerto Rico must be placed with the Local Digital Subsidiary:
809–754–7575

Tool Kit PRO/DATATRIEVE
Documentation Supplement
AA-CK37A-TH

Reader's Comments

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. _____

_____

_____

_____

_____

_____

_____

Did you find errors in this manual? If so, specify the error and the page number. _____

_____

_____

_____

_____

_____

_____

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

Do Not Tear - Fold Here and Tape

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: DISG Documentation ZKO2–2/N53
DIGITAL EQUIPMENT CORPORATION
110 SPIT BROOK ROAD
NASHUA, N.H. 03062

Do Not Tear - Fold Here and Tape

Cut Along Dotted Line