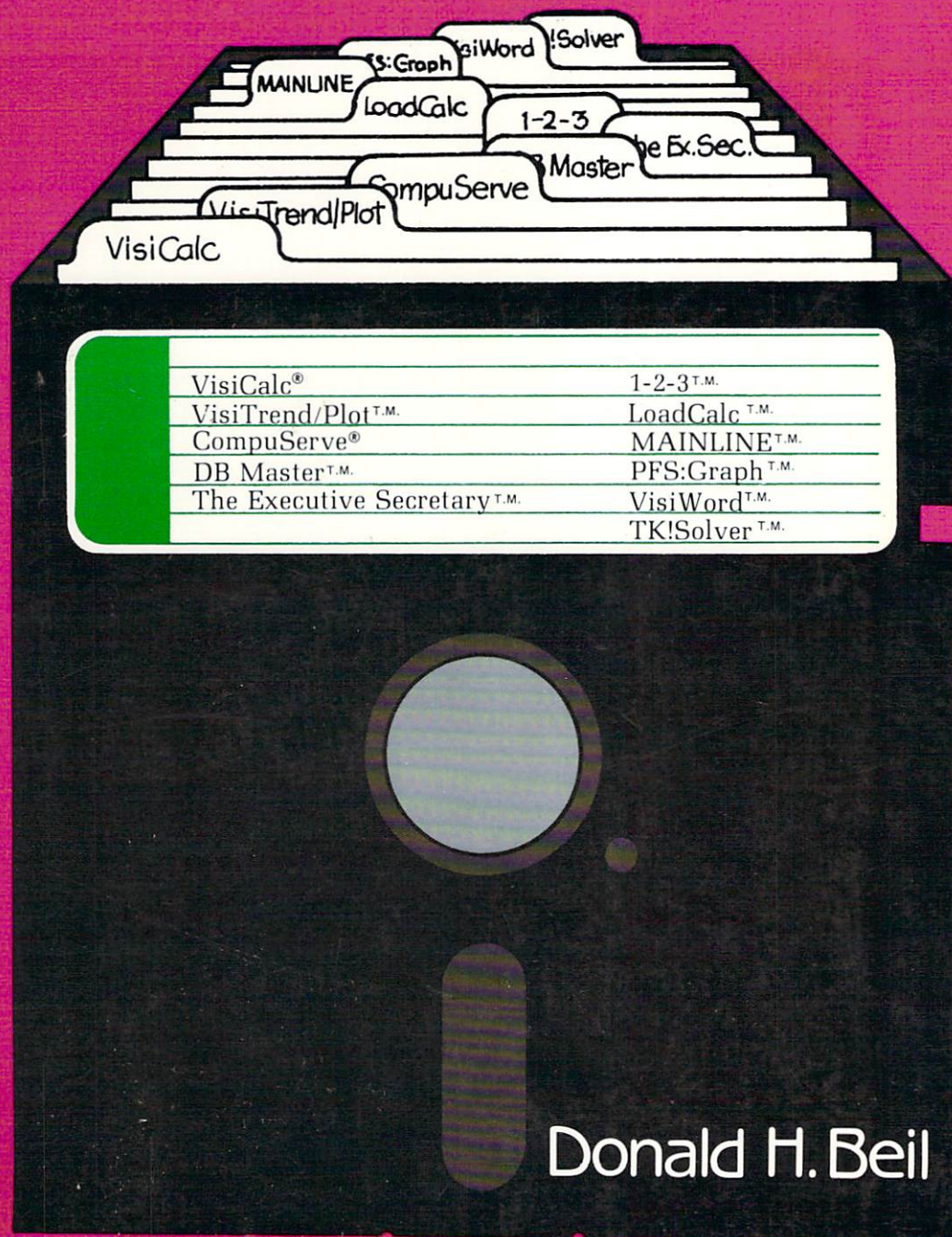




The DIFTM File

For Users of VisiCalc & Other Software



Donald H. Beil

The DIF File_{T.M.}

for users of VisiCalc® and
other software

Other books by Donald H. Beil from Reston Publishing Company

File Processing with COBOL, 1981

The VisiCalc Book, Apple Edition, 1982

The VisiCalc Book, Atari Edition, 1982

The VisiCalc Book for the IBM Personal Computer, 1983

SuperCalc! The Book, 1983

The DIF File
T.M.
for users of VisiCalc® and
other software

Donald H. Beil

National Technical Institute for the Deaf
Rochester Institute of Technology



Reston Publishing Company, Inc.
A Prentice-Hall Company
Reston, Virginia

For Martha, Tommy, and M-TV

Artist: Marian Haley Beil

Library of Congress Cataloging in Publication Data

Beil, Donald H.

The DIF file.

Bibliography: p.

Includes index.

1. File organization (Computer science) 2. Data structures (Computer science) I. Title. II. Title: D.I.F. file.

QA76.9.F5B44 1983 001.64'2 83-9455

ISBN 0-8359-1305-8

ISBN 0-8359-1306-6 (case)

This book is the first in a series of books on "software integration," by Donald H. Beil, Series Editor.

© 1983 by
Reston Publishing Company, Inc.
A Prentice-Hall Company
Reston, Virginia

All rights reserved. No part of this book may be reproduced in any way or by any means without permission in writing from the publisher.

10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

1-2-3 is a trademark of Lotus Development Corporation (registration pending).

Apple is a registered trademark of Apple Computer, Inc.

Chart-Master is a trademark of Decision Resources.

CompuServe is a registered trademark of CompuServe Inc.

COMPUSTAT II is a registered trademark, and proprietary product of Standard & Poor's Compustat Services, Inc.

DB MASTER is a trademark of Barney Stone and Alpine Software Inc.

dBASE II is a trademark of Ashton-Tate, Inc.

DEC is a trademark of Digital Equipment Corporation.

DIF is a trademark of Software Arts Products Corp.

Dow Jones News/Retrieval is a registered trademark of Dow Jones & Company, Inc.

EPSON is a trademark of EPSON America, Inc.

The Executive Secretary is a trademark of Sof/Sys, Inc.

FAST GRAPHS is a registered trademark of Innovative Software.

FileMaster is a registered trademark of Data Base Decisions.

FormsTemp is a trademark of SpreadSoft.

The Graphics Generator is a registered trademark of Robert J. Brady Co.

Graphmagic is a trademark of International Software Marketing.

Graphpower is a trademark of Ferox Microsystems, Inc.

GRAPPLER is a trademark of Orange Micro, Inc.

HealthTemp is a trademark of SpreadSoft.

IBM is a registered trademark of International Business Machines, Inc.

JINSAM is a trademark of JINI Micro-Systems, Inc.

LoadCalc is a trademark of Cypher.

Lotus is a trademark of Lotus Development Corporation (registration pending).

MAGICALC is a trademark of Artsci, Inc.

MAINLINE is a trademark of Gregg Corporation.

MergeCalc is a trademark of Cypher.

Micrograph by 2Y's is a trademark of 2Y's Associates Limited.

Micromodem II is a trademark of Hayes Microcomputer Products, Inc.

PEAR is a trademark of PEAR Systems Corp.

PEAR PLUS is a trademark of PEAR Systems Corp.

PFS: is a registered trademark of Software Publishing Corporation.

SATN is a trademark of Software Arts, Inc.

SMART is a trademark of Software Resources, Inc.

Smartware is a trademark of Cypher.

Software Arts is a trademark of Software Arts Products Corp. and Software Arts, Inc.
speedSTAT is a trademark of SoftCorp International.
SuperCalc is a trademark of Sorcim.
TK is a trademark of Software Arts, Inc.
TK! is a trademark of Software Arts, Inc.
TK!Solver is a trademark of Software Arts, Inc.
Trend-Spotter is a registered trademark of Friend Information Systems.
VALUE is a proprietary product of Capital Market Systems.
VAX is a trademark of Digital Equipment Corporation.
VisiCalc is a registered trademark of VisiCorp.
VisiCalc Advanced Version is a trademark of VisiCorp.
VisiFile is a trademark of VisiCorp.
VisiPlot is a trademark of VisiCorp.
VisiTrend/Plot is a trademark of VisiCorp.
VisiWord is a trademark of VisiCorp.
WordStar is a trademark of MicroPro International Corp.

Contents

Preface, xi

Acknowledgments, xv

- 1 General Information About the DIF Format 1**
 - What is the DIF Format and Who Created It? 1
 - Why is a Standard Data Format Important? 2
 - The DIF Clearinghouse 3
 - The DIF Format is Evolving 3
 - What Software Uses the DIF Format? 3
 - What Hardware Uses the DIF Format? 4
 - Which DIF Is This? 4
 - The DIF Format as Part of a Data Processing System 4
 - The Symbols Used in this Book 6
 - How the DIF Format is Used 8

- 2 Using the DIF Format in a VisiCalc Context 10**
 - Not All VisiCalc Versions Contain DIF Format Capabilities 10
 - Creating, Loading, and Naming DIF Files 10
 - Creating DIF Files 10
 - Loading DIF Files 17
 - Naming DIF Files 17
 - Comparing VC and DIF Files 19
 - Using DIF Files Productively in VisiCalc 24
 - Reconfiguring the Sheet 24
 - Printing Non-Contiguous Portions of the Sheet Contiguously 30

Copying Rectangular Areas	30
Simplifying Worksheet Construction	34
“Pounding” Values in Place	34
Providing Security	35
Increasing Available Memory	35
Reducing Disk Access Time	37
Reducing Diskette Space	37
Reducing Recalculation Time	37
Storing Multiple “What if...” Computations	41
Rolling Data from Period to Period	43
Combining Sheets	46
Computing Year-to-Date Totals	46
Communicating Data to Other Programs	48

3 Case Studies 49

Notes on the DIF Format Case Studies	49
Data Interchange Between VisiCalc and VisiTrend/Plot	51
Case Study 1: VisiCalc to VisiTrend/Plot. Plotting Quarterly Production Data	51
Case Study 2: VisiTrend/Plot to VisiCalc. Manipulating Plot Data in VisiCalc	62
Case Study 3: VisiTrend/Plot to VisiCalc to VisiTrend/Plot	66
Additional Factors in the VisiCalc to VisiTrend/Plot Interchange of Data	69
Data Interchange Between VisiCalc and PFS:Graph	72
Case Study 4: VisiCalc to PFS:Graph	72
Case Study 5: VisiCalc to PFS:Graph. Exchanging Date Information	80
Case Study 6: VisiCalc to PFS:Graph. Multiple DIF Files from the Same Data	83
Case Study 7: VisiCalc to PFS:Graph. Using Several DIF Files to Represent the Same Data	88
Additional Factors in the VisiCalc to PFS:Graph Interchange of Data	90
Data Interchange Between DB Master and The Executive Secretary	91
Case Study 8: DB Master to The Executive Secretary	91
Additional Factors in the Use of DIF Files by DB Master and The Executive Secretary	99
Data Interchange From VisiWord to VisiCalc Using LoadCalc	101
Case Study 9: VisiWord to LoadCalc to VisiCalc	101
Additional Capabilities and Considerations in Using LoadCalc Or VisiWord	111
Data Interchange From CompuServe to VisiCalc Using MAINLINE	112
Case Study 10: CompuServe to MAINLINE to VisiCalc	112
Additional Capabilities and Considerations in Using MAINLINE	120

Data Interchange From DIF Files to 1-2-3	121
What's Integrated Software?	121
What's the Value of the DIF Format with Integrated Software?	121
Case Study 11: DIF Files to 1-2-3	123
Additional Considerations When Using DIF Files with 1-2-3	128
Data Interchange From DIF Files to TK!Solver	129
Case Study 12: DIF Files to TK!Solver	129

4 Guidelines for Using DIF Files 139

Guidelines	140
Conclusion	147

5 Documenting Data Interchange 149

Documentation Within a VisiCalc Environment	150
Documentation Where Several Programs Share the File	150
Conclusion	151

6 A Tutorial on the DIF Format 154

A Place to Start: The Data Section	155
Additional Considerations of the Data Section of a DIF File	162
The Header Section	162
Additional Considerations of the Header Section of a DIF File	166
Additional Information on the DIF Format	166
Manipulating DIF Files with Word Processors	166

7 Limitations of a System Using the DIF Format 170

Limitations of the DIF Format	170
Limitations of Software Using the DIF Format	172

Appendix A: DIF Technical Specification 173

Appendix B: References to Published Listings of BASIC and Pascal Programs That Process DIF Files **200**

Appendix C: Products That Use the DIF Format **202**

Bibliography on the DIF Format 213

Index 225

Preface

This book reports on the exciting DIF file phenomenon.

DIF is a file format created by Software Arts, Inc. as a method of exchanging data between software products. Its use has spread dramatically, and a rapidly growing number of products support the format. This broad support means that a wide variety of potentially unassociated software now has a method of sharing data, a common bond.

In writing this book I have tried to serve a variety of potential readers including the following.

1. VisiCalc users who run no other software will find a full description, with examples, of the multiple applications of the DIF format when used solely within a VisiCalc environment.

2. VisiCalc users for whom “the next step” is data exchange with another software package, such as a graphics package, will find case studies with details of data interchange between VisiCalc and other software that uses the DIF format.

3. Users of one or more of the many applications packages using DIF files will find case studies demonstrating data interchange for a variety of software, and showing by example the variety of strategies used by different software developers to accomplish data interchange with DIF files.

4. Users of a single “integrated” software package will find a case study that details reasons why DIF files are important in that environment.

5. Those who intend to write programs to support DIF file processing will find a tutorial on the format, and a copy of the *DIF Technical Specification* (reprinted through the courtesy of Software Arts, Inc.).

6. Those interested in the process of data interchange will find that the discussion of the DIF format serves to illustrate the problems associated with data exchange no matter what format is used.

7. Those simply curious about DIF files as a result of the expanding use of DIF files by many software packages and the many articles that have appeared referencing the format will hopefully have that curiosity satisfied.

This book began as a brief tutorial on the DIF format. As the book developed, however, its purpose changed. The topic remains the DIF format, and the tutorial is here, but the real focus has become the *process* of exchanging data via DIF files rather than the *format*.

To understand the distinction between the *process* of data exchange and a *format* for data exchange, we need to consider the types of problems we are trying to solve. In each example below we have two software products and want to exchange data between them.

- We may have information on a VisiCalc spreadsheet from which we want to produce a graph, and we don't want to reenter the data into the graphics program.
- We may have a personnel data base file maintained with one software product, and want to use the power of VisiCalc to perform "what if..." analysis to compute salary increments for our employees. Again, we don't want to reenter our data.
- We may have a data base of information about our customers, and want to use word processing software to send individual letters to all of our customers, without reentering our data.
- We may want to extract information from a public network about current stock prices and update a portfolio that we keep either in a data base or on a VisiCalc worksheet.

Within this framework, by *process* we mean the steps followed to create a file of data from one program and read it into another. These steps include selecting desired records and fields from the data of one software system, creating a file from that data, reading that file into a second piece of software and processing it.

By *format* we mean the organization of the data during the middle of the process described above. This may sound like the format is only coincidental to the process; however, its existence and acceptance by a wide variety of software are what make the process possible, as we'll see in this book.

We'll cover the following topics:

- Information about the creator of the DIF format, its origin, its intended uses, its importance, and the problems it helps us solve. (Refer to Chapter 1, General Information About the DIF Format.)
- Directions for using DIF files within a purely VisiCalc environment, that is, a description of its usefulness, ignoring the interchange process. This information will be important to VisiCalc users who have avoided this feature thinking it useful only for communicating data to other programs. (Refer to Chapter 2, Using the DIF Format in a VisiCalc Context.)
- Demonstrations with multiple case studies of the DIF file interchange process between two or more pieces of software. Specific products have been chosen so that the examples are real; however, the level of detail regarding each product (except VisiCalc) is minimal. The emphasis is on the considerations of exchange so that the reader unfamiliar with the specific product will still be able to follow the discussion. (Refer to Chapter 3, Case Studies.)
- A tutorial on the DIF format, including information on processing the files. We'll see that this knowledge is needed if we wish to create our own programs, and that it's also helpful for those of us who will use the DIF format only with commercially available programs that create and read DIF files. (Refer to Chapter 6, A Tutorial on the DIF Format.)
- General advice, in the form of guidelines, documentation suggestions, and a discussion of the limitations of this format. We'll see that we may need to keep detailed documentation of the procedures used to exchange data and of the data exchanged. (Refer to Chapter 4, Guidelines for Using DIF Files; Chapter 5, Documenting Data Interchange; and Chapter 7, Limitations of a System Using the DIF Format.)
- Technical information about the format. (Refer to Appendix A, *DIF Technical Specification*.)
- References to published listings of programs in BASIC and Pascal that process DIF files. (Refer to Appendix B, References to Published Listings of BASIC and Pascal Programs that Process DIF Files.)
- A listing of commercially available software products that support the DIF file format. (Refer to Appendix C, Products That Use the DIF Format.)
- An annotated bibliography of articles and books containing information of interest to those working with the DIF format. (Refer to the Bibliography on the DIF Format.)

Acknowledgments

Once again I find myself indebted to Software Arts, Inc. for providing me with the subject for a book. I recognize their creativity and leadership in developing software for personal computers and believe that their impact on computing by the creation of VisiCalc has been far-reaching and profound. In particular I acknowledge the work done by Daniel S. Bricklin, Chairman of the Board and Executive Vice President, and Robert M. Frankston, President.

Frankston, who designed the DIF format, reviewed an early partial draft of the manuscript and provided valuable comments.

Larry Benincasa, Executive Editor at Reston Publishing Company continues to provide me with complete support for my writing. Ellen Cherry, Production Editor, also has my continuing respect and thanks.

Bill Clymer, Dom Fantauzzo, and Barry Keesan, all colleagues at the National Technical Institute for the Deaf at the Rochester Institute of Technology, provided important critical reviews of the manuscript.

The following individuals have also provided help or encouragement (often unknowingly) with this project: Barry Berkov, Bob Campbell, Tom Castle, Laurence A. Chapman, George Cherry, Doug Ford, Earnest Foreman, Sondra Foster, Tom Hall, Candace Kalish, Dan Kelly, Joellyn Kinzer, Mike Kleper, David Kroenke, Stephen Lindemann, Beth Luchner, Malinda Maher, Jack McGrath, Steven Miller, Chris Morgan, Jean Guy Naud, Steve Nicot, Linda Nolden, Lorna O'Brien, Bob O'Malley, Marv Parsons, Dawna Paton, Sarah Perkins, Bruce Rampe, Barry Siegel, Al Smith, Mary Carol Smith, John Stonner, Warner Strong, Bill Tauskey, Ken Tenuity, Gus Thompson, Ron Till, Ron Tucker, Mike Voelkl, Larry Wardlow, Laurie Whitley, Steve Wilkins, Alan Willett, and Larry Wilson.

I want to thank by name those in word processing who worked on this document: Sharyn Bendzus, Dorothy Cerniglia, Petr J. Chudoba, Debra Dietch, Kathy Exner, Dorris Fox, Mary Jo Ingraham, Irene Kulesa, Barbara Lewis, Tammy Marin, Jane Marvin, Lisa Pisano, Katrina Poquette, Laura Rogers, Betty Shaffer, Anita Sherman, Lisa Smith, and Gary Stape.

My wife, Marian, did an outstanding job with the art in this book, and provided the idea for the cover.

Donald H. Beil

Chapter 1

General Information About the DIF Format

WHAT IS THE DIF FORMAT AND WHO CREATED IT?

The DIF format is a data interchange format—that is, a file format for exchanging data between computer programs. It was developed by Robert M. Frankston, President of Software Arts, Inc. (the creators of VisiCalc), as a method of communicating data between VisiCalc and other computer software. Software Arts, Inc. is actively encouraging DIF's use as a standard format for data interchange.

The DIF format is a concept, a method of organizing data, and not a product that can be purchased.

To date it has been used primarily on personal computers, but it can also be an important bridge for data exchange between personal computers and larger multi-user mainframe computers.

It has been the subject of increasing attention, as evidenced by the list of commercially available products that can process DIF files (refer to Appendix C, *Products That Use the DIF Format*), and by the number of articles published recently on the topic (refer to the *Bibliography on the DIF Format*). This is true because of:

- The extraordinary popularity of VisiCalc, and the resultant desires of software developers to be compatible with it.
- The recognized need for a common or standard format for data interchange.
- DIF itself, designed, its creators say, to be easily used.

Its creators describe it as “the first attempt to establish a standard for exchanging data among personal computers.” Until now it has received no formal recognition from organized standards groups; however, its potential as a *de facto* standard, based on its growing use for interchange, is significant.

WHY IS A STANDARD DATA FORMAT IMPORTANT?

A principal reason for the establishment of a common data format is to simplify the process of sharing data between two or more application programs. We do so to take advantage of the capabilities of one of the programs and thus overcome the limitations of the other.

For example, VisiCalc does not contain sorting or selecting capabilities, and only has very limited graphing capabilities. However, other software packages, or programs we prepare ourselves, may contain these features. If a data file is created by VisiCalc using a common data format, the file can be more easily shared by these multiple programs. In doing so, the user can avoid the often tedious, error-prone activity of reentering the data for the second program.

Using a common data format does more than simply eliminate duplicate data entry. Consider a complicated budget forecast created by VisiCalc. Our data entry may be limited to a single piece of data. For example, suppose we create the forecast based on an assumption about the inflation rate. The budget, consisting of hundreds of values, may be fully generated by entering the single inflation rate used. If we want to plot those values, we don’t want to have to enter them all by hand to the graphics routine. By creating a DIF file directly from VisiCalc, we completely avoid data entry with the exception of the single value we enter onto the VisiCalc worksheet.

In addition, there are many instances where the amount of data to be interchanged may be small, but where we want to use formal data interchange procedures rather than data entry to ensure consistent data in both applications. The propagation of data from one file to another in data processing applications means that we introduce problems in maintaining the integrity of each separate file. Reentering data for several applications may create problems, while shared files or systematic procedures for moving data can reduce potential problems.

THE DIF CLEARINGHOUSE

Software Arts, Inc. established The DIF Clearinghouse to “coordinate and distribute information about the DIF format.” The nonprofit Clearinghouse distributes written technical specifications of the DIF format, and publishes a list of products that support the DIF file format. The first has been reprinted with permission and appears as Appendix A of this book.

The clearinghouse can be reached at:

The DIF Clearinghouse
P.O. Box 638
Newton Lower Falls, MA 02162

THE DIF FORMAT IS EVOLVING

The DIF file format contains provisions for users to invent optional items necessary to meet the needs of their programs. The technique for doing so does not interfere with the standard required portions of the format. Users who do so are encouraged to notify The DIF Clearinghouse so that this information is available to others.

This means that the format has incorporated a method of evolving to meet emerging needs. Such freedom provided to program developers means that while some software may require these optional entries before reading a DIF file successfully, other programs may not insert them. The specifications state that, “if a reading program requires the information provided by an optional item, it should prompt the user to supply the missing information and not require the item itself.” We’ll see examples of this later in the book.

WHAT SOFTWARE USES THE DIF FORMAT?

A large, and growing, number of commercial software application packages include the capability of reading and/or writing DIF files. A list of those known to the author appears as Appendix C, Products That Use the DIF Format. In addition to graphics packages, data-base packages, word processing packages, etc., a number of utility programs are on the market that create DIF files from data in other formats. These enable nonprogramming computer users, or programmers who do not wish to program this interchange, to create DIF files from other files.

An individual with programming skills can prepare a program that

uses DIF files. Examples of such programs in BASIC and Pascal are contained in Appendix A. Other examples have appeared in the computing literature and are referenced in Appendix B. Chapter 6, A Tutorial on the DIF Format, includes additional discussion and explanation of DIF files.

The existence of all-in-one single software packages with multiple functions does not reduce the need for data interchange. In fact, DIF format capabilities are included in some of the packages. Users of this type of software who employ other software or who access data from a mainframe may have use for the DIF format. However, for those whose complete computer use is limited to the functions of this single product, the DIF format may be less meaningful, although it may have uses within the context of the single product.

WHAT HARDWARE USES THE DIF FORMAT?

As a file format, DIF is essentially unrelated to hardware, and can be used to organize data on any storage device.

The DIF format, as a standard, allows files from one machine to be used on a machine from a different vendor, assuming that a hardware, and potentially a software link, exists between the two.

WHICH DIF IS THIS?

This DIF is *not* related to any of the following designations also used within computing contexts:

- Document Interchange Facility, two IBM program products for distributed office systems (reference IBM document GH20-2440-0).
- Display Information Facility (DIF), an IBM Licensed Program for the IBM System/38 (reference IBM document GH30-0145-2).
- the DIF suffix used to indicate “difference” files on Digital Equipment Corporation VAX computer files (reference DEC document AA-D023C-TE).

THE DIF FORMAT AS PART OF A DATA PROCESSING SYSTEM

It is extremely important to recognize that a common data format is only a part, and potentially a small part, of the process of sharing data between software packages. Data is only one element of a data processing system. Procedures are another important aspect. We’ll see that moving

data from one program to another is a multistep process in which each step must be performed carefully.

Additionally, as we leave one program in which we've created a DIF file and enter another program in which we want to read the DIF file, we leave one software environment for another. The two programs may function quite differently. For example, one program may be menu-driven in orientation while a second requires a user to learn a command language. A third program may prompt the user for a series of responses to specific software-generated questions. Each of these different types of programs may use the keyboard in different ways, with a single key having different meanings in each piece of software.

Moving from application to application involves much more than interchanging data. The multistep processes we'll encounter with software that functions inconsistently creates a situation that requires our full attention to the process.

A system of procedures is needed. We'll suggest guidelines for such an exchange, as well as methods of documenting the necessary steps. (Refer to Chapter 4, Guidelines for Using DIF Files and Chapter 5, Documenting Data Interchange.)

THE SYMBOLS USED IN THIS BOOK

This book presents ideas in a visual way, using figures where possible to complement and illustrate concepts presented in the text. In this section, we'll briefly present some of the graphic symbols that have been adopted in the figures. An arrow is used to connect these symbols, and represents the flow of data.

Concept or Topic

Symbol Used

A printed report.

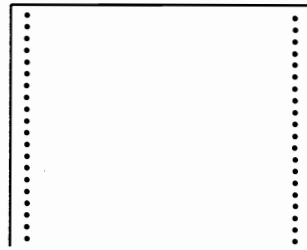



Figure 1-1

VisiCalc.

Figure 1-2

Concept or Topic	Symbol Used
Files used on a personal computer. We'll often show the filename above the symbol with a suffix to show the file type. For example we'll use <u>Suffix</u> to indicate .DIF a DIF file .VC a VisiCalc file	 Figure 1-3

A commercially produced or user written piece of software.

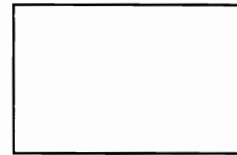


Figure 1-4

Plotted data.

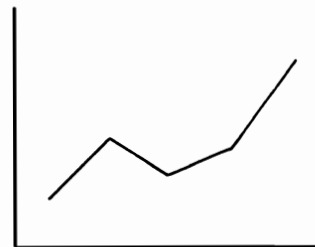


Figure 1-5

HOW THE DIF FORMAT IS USED

This section contains graphic representations of several of the many ways that DIF files can be used to interchange data.

Example 1: Plot data generated as a DIF file by VisiCalc and read by plotting software, as shown in Figure 1-6.

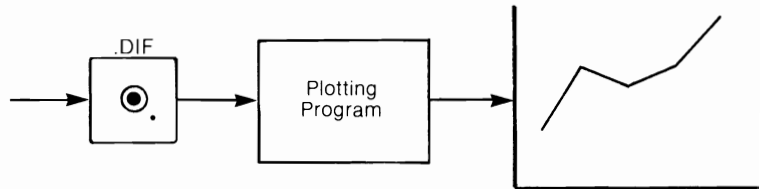


Figure 1-6

Example 2: Pass salary data from a personnel data base on a micro-computer to VisiCalc. Use the "what if..." capabilities of VisiCalc to determine salary increases for employees, and send the new salaries back to the data base, all using DIF files, as shown in Figure 1-7.

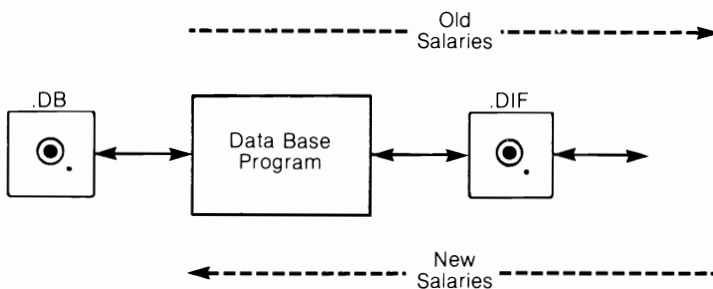


Figure 1-7

Example 3: Pass salary data in a DIF format from a personnel file of a data base product to word processing software where new contracts are generated for each employee for the coming year, as shown in Figure 1-8.

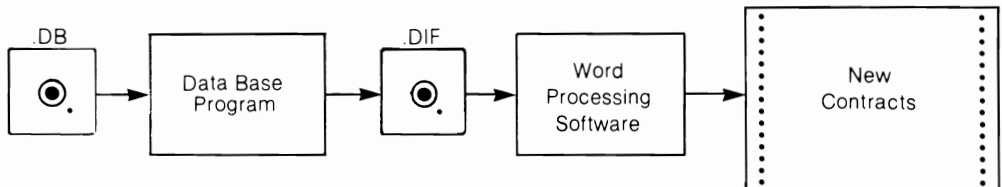


Figure 1-8

Chapter 2

Using the DIF Format in a VisiCalc Context

In addition to the uses of the DIF format for exchanging data between programs, the format also provides significant capabilities within a VisiCalc environment. This means that the VisiCalc user who has no need to interface data with other programs can still use the DIF format to advantage in many instances. In this chapter it's assumed that the reader is knowledgeable about VisiCalc, although unfamiliar with the impact of DIF files on its use. We'll see that a knowledge of the technical specifications, and of the file formatting technique associated with DIF files, is not necessary to take advantage of DIF files within this context.

This chapter details the methods of creating and retrieving DIF files within VisiCalc, and then presents a series of examples in which we'll see how the DIF format contributes to the problem solving capabilities of VisiCalc.

NOT ALL VISICALC VERSIONS CONTAIN DIF FORMAT CAPABILITIES

The capability to use DIF files is available to users of VisiCalc except for those using VisiCalc version 1.37 for the Apple II or II Plus computer or VisiCalc PLUS for the HP-83/85. Users of version 193 on the Apple II or II Plus should upgrade to version 202 since saving DIF files with the earlier version may destroy files on the diskette.

CREATING, LOADING, AND NAMING DIF FILES

Creating DIF Files

We will create a DIF file from the data that we see on the VisiCalc screen of Figure 2-1. This screen is part of a larger quarterly sales report that we'll work with later in this chapter. That larger report is shown next.

Sales by Quarter		
Quarter:	Sales (000)	Percent %
First	5	10
Second	20	40
Third	25	50
Fourth	0	0
TOTAL	50	100

With VisiCalc, DIF files are created by using the Storage command. The DIF files we create can be placed on a diskette with other DIF or non-DIF files. To save a DIF file, a rectangle is specified in a way that is similar to indicating a rectangle with the Print command. Thus, the position of the cursor with this command is important, and we've carefully placed it at entry A1. We'll see examples later in the chapter in which we place it deliberately at other entries.

	A	B	C	D	E	F	G	H
1	5	10						
2	20	40						
3	25	50						
4	0	0						
5	50	100						
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								
21								

Figure 2-1

The exact VisiCalc prompt lines associated with the Storage command vary slightly from version to version; however, to work with DIF files on any version we begin with the three characters

/S#

where the keystrokes have the following meanings:

Keystroke	Means
/	Enter the command structure of VisiCalc.
S	Choose the Storage command.
#	Select the DIF file format.

At this point we see a prompt line which says

DATA: SAVE LOAD

Although it is not clear from the prompt line, we are being asked to enter either

Keystroke	Means
S	SAVE, or create a DIF file.
L	LOAD, or retrieve a DIF file.

Since we wish to create a DIF file we'll type an

S

We are now prompted with a line saying

DATA SAVE: FILE FOR SAVING

We now provide a filename for the file we are creating by entering the name at the keyboard. We can also use the right arrow key to scroll the diskette directory until we find the name we wish to use, either as is, or by changing it while it is on the edit line.

In this instance we'll name the file

FIG21R.DIF

The next section of this chapter discusses naming DIF files. Here we type these characters followed by the appropriate key to enter data, for example the RETURN key on the Apple, the ENTER key on the IBM

Personal Computer, etc. Throughout this book we'll refer to that key as

RETURN

If the name we enter for a file already exists on the diskette we'll be prompted with

DATA SAVE: FILE EXISTS. Y TO REPLACE

If we type

Y

our file from the worksheet in memory will replace the existing file on the diskette. Although most other keys, if pressed, will cancel the Storage command, we should develop the habit of using the key (for example, BREAK on the IBM Personal Computer) or key combination (CTRL-C on the Apple II Plus) designated as the cancel key for our VisiCalc version. If we simply press "any" key to prevent replacement here, we may press the Y accidentally and cause replacement.

We now see another prompt,

DATA SAVE: LOWER RIGHT

Here we'll enter

B5

either by typing it at the keyboard, or by moving the cursor to that location, that is, by pointing the cursor with the arrow keys. With B5 on the entry line we'll press

RETURN

which generates the prompt

DATA SAVE: R, C OR RETURN

At this point we can save the data by row (R), or by column (C). Pressing RETURN will store the file by row, a redundancy we'll explain below. Let's conceptually look at the difference by listing Figure 2-1 by row and by column.

In Figure 2-2 we've listed the values by row (press R or RETURN), and in Figure 2-3 we've shown the values by column (press C). We have shown both the row and column representations horizontally in those figures. We could have shown them both vertically. We have also added vertical lines as breaks between the data.

It is very important to realize that we are *not* illustrating a full DIF file in these figures, but are simply examining the order of the values if

represented in row or column order. When we study the full format of the DIF file in Chapter 6, A Tutorial on the DIF Format, we'll see that a great deal of information appears in the file in addition to the values shown in these two figures. (A confusing element of DIF files created from VisiCalc is that the reverse of what we think should happen actually does happen. For now we'll accept this seeming discrepancy since using DIF files successfully within a VisiCalc context does not require a full explanation of the DIF file format.)

A1 (V) 5

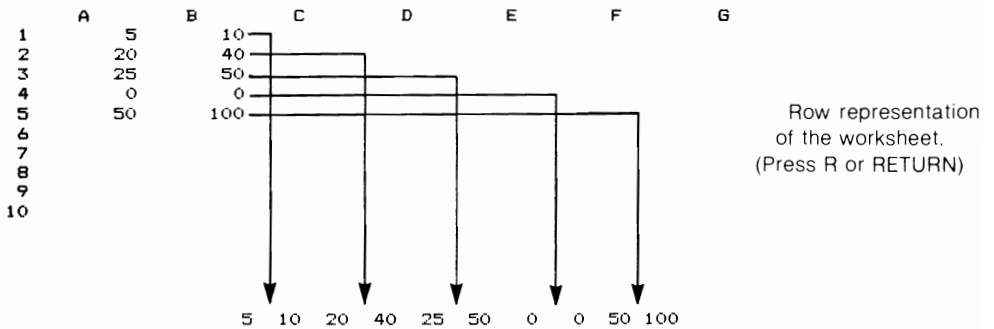


Figure 2-2

A1 (V) 5

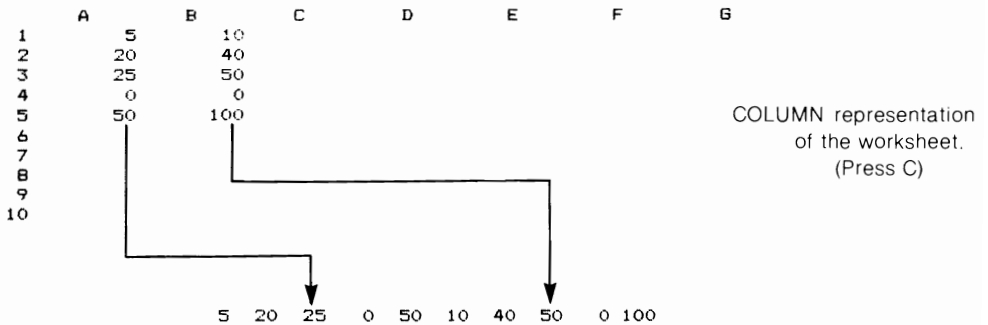


Figure 2-3

For our file we'll store the values in row order by pressing RETURN, at which time the file is finally created. We'll see that for many examples in this chapter it doesn't matter if we store the file by row or column as long as we load it in the same order. In those instances, pressing the RETURN key may be the habit to develop. This key may be easier to locate than the C or R at this point in the process, and removes some user decision making by automatically using this default to row order.

Let's review in full the series of steps required to create a DIF file from VisiCalc. We'll present, in Figure 2-4, the keystrokes we enter, the VisiCalc prompts, and an explanation.

Because of the options available with DIF files, we see that creating the file is a multistep operation.

Summary:
Creating a DIF File from VisiCalc

Keystrokes	VisiCalc Prompt Line	Explanation
/		Enter the VisiCalc command structure.
	COMMAND: BCDEFGIMPRSTVW-	The list of available commands.
S		Select the Storage command.
	STORAGE: L S D I Q #	The list of available storage options. (This will vary depending on the version of VisiCalc used.)
#		Indicate the storage option for DIF files.
	DATA: SAVE LOAD	Creating (S=SAVE) or retrieving (L=LOAD) the file?
S		Select the save option.
	DATA SAVE: FILE FOR SAVING	A request for the file-name.
FIG21R.DIF RETURN		Enter the name we have selected for the file. If the name already exists on the diskette we'll see an additional prompt of Y TO REPLACE
		If we type Y, we replace the file. Most other keys cancel the Save command.

Figure 2-4

	DATA SAVE: LOWER RIGHT	A request for us to enter the lower right of the area of the worksheet we want saved.
B5 RETURN		The desired coordinate.
	DATA SAVE: R, C OR RETURN	Is the file to be saved by row (R or RETURN) or column (C) order?
RETURN		Create the file in row order.

Figure 2-4 continued

Loading DIF Files

Loading a DIF file into VisiCalc involves responses to a set of prompts that are similar to those we observed with our actions above to save a DIF file. Figure 2-5 presents these steps in a format similar to that just presented for saving a DIF file.

As with saving the file, the position of the cursor is important when loading a DIF file. It indicates the top left corner of the location on the current worksheet where we want the DIF file loaded. For our example, we'll reload the file at the location from which it was saved, here with the cursor at A1. Later in the chapter we'll demonstrate the advantage of loading files at locations other than those from which they were saved.

At this point the file is loaded, and appears to be the same file that we've saved. We'll shortly discover the significant differences between the worksheet before and after we save it.

Naming DIF Files

As we'll see in the next section, the content of DIF files is significantly different from the content of other files stored with the save option of the Storage command. We may have both types on the same diskette. Since each is loaded differently, it is important to be able to distinguish between the two types of files. The VisiCalc documentation suggests that the suffix

.DIF

Summary:
Loading a DIF File into VisiCalc

Keystrokes	VisiCalc Prompt Line	Explanation
/S#	... (same as Figure 2-4)	Select the DIF option of the Storage command.
	DATA: SAVE LOAD	Store (S=SAVE) or retrieve (L=LOAD) the file?
L		We want to load the file.
	DATA LOAD: FILE TO LOAD	What is the filename? Here we can type the name, or scroll the directory with the right arrow key.
FIG21R.DIF RETURN		We've entered the name of the file we want loaded.
	DATA LOAD: R, C OR RETURN	Load in row (R or RETURN) or column (C) order? The RETURN provides the default of row order.
RETURN		We'll load the file in the same order in which it was saved. Later in this chapter we'll see examples in which we change the order. For applications in which we are indifferent as to the order of the file (row or column), we should use the RETURN key when saving and loading.

Figure 2-5

be appended to the names of DIF files, and that

.VC

be appended as a suffix to filenames of files saved with the /SS option. Some versions of VisiCalc (for example, the IBM Personal Computer and the ATARI versions) do this automatically while others (for example, the Apple II and II Plus 16 Sector version) do not.

Since the row/column designation may be easily forgotten, we may wish to include either a C or an R as part of our name, perhaps as the character before the period, to serve as a reminder of the order in which this file has been created. We'll assume that a filename without such a designation is in row order, the default.

When scrolling filenames, some versions of VisiCalc only present ".DIF" files when searching under the /S#S or /S#L options and ".VC" files when searching under the /SS or /SL options.

COMPARING VC AND DIF FILES

When a DIF file is created by VisiCalc, only labels and current values (to full precision) are stored. The formulas and formats associated with the entries are not stored. Let's demonstrate this by showing a series of figures which illustrate these points.

In Figure 2-6 we are presented with

- a VisiCalc screen before we create a DIF file.
- a listing of the formulas and formats associated with this screen. (The bottom of the listing has been deliberately omitted since it is not of interest for this discussion.) This is obtained in slightly different ways with different versions of VisiCalc, for example

/SS LPT1: ENTER is used for the IBM Personal Computer,
and /SS,SP RETURN is used for the Apple II and II Plus, (with VisiCalc finding the printer slot).

Notice in the listing that we see the numeric values we have entered at A1, A2, A3, and A4, that there is a sum formula for the columns at A5 and B5, and that at B1, B2, B3, and B4 we have formulas that compute the percentage by dividing the relative value in column A by the sum at entry A5.

However, study Figure 2-7. The screen of that figure is created by storing the spreadsheet of Figure 2-6 as a DIF file, clearing memory, and

A1 (V) 5

	A	B	C	D	E	F	G
1	5	10					
2	20	40					
3	25	50					
4	0	0					
5	50	100					
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							

Print formulas and formats.

```
· >B5: @SUM(B1...B4) ·  
· >A5: @SUM(A1...A4) ·  
· >B4: (A4/A5)*100 ·  
· >A4: 0 ·  
· >B3: (A3/A5)*100 ·  
· >A3: 25 ·  
· >B2: (A2/A5)*100 ·  
· >A2: 20 ·  
· >B1: (A1/A5)*100 ·  
· >A1: 5 ·  
· : ·  
· : ·
```

Figure 2-6

A1 (V) 5

	A	B	C	D	E	F	G
1	5	10					
2	20	40					
3	25	50					
4	0	0					
5	50	100					
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							

Print formulas and formats.

```
· >B5: 100 ·  
· >A5: 50 ·  
· >B4: 0 ·  
· >A4: 0 ·  
· >B3: 50 ·  
· >A3: 25 ·  
· >B2: 40 ·  
· >A2: 20 ·  
· >B1: 10 ·  
· >A1: 5 ·  
· : ·  
· : ·
```

Figure 2-7

then loading the DIF file. Then a printed listing is obtained as discussed above. Notice in the printed listing that we have *only* values, and that *no* formulas have been preserved.

DIF files created by VisiCalc do *not* have formulas preserved, only the current value of that formula to full precision.

It is important to realize that when we obtain a listing of the contents of the cells, as we have done here, we are *not* listing the contents of the DIF file. Instead we are listing the contents of the VisiCalc sheet that were generated by VisiCalc after reading and processing the DIF file. We are not viewing the DIF file.

In the example above the values have been carefully chosen to ensure that the precision is identical in the two figures.

However in Figure 2-8 we've selected different values for column A, and also have added a format of

/FI (local integer format)

to each item in column B. This can be seen on the entry contents line of the top screen where

B1 /FI (V) (A1/A5)*100

appears. However, in the middle of that figure we see the screen after saving the contents as a DIF file, clearing the sheet, reloading that DIF file, and moving the cursor to B1. We now have the entry

B1 (V) 38.3333333333

The listing at the bottom right of that figure confirms that formats, e.g. the /FI here, and formulas, have not been preserved; instead, values calculated to full precision appear.

The next two figures, Figure 2-9 and Figure 2-10, illustrate how labels, including those produced with the Repeating Label command, are handled when DIF files are created by VisiCalc. The screen of Figure 2-9 contains row and column titles, including some that are right justified. For example at location A8 in the listing we see

>A8:/FR"fourth

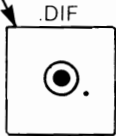
and at C9 we see a repeating label of equal signs, shown in the listing as

>C9:/-=-

B1 /FI (V) (A1/A5)*100

	A	B	C	D	E	F	G
1		23	38				
2		12	20				
3		21	35				
4		4	7				
5		60	100				
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							

Save the full sheet as a DIF file.



Clear the sheet.

Load the DIF file.

B1 (V) 38.3333333333

	A	B	C	D	E	F	G
1		23	38.333333				
2		12	20				
3		21	35				
4		4	6.666667				
5		60	100.				
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							

Print formulas and formats of this file.

```
>B5: 99.9999999999
>A5: 60
>B4: 6.6666666666
>A4: 4
>B3: 35
>A3: 21
>B2: 20
>A2: 12
>B1: 38.3333333333
>A1: 23
.
.
.
```

Figure 2-8

A1

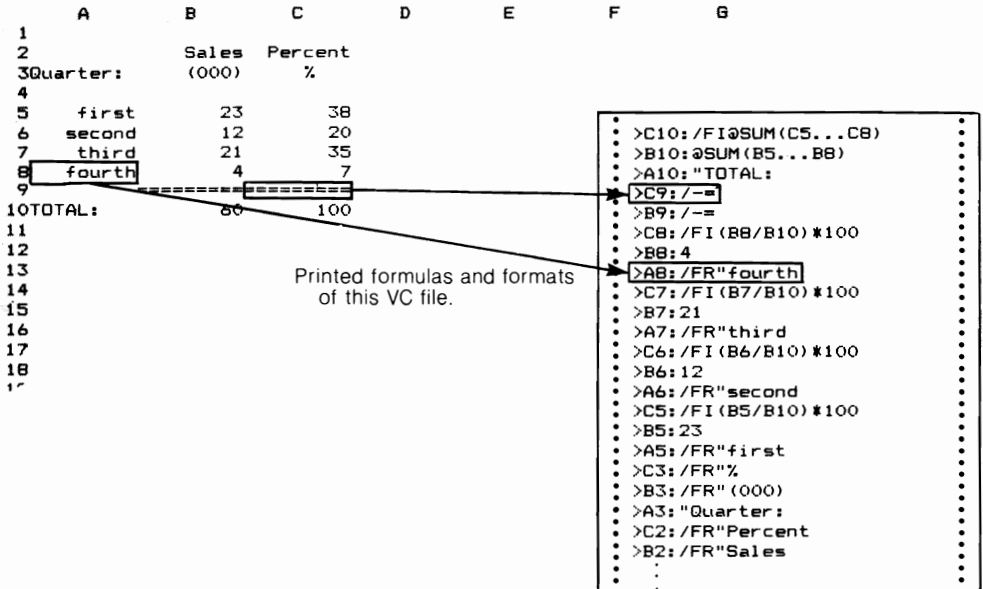


Figure 2-9

A1

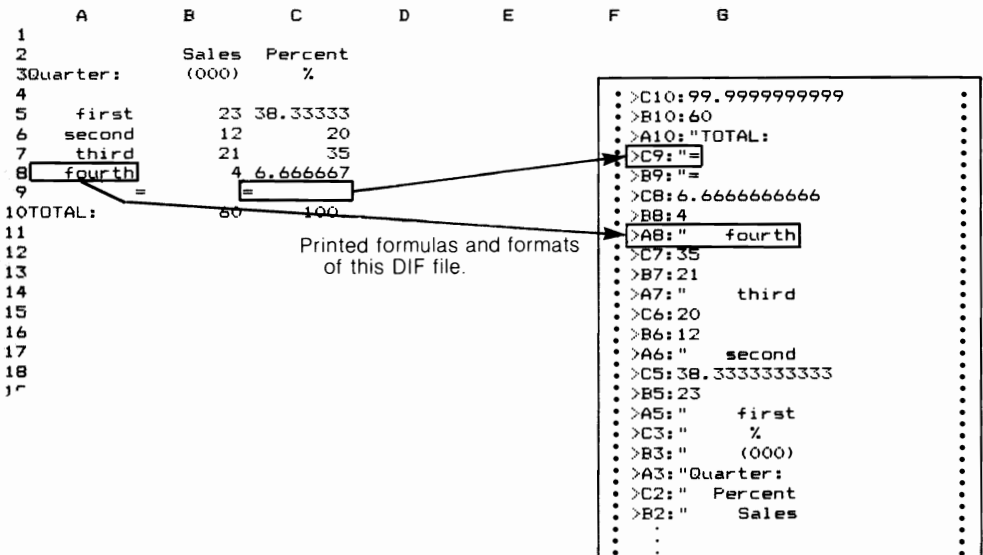


Figure 2-10

Now let's save that sheet as a DIF file, clear memory, and load the DIF file. Figure 2-10 results. Notice that on the sheet the justification of the labels has been preserved by inserting sufficient spaces before the label to place the label correctly. We see this at A8 in the listing, where we now have

```
>A8:"   fourth
```

with three spaces after the quote (") character. However, if we now increase the width of the columns on this sheet, the labels will remain where they are since the true format has not been saved. (When *values* have left justification, that justification is lost when DIF files are created.)

At location C9 in Figure 2-10, the repeating label has not been retained. In its place a single equal sign, the item being repeated, appears. This we see on the screen, and on the listing at C9 where we have

```
>C9:"=
```

If we will be saving DIF files and then using them in a VisiCalc context we may wish to *avoid* the use of the Repeating Label command, and instead enter those items in the same way that other labels are entered.

Before finishing with this topic, let's observe how several unique items appear when saved as part of a DIF file and then retrieved. If the functions @ERROR, @FALSE, @NA, or @TRUE (or any entry evaluated to one of these four values) are saved to a DIF file, the value returned when reloaded with VisiCalc will be @ERROR, @FALSE, @NA, or @TRUE, respectively. When @PI is saved as a single entry and then retrieved, the function no longer appears, and 3.1415926536 will be substituted for it.

USING DIF FILES PRODUCTIVELY IN VISICALC

We are now ready to examine a variety of specific instances where the use of DIF files adds to our problem solving capabilities in VisiCalc.

Reconfiguring the Sheet

Suppose that we have just carefully entered the labels

```
WEEK 1 ... WEEK 52
```

as shown in the top of Figure 2-11, and decide that we want to list these labels down column A instead of across row 1. This cannot be done with the Replicate command, but can be done with DIF files.

First store the sheet from A1 to AZ1 as a DIF file in row order. Clear memory, then load the DIF file in column order. (Or store in column order and load in row order.) The results appear in Figure 2-11.

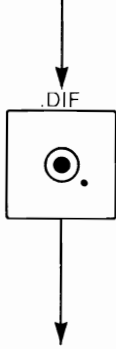
Another example appears in Figure 2-12 where we see the sheet we have worked on earlier in the chapter. It appears at the top of the figure with a vertical orientation, with the sums at the bottom of the columns. Suppose we want to change to the horizontal orientation shown in the lowest screen of that figure. We can save the sheet in row order, clear, and load in column order, giving us the screen in the middle of Figure 2-12. On first glance the sheet appears to need considerable work; however, to change from the middle to the bottom sheet we: reentered one label, deleted several columns, changed to a global integer format, reentered two formulas, and changed the order of computation to row (R). In some instances this cosmetic work will be significantly less than the work of completely reentering the full sheet with a different orientation.

We must remember that the middle step here, creating a DIF file, was done at the cost of removing all formulas and replacing them with calculated values. Although we're assuming this is satisfactory here, it will be undesirable for other applications.

C1 (L) Week 3

C
34

	A	B	C	D	AY	AZ	BA
1	Week 1	Week 2	Week 3	Week 4	1Week 51	Week 52	
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							
20							
21							



Save the full sheet as a DIF file in ROW order.

Clear the sheet.

Load the DIF file in COLUMN order.

A3 (L) Week 3

C
34

	A	B	C	D	E	F	G	H
1	Week 1							
2	Week 2							
3	Week 3							
4	Week 4							
5	Week 5							
6	Week 6							
7	Week 7							
8	Week 8							
9	Week 9							
10	Week 10							
11	Week 11							
12	Week 12							
48	Week 48							
49	Week 49							
50	Week 50							
51	Week 51							
52	Week 52							
53								
54								
55								

Figure 2-11

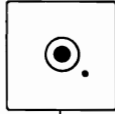
A1

C
35

1	A	B	C	D	E	F	G	H
2		Sales	Percent					
3	Quarter:	(000)	%					
4								
5	first	23	38					
6	second	12	20					
7	third	21	35					
8	fourth	4	7					
9		=====						
10	TOTAL:	60	100					

Save the full sheet as a DIF file in ROW order.

.DIF



Clear the sheet.

Load the DIF file in COLUMN order.

A1

C
35

1	A	B	C	D	E	F	G	H
2		Quarter:	(000)		first	second	third	fourth
3		Sales	%		23	12	21	4
4		Percent			38.33333	20	35	6.666667

Cosmetic changes.

A1 /FR (L) Quarter

R
35

1	A	B	C	D	E	F	G	H
2	Quarter		first	second	third	fourth	TOTAL:	
3	Sales		23	12	21	4	60	
4	Percent		38	20	35	7	100	

Figure 2-12

Let's look at an example where we reconfigure *part* of the sheet, instead of the full sheet as we've done with the previous two examples. In Figure 2-13 we've prepared a Quarterly Budget Report that provides the status of our budget by individual budget codes. Our year-to-date expenditure (YTD AMOUNT) and our projected expenses (EXPENSES PROJECTED) are entered and VisiCalc then produces a number of values, including information we can use to determine if we are over or under our budget allocation.

Suppose we realize that we've incorrectly entered data and erroneously entered all YTD AMOUNTS in the EXPENSES PROJECTED column. We cannot use the Move command to correct our error since this may have an undesired effect on our formulas in other columns. In addition, the Move command will destroy the report and column headings.

DIF files can be used to switch the data as indicated in the marked areas of Figure 2-13. The steps we'll take include:

- Creating a DIF file from the area indicated in the YTD AMOUNT column. We'll carefully place the cursor at the item at the top of the marked column and indicate the last item of the marked area as the lower right of this file.
- Similarly creating a DIF file from the marked area of the EXPENSES PROJECTED column.
- Loading each column in the opposite location, effectively switching the data.

This type of reconfiguration works successfully here since all of the values in the columns were entered as data values, and were therefore not computed by formulas at each entry. If the entries did contain formulas, our storing and reloading of columns would destroy the formulas.

Switching these columns did not disturb any other areas of the worksheet, and thus we see that DIF files can be loaded onto a worksheet that already has other data and formulas in place.

QUARTERLY BUDGET REPORT
 INCLUDING PROJECTED EXPENSES
 NUMBER: 481 PERIOD ENDING: 03/31/99

CODE	BUDGET START OF YEAR	BUDGET CURRENT	BUDGET-DIFFERENCE		YTD AMOUNT	-----EXPENSES-----		OVER/UNDER BUDGET	
			AMOUNT	PERCENT		% TO DATE	PROJECTED TOTAL		

110	18100.00	18100.00	0.00	0	8832.00	49	9450.00	18282.00	182.00
1120	95400.00	87500.00	-7900.00	-8	41084.10	47	46440.00	87524.10	24.10
1141	20000.00	20000.00	0.00	0	9744.00	49	10230.00	19974.00	-26.00
142	8600.00	8700.00	100.00	1	3967.44	46	4724.00	8691.44	-8.56
145	7000.00	13500.00	6500.00	93	11897.61	88	7612.39	19500.00	6000.00
150	13600.00	13600.00	0.00	0	163376.60	1201	12042.23	175418.83	161818.83
152	13600.00	13600.00	0.00	0	2500.63	18	3924.46	6425.09	-7174.91
155	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
160	12500.00	6000.00	-6500.00	-52	0.00	0	0.00	0.00	-6000.00
1200	32100.00	30800.00	-1300.00	-4	0.00	0	16909.20	16909.20	-13890.80
210	0.00	0.00	0.00	0	4541.14	0	0.00	4541.14	4541.14
220	0.00	0.00	0.00	0	1771.20	0	0.00	1771.20	1771.20
230	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
232	0.00	0.00	0.00	0	218.16	0	0.00	218.16	218.16
235	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
240	0.00	0.00	0.00	0	1450.05	0	0.00	1450.05	1450.05
250	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
255	0.00	0.00	0.00	0	215.70	0	0.00	215.70	215.70
260	0.00	0.00	0.00	0	209.25	0	0.00	209.25	209.25
270	0.00	0.00	0.00	0	5489.50	0	0.00	5489.50	5489.50
310	1800.00	1800.00	0.00	0	1929.72	107	270.28	2200.00	400.00
320	2300.00	2300.00	0.00	0	1542.91	67	357.09	1900.00	-400.00
408	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
590	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
430	200.00	200.00	0.00	0	181.50	91	18.50	200.00	0.00
470	2500.00	2500.00	0.00	0	1108.42	44	1108.42	2216.84	-283.16
515	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
520	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
585	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
620	800.00	800.00	0.00	0	568.68	71	920.00	1488.68	688.68
630	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00
1755	150800.00	150800.00	0.00	0	101545.06	67	49254.94	150800.00	0.00
112	0.00	0.00	0.00	0	0.00	0	0.00	0.00	0.00

TOTALS	379300.00	370200.00	-9100.00	-2	362163.67	98	163261.51	525425.18	155225.18

Figure 2-13

Printing Non-Contiguous Portions of the Sheet Contiguously

Figure 2-14 presents a 10-year fiscal forecast for a company. We wish to print the two areas as shown. These are the salary accounts, and we want to study separately the future values in these accounts. We can not do so in this case with the Print command, since deleting the middle columns will cause the rightmost columns to appear with values ERROR. However, as indicated in the figure, we can store the two areas as DIF files, clear memory, load the files as shown, and then print.

If we wish, we can save the new composite sheet, either as a .DIF or VC file.

As another example of the usefulness of DIF files in reconfiguring a sheet for printing let's examine Figure 2-15 Part A and B. Suppose that we have the long vertical worksheet at the left in memory and that we want to print it as shown in Part A of Figure 2-15. We can print it as is, including printing the headings twice, and then cut-and-paste, or we can use DIF files to help us rearrange the elements of the sheet.

We will divide up the sheet and store it as three files:

HEADINGS.DIF
LEFT.DIF
RIGHT.DIF

as shown. We clear memory, then load the file HEADINGS.DIF twice, as indicated in Part B of Figure 2-15. Then load

LEFT.DIF and RIGHT.DIF

as shown. We now have the sheet reorganized as we want, and we can print the report.

Here DIF files have really served as a cut-and-paste software tool, for they have allowed us the flexibility of moving sections of the sheet.

Copying Rectangular Areas

In Figure 2-15 we not only moved blocks of the sheet, but we copied one rectangular area into several locations. This occurred with the HEADINGS.DIF file, for we were able to copy a rectangular area from one sheet onto several areas of a second sheet. This capability is different from the VisiCalc Replicate command because it contains neither Relative (R) nor No Change (N) options for coordinates; however, it does allow us to copy from a source area that is a rectangle, which we can't do with the Replicate command. If any formulas with coordinates appear in the rectangle we copy *from*, they will not appear in the areas we copy *to*.

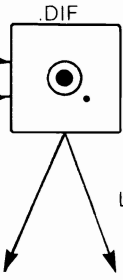
Ten Year Fiscal Forecast Based on a
1% Increase per Year

Code	Description	Fiscal Year 0	Fiscal Year1	Fiscal Year2	Fiscal Year3	Fiscal Year4	Fiscal Year5	Fiscal Year6	Fiscal Year7	Fiscal Year8	Fiscal Year9
8110	Prof. Salr.	1502	1708	1878	2065	2271	2498	2747	3021	3321	3655
8120	P/T Prof.	38	39	42	44	47	50	53	56	59	62
8141	Consultant	99	100	102	104	107	110	113	117	120	125
	Sub.	1469	1686	1929	2204	2511	2865	3263	3714	4221	4796
142	P/T I	116	127	139	152	167	183	201	221	243	267
143	P/T II	34	36	38	41	44	47	50	53	56	60
	Sub.	170	186	202	222	244	267	293	322	354	389
144	P/T III	187	197	208	220	234	249	265	282	299	318
145	P/T IV	18	19	20	21	22	23	24	25	26	28
	Sub.	175	186	198	212	226	242	259	277	296	318
150	Hourly I	62	66	70	75	80	85	90	97	104	112
152	Hourly II	15	16	17	18	19	20	21	22	23	24
155	Hourly III	0	0	0	0	0	0	0	0	0	0
	Sub.	77	84	91	99	108	117	128	140	153	167
220	Benefits	269	279	291	304	318	333	349	367	386	406
221	Retiremt	112	122	132	143	154	167	180	195	211	228
222	L/T Dis	12	13	14	15	16	17	18	19	20	22
240	Wthr Lo	51	56	61	67	73	80	88	96	105	115
260	Wthr Med	15	16	17	18	19	20	22	24	26	28
270	Wthr Cap	27	29	31	34	37	40	44	48	52	57
	Sub.	426	466	509	556	607	663	723	787	856	930
310	Supplies I	64	68	72	77	82	88	94	101	108	116
320	Supplies II	62	66	70	75	80	86	92	99	106	114
	Sub.	126	134	142	152	162	174	186	199	214	230
400	Commutat	72	76	80	85	90	96	102	109	116	124
390	Cons. Trvl	12	13	14	15	16	17	18	19	20	22
	Sub.	84	92	100	109	119	126	134	143	154	166
430	Telephone	25	27	29	31	34	37	40	44	48	52
	Sub.	25	27	29	31	34	37	40	44	48	52
470	Travel I	68	72	77	82	88	94	101	108	116	124
515	Travel II	27	29	31	34	37	40	44	48	52	57
	Sub.	75	81	88	96	105	114	125	137	149	163
520	Hospitality	19	20	21	22	23	24	25	26	27	29
540	Meals	36	38	40	43	46	49	52	56	60	64
580	Meals I	0	0	0	0	0	0	0	0	0	0
585	Meals II	0	0	0	0	0	0	0	0	0	0
	Sub.	46	50	54	59	64	70	76	83	90	98
620	Equip. Maint	37	40	43	46	50	54	58	63	68	74
	Sub.	37	40	43	46	50	54	58	63	68	74
630	Repair	38	41	44	48	52	56	61	66	71	77
651	Maintenance	42	46	50	55	60	66	72	79	86	94
	Sub.	80	87	95	104	113	124	135	148	161	176
1725	Contingency	18	19	20	21	22	23	24	25	26	28
726	Conting 2%	18	19	20	21	22	23	24	25	26	28
	Sub.	36	38	40	42	44	46	48	51	52	56
8912	Equipment	120	132	145	159	174	191	209	229	251	279
8914	Supplies 20%	184	194	205	217	230	245	261	279	298	329
	Sub.	304	326	350	376	404	436	470	508	549	608
TOTAL		3195	3581	3984	4504	5057	5646	6274	6945	7663	8436

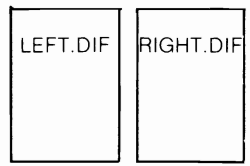
Create two DIF files.

RIGHT.DIF

LEFT.DIF



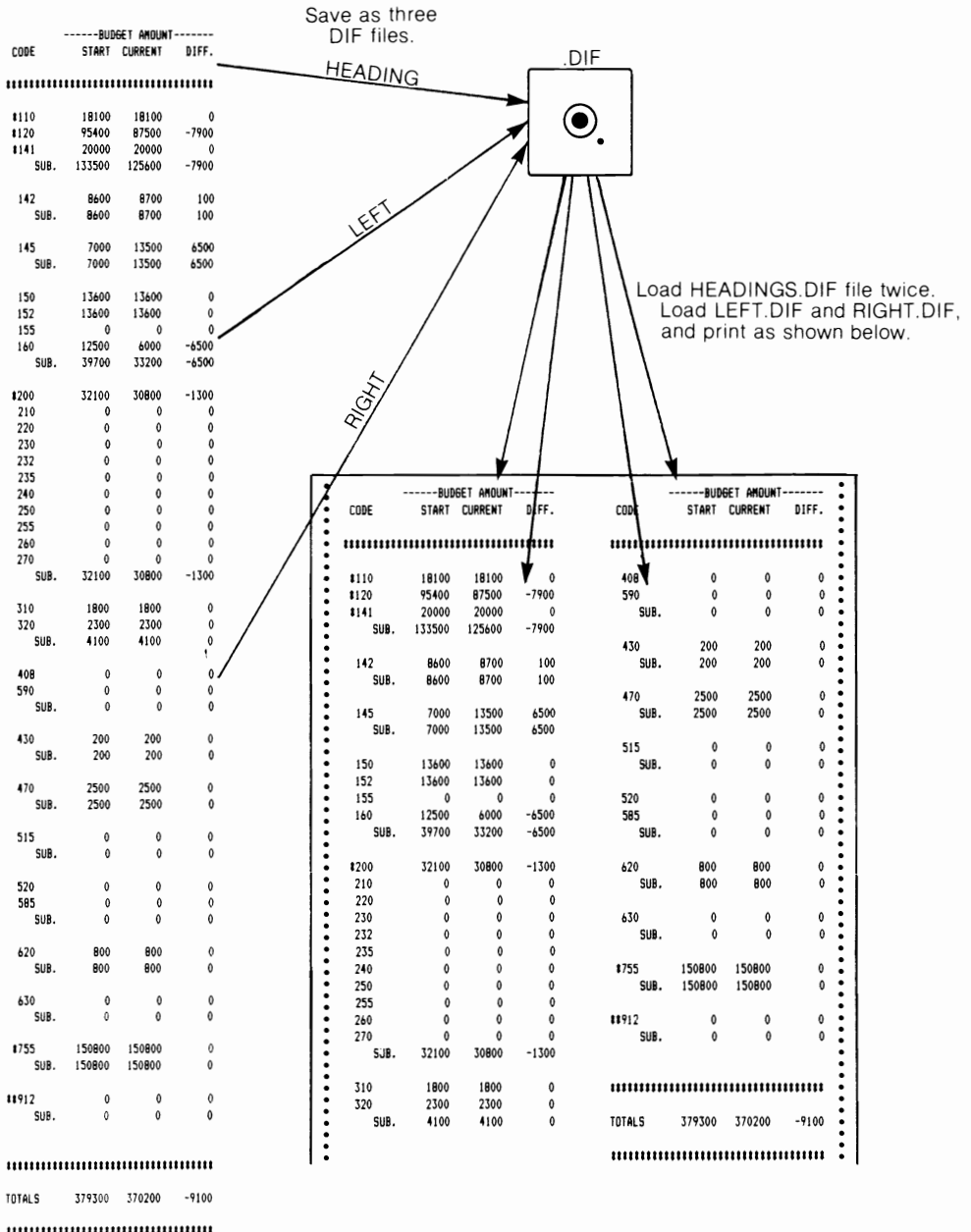
Load two DIF files.



Print.

Code	Description	Fiscal Year+8 (000)	Fiscal Year+9 (000)
8110	Prof. Salr.	3323	3655
8120	P/T Prof.	55	60
8141	Consultant	205	225
	Sub.	3583	3940
142	P/T I	243	267
143	P/T II	111	122
	Sub.	354	389
144	P/T III	223	245
145	P/T IV	33	36
	Sub.	256	281
150	Hourly I	127	139
152	Hourly II	26	28
155	Hourly III	0	0
	Sub.	153	167

Figure 2-14



Load HEADINGS.DIF file twice.
Load LEFT.DIF and RIGHT.DIF,
and print as shown below.

Figure 2-15A

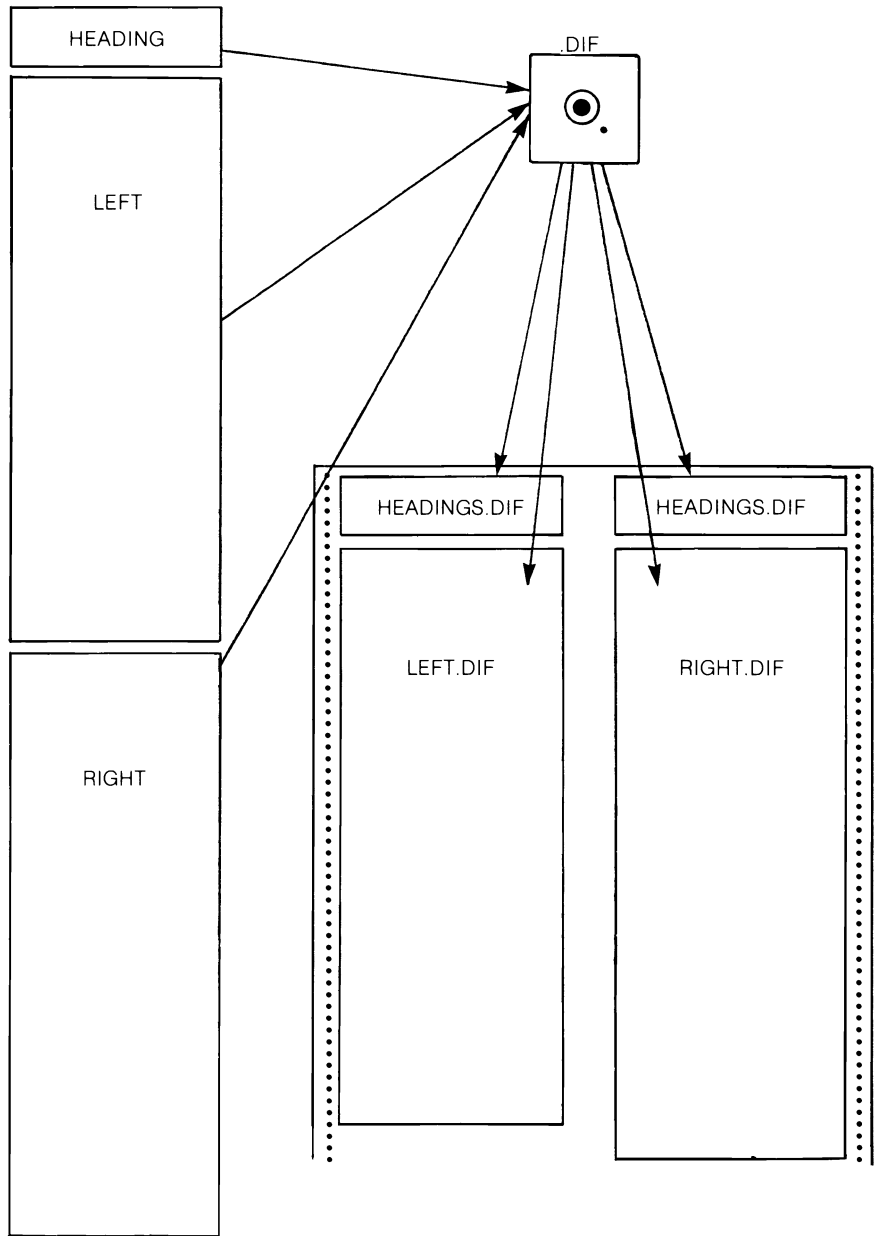


Figure 2-15B

Simplifying Worksheet Construction

Because DIF files may be loaded onto a worksheet at any location, we can build multiple sheets that have common elements. For example suppose we frequently include lengthy budget code descriptions that span multiple columns. We can build a DIF file with this data, and then load it as needed in different locations of separate worksheets.

“Pounding” Values in Place

VisiCalc provides the capability, with the pound sign (#), to replace a formula at a single entry with its current value. With the cursor at the desired location, typing

```
# RETURN
```

will perform this replacement action for us. The pound sign only works at a single location and its action can not be replicated over an area of the sheet unless we move from entry to entry repeating this action.

However, since DIF files store *only* the values, and not the formulas, the DIF format capability can be used to pound into place selected rectangular areas, or an entire sheet. Save the area to a DIF file, then load it into the same area.

As we’ll see below, this action can, for appropriate applications, save time and provide additional memory space for us.

As we’ve seen before, the value stored may be different from the value displayed on the screen. If we want the values to be identical, then we are responsible for manipulating the formula at the entry so that its value and the displayed value are the same. For example, suppose we are displaying values on the screen with the integer format (/FI or /GFI). This format rounds the value for display, meaning that

6.8 will be displayed as 7

If we use the Integer built-in function, for example

```
@INT(entry)
```

(that is, enclose the formula at this entry within the Integer function) we will not round, but truncate instead, and actually change the value, so

6.8 will become 6

To round the value to an integer we can use the formula

```
@INT(entry+0.5)
```

As another example, the formula

$$\text{@INT}((\text{entry}+0.005)*100)/100$$

will round the computed value to dollar and cents. This formula adds 0.005 (one-half a cent), multiplies by 100 (shifting the decimal point to the right of the cent portion of the number), truncates that value (with the @INT function), and then divides by 100 to move the decimal point back to its original position.

We must often perform considerable preparation of the elements of our spreadsheets to take advantage of DIF capabilities.

Providing Security

Because DIF files do not contain formulas, this file format can be used as a means of electronically sharing data with others while not revealing to them the assumptions or computations (formulas) used to generate the data.

Increasing Available Memory

DIF files can help overcome the constraint that limited memory imposes on the VisiCalc user. For example, the upper screen of Figure 2-16 requires almost all of the available memory, leaving only one kilobyte of storage, as indicated by the memory indicator on the screen of that figure. The impact on available memory of storing this sheet as a DIF file, clearing memory, and then reloading the file is apparent on the lower screen of Figure 2-16. The memory indicator records 21 kilobytes of memory now available. (The difference will vary for other sheets). Also note that the recalculation order indicator that had been R (rowwise) has now become C (columnwise), VisiCalc's default.

The additional 20 kilobytes were obtained at the cost of losing all formulas. Therefore, we can only use this technique when we have no need for the "what if..." capabilities of the sheet to be reduced in size.

Although for our example we stored the full sheet in the DIF format, we could selectively store and load only some areas of the sheet, and thus obtain some additional memory without destroying all formulas. To accomplish memory reduction in this instance we'd need to save then load the DIF format portion of the file, then save the full sheet, clear memory, and load the full sheet.

This can also be helpful when consolidating several worksheets into one when the worksheets would not otherwise fit together on a single sheet.

A1 (L) LEASE

R
01

	A	B	C	D	E	F	G	H
1	LEASE STARTS:		15 JUNE 1999					
2								
3	EQUIPMENT NAME	MODEL	PURCHASE	DATE	BASE	MAINTAIN.		!
4		NUMBER	PRICE	PURCH'D	LEASE PR.	COST		!
5	-----							
6		4331-J02			2748.05			
7		3278-A02			111.35			
8		5424-A01			531.25			
9		3262-001			409.70			
10		3340-A02			1096.50			
11	-----							
12		3310-A02			608.60			
13		3310-B02			531.25			
14		3411-001			651.10			
15		3410-001			243.10			
16		3278-002			146.20			
17		3279-B03			147.90			
18	-----							
19	VSE/ADV. FUNCT.				153.00			
20	VSE/IPF				34.85			



Save as a DIF file.

Clear memory.

Load the DIF file.

A1 (L) LEASE

C
21

	A	B	C	D	E	F	G	H
1	LEASE STARTS:		15 JUNE 1999					
2								
3	EQUIPMENT NAME	MODEL	PURCHASE	DATE	BASE	MAINTAIN.		!
4		NUMBER	PRICE	PURCH'D	LEASE PR.	COST		!
5	-----							
6		4331-J02			2748.05			
7		3278-A02			111.35			
8		5424-A01			531.25			
9		3262-001			409.7			
10		3340-A02			1096.5			
11	-----							
12		3310-A02			608.6			
13		3310-B02			531.25			
14		3411-001			651.1			
15		3410-001			243.1			
16		3278-002			146.2			
17		3279-B03			147.9			
18	-----							
19	VSE/ADV. FUNCT.				153			
20	VSE/IPF				34.85			

Figure 2-16

Reducing Disk Access Time

Reducing the size of the sheet also means that the time required to save and load a sheet may be reduced. As examples, timings were made for the sheet of Figure 2-16 on an Apple II Plus microcomputer. The results appear in Figure 2-17 (timings will vary for other sheets). The recorded times include only disk drive access time and not operator time. On microcomputer systems with larger memory, these timing differences are significant for large worksheets.

Reducing Diskette Space

DIF files *may* require less space on a diskette than associated VisiCalc files, although this is not always true. Let's work with the VisiCalc file at the top of Figure 2-16 in memory. Storing it on an Apple II Plus microcomputer requires 116 sectors of a diskette, as shown in Figure 2-18. When saved as a DIF file the number of sectors decreased to 62. For worksheets where we will be calculating a variety of "what if..." values and then storing each separately, this can potentially reduce our diskette storage.

However if we clear memory, load the DIF file, and then save the worksheet that was loaded from the DIF file as a "VC" file, the number of sectors decreases again to 45. Thus, the two-step process *may* reduce requirements even more.

We should observe that if our original VisiCalc file had been the latter file requiring 45 sectors, and if we stored it as a DIF file, we would increase the diskette resource used to 62 sectors for this file. This illustrates the amount of overhead associated with DIF files, a topic we will discuss again in Chapter 6, A Tutorial on the DIF Format.

Reducing Recalculation Time

A recalculation for the full sheet associated with the screen of Figure 2-16 requires approximately 22 seconds. Figure 2-19 contains part of a fourteen-page listing of the formulas and formats associated with the screen shown in Figure 2-16 and is included to give an idea of the complexity level of the formulas.

This sheet, when saved as a DIF file and reloaded, requires less than one second to recalculate (there are no formulas). This can be useful for some applications where we will be recalculating regularly and the recalculation only affects selected areas of the sheet. Other areas can be pounded into place, with resultant time savings for the recalculations.

Timings for Saving and Loading the Sheets
of Figure 2-16 as "VC" and "DIF" Files

Sheet in Memory is from	Sheet on diskette is:	Action/Explanation	Time (in seconds)
VC file		Save the sheet as a VC file.	55
VC file		Save the sheet as a DIF file.	24
DIF file		Save the sheet as a VC file.	25
		The distinction between VC files and DIF files disappears once the file is loaded by VisiCalc. For this line and the next, the DIF file "in memory" means the sheet has no formulas or formats.	
DIF file		Save the sheet as a DIF file.	24
	VC file	Load the sheet as a VC file.	113
		Timing includes calculation time which occurs automatically after loading.	
	VC file	Load the sheet as a DIF file.	34
		Although loading action occurs, no sheet is available in memory, and a clear worksheet results from this erroneous request to VisiCalc to load a VC file with the "load DIF" option. We'll need to distinguish carefully between the two types of files.	
	DIF file	Load the sheet as a VC file.	57
		The result of this misloading of a DIF file is that the full sheet contains only the label EOD in entry A1.	
	DIF file	Load the sheet as a DIF file.	32

Figure 2-17

Diskette Storage Requirements of the "Same" File Saved
in Several Formats

VisiCalc Command	Filename	Sectors	Notes
Note: We'll begin this process with a worksheet we've just developed in memory under VisiCalc. We have not yet saved it to a diskette.			
/SS EXAMPLE1.VC RETURN	EXAMPLE1.VC	116	Save the worksheet as a VC file. It occupies 116 sectors.
/S#S EXAMPLE2.DIF ...	EXAMPLE2.DIF	62	Save the same worksheet as a DIF file. It requires 62 sectors.
/CY			Clear memory.
/S#L EXAMPLE2.DIF ...			Load the DIF file that we've just created. This gives us the worksheet without formulas or formats.
/SS EXAMPLE3.VC	EXAMPLE3.VC	45	Store this file as a VC file. We can now see the overhead of the DIF file format, here 17 sectors (62-45).

Figure 2-18


```

>F31:111.35
>A31:*ACF/VTAME
>AM30:@IF (@OR (E30=0,AM4<E30),AL30,@IF (E30=AM4,D30,630))
>AL30:@IF (@OR (E30=0,AL4<E30),@INT (V30+M30+AI30#F30+.5)/100+F30,@IF (E30=AL4,D30,630))
>AK30:@IF (@OR (E30=0,AK4<E30),C1/30#Y30+(1-(C1/30)*(@INT (V30+M30+AI30#F30+.5)/100+F30)),@IF (E30=AK4,D30,630))
>AJ30:@IF (@OR (E30=0,AJ4<E30),Y30,@IF (AJ4=E30,D30,630))
>AI30:5
>AH30:@IF (@OR (E30=0,AH4<E30),Y30,@IF (E30=AH4,D30,630))
>Z30:@IF (@OR (E30=0,Z4<E30),Y30,@IF (E30=Z4,D30,630))
>Y30:@IF (@OR (E30=0,Y4<E30),@INT (V30+M30#F30+.5)/100+F30,@IF (E30=Y4,D30,630))
>X30:@IF (@OR (E30=0,X4<E30),C1/30#N30+(1-(C1/30)*(@INT (V30+M30#F30+.5)/100+F30)),@IF (E30=X4,D30,630))
>M30:@IF (@OR (E30=0,M4<E30),N30,@IF (M4=E30,D30,630))
>V30:5
>U30:@IF (@OR (E30=0,U4<E30),N30,@IF (U4=E30,D30,630))
>T30:@IF (@OR (E30=0,T4<E30),N30,@IF (T4=E30,D30,630))
>N30:@IF (@OR (E30=0,N4<E30),@INT (M30#F30+.5)/100+F30,@IF (N4=E30,D30,630))
>M30:5
>L30:@IF (@OR (E30=0,L4<E30),F30,@IF (L4=E30,D30,630))
>K30:@IF (@OR (E30=0,K4<E30),F30,@IF (K4=E30,D30,630))
>J30:@IF (@OR (E30=0,J4<E30),(C1/30)#F30,@IF (J4=E30,D30,630))
>I30:0
>F30:436.9
>A30:*BASIC
>AM29:@IF (@OR (E29=0,AM4<E29),AL29,@IF (E29=AM4,D29,629))
>AL29:@IF (@OR (E29=0,AL4<E29),@INT (V29+M29+AI29#F29+.5)/100+F29,@IF (E29=AL4,D29,629))
>AK29:@IF (@OR (E29=0,AK4<E29),C1/30#Y29+(1-(C1/30)*(@INT (V29+M29+AI29#F29+.5)/100+F29)),@IF (E29=AK4,D29,629))
>AJ29:@IF (@OR (E29=0,AJ4<E29),Y29,@IF (AJ4=E29,D29,629))
>AI29:5
>AH29:@IF (@OR (E29=0,AH4<E29),Y29,@IF (E29=AH4,D29,629))
>AB29:"----->)
>AF29:/--
>AE29:/--
>AD29:/--
>AC29:/--
>AB29:/--
>AA29:" -----
>Z29:@IF (@OR (E29=0,Z4<E29),Y29,@IF (E29=Z4,D29,629))
>Y29:@IF (@OR (E29=0,Y4<E29),@INT (V29+M29#F29+.5)/100+F29,@IF (E29=Y4,D29,629))
>X29:@IF (@OR (E29=0,X4<E29),C1/30#N29+(1-(C1/30)*(@INT (V29+M29#F29+.5)/100+F29)),@IF (E29=X4,D29,629))
>M29:@IF (@OR (E29=0,M4<E29),N29,@IF (M4=E29,D29,629))
>V29:5
>U29:@IF (@OR (E29=0,U4<E29),N29,@IF (U4=E29,D29,629))
>T29:@IF (@OR (E29=0,T4<E29),N29,@IF (T4=E29,D29,629))
>S29:"----->)
>R29:/--
>Q29:/--
>P29:/--
>O29:" -----
>N29:@IF (@OR (E29=0,N4<E29),@INT (M29#F29+.5)/100+F29,@IF (N4=E29,D29,629))
>M29:5
>L29:@IF (@OR (E29=0,L4<E29),F29,@IF (L4=E29,D29,629))
>K29:@IF (@OR (E29=0,K4<E29),F29,@IF (K4=E29,D29,629))
>J29:@IF (@OR (E29=0,J4<E29),(C1/30)#F29,@IF (J4=E29,D29,629))
>I29:0

```

Figure 2-19

Storing Multiple “What if...” Computations

Figure 2-20 is a sheet from which a variety of budget projections can be prepared. Notice that the bottom half of the sheet contains directions for using the sheet and also a data entry area. When loaded, part of these instructions will appear as the first window, and following the directions the user will move through the process of entering the data (at the bottom of the sheet) from which projections are produced. This process may be followed for a series of projections based upon varying assumptions. For each set, a report is printed if desired, and a backup copy of the template is saved.

After each series of data is entered, if we wish to save the spreadsheet on a diskette, we may decide that we want to save only the upper half of the sheet, and not repeatedly save the instructions and other information on the sheet.

Saving the top report area as a DIF file provides this capability. We'll require less time for saving and reloading the sheet at a later time. In this way we can produce and then save multiple copies of the same sheet, each based upon a different series of “what if...” data.

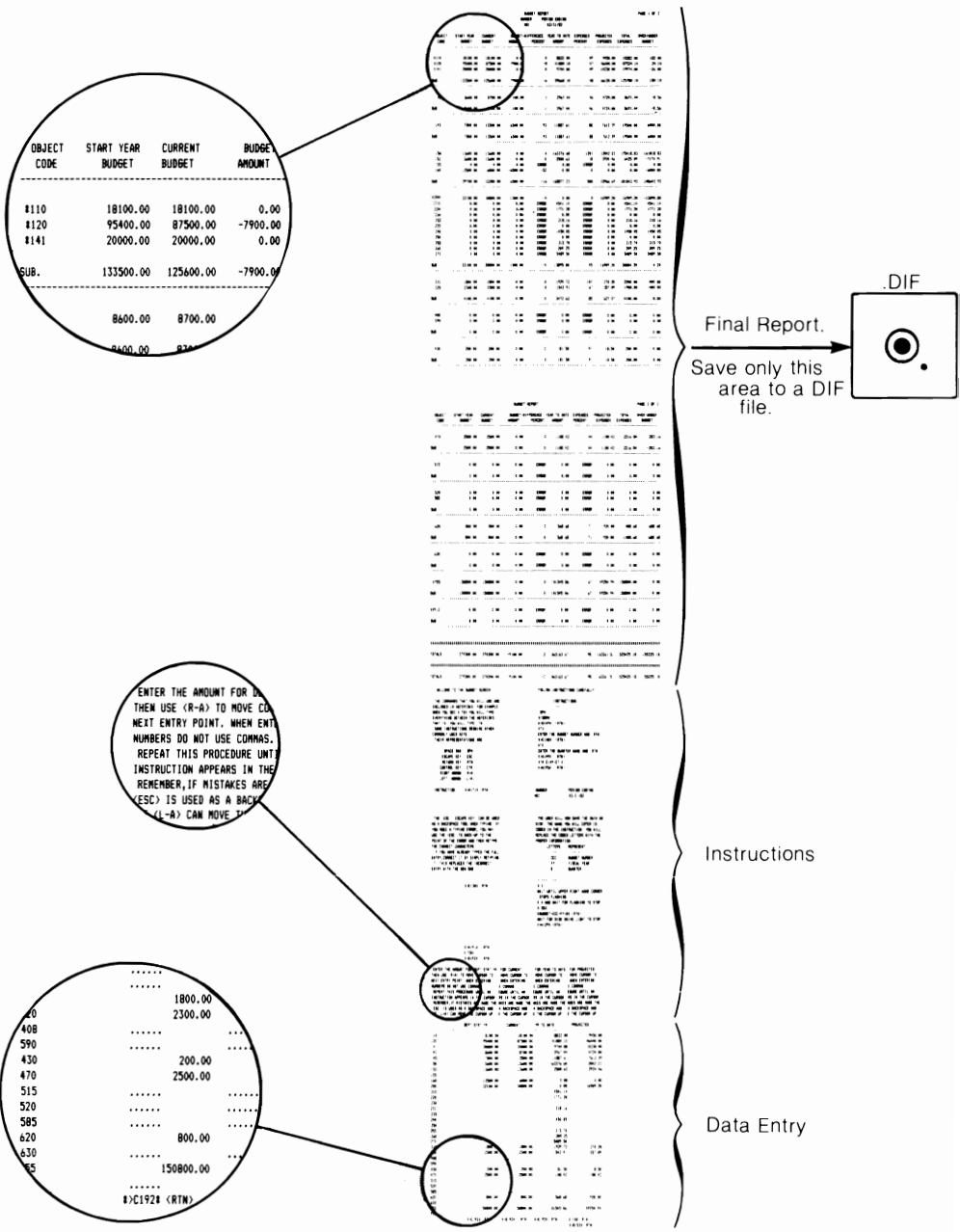


Figure 2-20

Rolling Data from Period to Period

Often data from the end of one application is used to begin another. Common examples include rolling financial data from the end of one time period to the start of another, as illustrated in Figure 2-21. In that figure we begin with a spreadsheet that contains columns for the budget at the start-of-quarter, and then columns for: additions/reductions during the quarter, expenses this quarter, and the end-of-quarter balance.

We want to roll the last column from this report (end-of-quarter balance) to the start-of-quarter column for the next quarter. To do so we save the last column, the marked area, without the column headings, as a DIF file and load it as explained next.

For this example we've prepared a template with a "hole" in it as shown in the figure. On that template we've written the line

LOAD DIF FILE AT B14

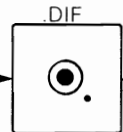
This line includes a direction (really a reminder for us), for loading the DIF file. We'll position the cursor at location B14, load the DIF file, and we've now initialized our template (as shown at the far right), so that it's ready for use during the coming quarter.

QUARTERLY BUDGET REPORT
INCLUDING PROJECTED EXPENSES
NUMBER: 848 PERIOD ENDING: 03/30/99

CODE	START OF QTR.	ADDITIONS REDUCTIONS	EXPENSES THIS QTR	END OF QTR

*110	255000.00	0.00	155000.00	100000
*120	745800.00	-7900.00	280000.00	457900
*141	5800.00	0.00	800.00	5000
142	500.00	100.00	4000.00	-3400
145	50000.00	6500.00	23000.00	33500
150	0.00	0.00	0.00	0
152	6400.00	0.00	6400.00	0
155	2900.00	0.00	2900.00	0
160	7000.00	-6500.00	1200.00	-700
*200	400.00	-1300.00	1700.00	-2600
210	0.00	0.00	0.00	0
220	0.00	0.00	0.00	0
230	0.00	0.00	0.00	0
232	0.00	0.00	0.00	0
235	0.00	0.00	0.00	0
240	0.00	0.00	0.00	0
250	0.00	0.00	0.00	0
255	0.00	0.00	0.00	0
260	0.00	0.00	0.00	0
270	0.00	0.00	0.00	0
310	5800.00	0.00	5800.00	0
320	6900.00	0.00	900.00	6000
408	40000.00	0.00	15000.00	25000
590	6100.00	0.00	6400.00	-300
430	4400.00	0.00	400.00	4000
470	0.00	0.00	0.00	0
515	5300.00	0.00	5300.00	0
520	500.00	0.00	100.00	400
585	0.00	0.00	0.00	0
620	0.00	0.00	0.00	0
630	0.00	0.00	0.00	0
*755	0.00	0.00	0.00	0
**912	450000.00	0.00	212000.00	238000

TOTALS	1592800.00	-9100.00	720900.00	862800.00



Save the last
column as a
DIF file.

Figure 2-21

LOAD DIF FILE AT B14

QUARTERLY BUDGET REPORT
INCLUDING PROJECTED EXPENSES
NUMBER: 848 PERIOD ENDING:

CODE	START OF QTR.	ADDITIONS REDUCTIONS	EXPENSES THIS QTR	END OF QTR

#110		0
#120		0
#141		0
142		0
145		0
150		0
152		0
155		0
160		0
#200		0
210		0
220		0
230		0
232		0
235		0
240		0
250		0
255		0
260		0
270		0
310		0
320		0
408		0
590		0
430		0
470		0
515		0
520		0
585		0
620		0
630		0
#755		0
#912		0

TOTALS		0.		

Load DIF file as directed. →

QUARTERLY BUDGET REPORT
INCLUDING PROJECTED EXPENSES
NUMBER: 848 PERIOD ENDING: 06/30/99

CODE	START OF QTR.	ADDITIONS REDUCTIONS	EXPENSES THIS QTR	END OF QTR

#110	100000.00	100000
#120	457900.00	457900
#141	50000.00	50000
142	-34000.00	-34000
145	335000.00	335000
150	0.00	0
152	0.00	0
155	0.00	0
160	-7000.00	-7000
#200	-26000.00	-26000
210	0.00	0
220	0.00	0
230	0.00	0
232	0.00	0
235	0.00	0
240	0.00	0
250	0.00	0
255	0.00	0
260	0.00	0
270	0.00	0
310	0.00	0
320	60000.00	60000
408	250000.00	250000
590	-3000.00	-3000
430	40000.00	40000
470	0.00	0
515	0.00	0
520	4000.00	4000
585	0.00	0
620	0.00	0
630	0.00	0
#755	0.00	0
#912	2380000.00	2380000

TOTALS	862800.00	0.00	0.00	862800.00

Figure 2-21 continued

Combining Sheets

DIF files can also be used to combine similar data from several sheets. For example, Figure 2-22 contains three sheets at the top that have proposed employee salary increases. These have been generated for members of the three departments by the managers of each department. Now the division director wants to combine the data for divisional review and totaling. That's done here by creating the DIF files indicated and recombining them as shown.

Computing Year-to-Date Totals

DIF files can be helpful in computing year-to-date totals on a VisiCalc worksheet. As an example, we'll assume that we have set up a worksheet with three columns, A, B, and C, and that the columns have titles as below.

A	B	C
New	Old	New
Period	YTD	YTD

On this worksheet, column C contains formulas that add together values from column A and B. Our updating process for the year-to-date computation now occurs correctly if we follow the steps below.

1. With the year-to-date values in column B (except for the current period), place the numeric values for our most recent period in column A. This can occur by entering the numbers by hand, or by loading a column of data from a DIF file into column A. This results in the automatic updating of column C.
2. Store column C as a DIF file.
3. Blank column A to prepare it for the end of the next quarter.
4. Load this DIF file (from column C) into column B, thereby updating column B, and preparing it for the next update.
5. Save the sheet.

By following this multistep process carefully at the end of each time period, we are able to compute year-to-date figures with VisiCalc.

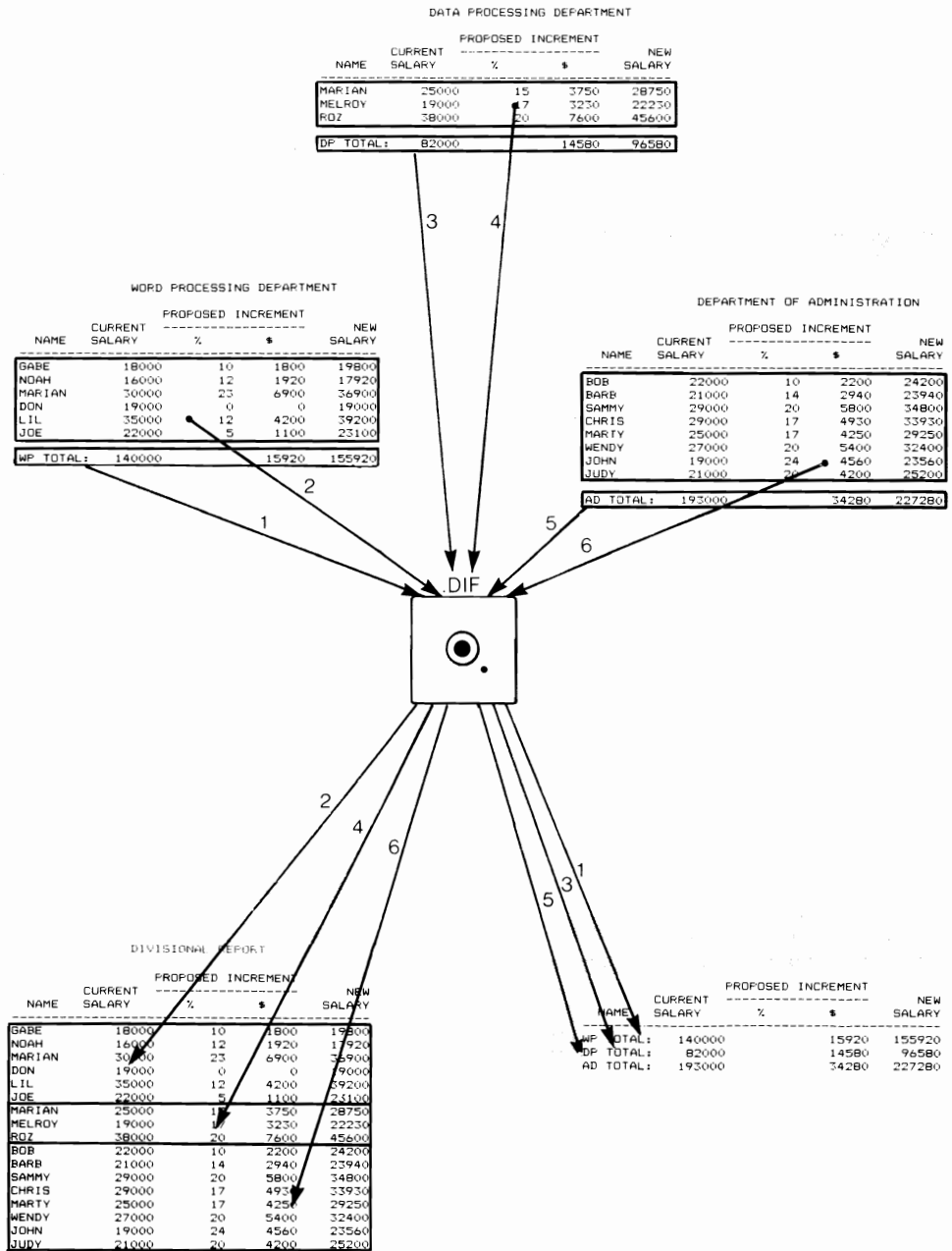


Figure 2-22

The same technique can be used to summarize and update a variety of similar kinds of data, for example survey results.

Communicating Data to Other Programs

For many reasons, we may want to use VisiCalc data with other programs. For example, suppose that we want to sort the names of the individuals shown in the combined sheet at the lower left of Figure 2-22. VisiCalc does not contain sorting capability; however, other programs do. If the other programs are written to accept DIF files, then we can have those programs read DIF files created elsewhere. For our example in Figure 2-23, we can create a DIF file from the data of Figure 2-22. We can move the name and new salary for each division member to a DIF file and set the file aside for possible use later.

DIF is the bridge, capable of assisting in the communication of data between two programs.

NAME	NEW SALARY
GABE	19800
NOAH	17920
MARIAN	36900
DON	19000
LIL	39200
JOE	23100
MARIAN	28750
MELROY	22230
ROZ	45600
BOB	24200
BARB	23940
SAMMY	34800
CHRIS	33930
MARTY	29250
WENDY	32400
JOHN	23560
JUDY	25200

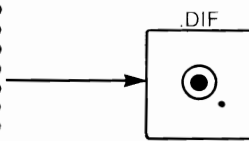


Figure 2-23

Chapter 3

Case Studies

NOTES ON THE DIF FORMAT CASE STUDIES

The concept of using a DIF file as a bridge to communicate data between different software applications was introduced at the end of Chapter 2, *Using the DIF Format in a VisiCalc Context*. In this chapter we'll examine that concept by studying examples of exchanging data via DIF files. To do so we will introduce a series of case studies that use the DIF format. Each study illustrates considerations involved in the interchange process.

Using specific commercial software packages for illustration makes the case studies real. The packages have been selected for one or more of a variety of reasons, including:

- their ability to use DIF files.
- product quality and popularity (as reflected by sales).
- considerations they illustrate regarding DIF file usage.
- the desire to use products from a variety of software producers.

Additional information about these products, and others that use DIF files, is contained in Appendix C, *Products That Use the DIF Format*.

For our purposes it's not important to have full technical knowledge of the DIF file format. We will work with programs that create and read DIF files, and thus we can leave the details to these programs. This is an important concept. As DIF file users, we do not need to learn the technical descriptions of the file format in order to use it successfully. However, we do need knowledge of the software packages we'll use, especially those elements of the product which involve the DIF format.

Studying the examples will provide insight into the process of data exchange even if the products discussed are not familiar to the reader. For each, specific details of the hardware and software used (version numbers, etc.) are provided, since the same software product often contains different features in different hardware implementations.

VisiCalc is used in a number of the case studies. Chapter 2, Using the DIF format in a VisiCalc Context, contains information about creating DIF files from VisiCalc and reading DIF files into VisiCalc. Those unfamiliar with that topic may wish to refer to Chapter 2.

Now for the case studies.

DATA INTERCHANGE BETWEEN VISICALC AND VISITREND/PLOT

In this section we'll study a number of examples of the data interchange process between VisiCalc and VisiTrend/Plot, using the DIF file format, with data moving in both directions. The recent growth in interest in business graphics programs on microcomputers has made this type of data exchange, from a spreadsheet program to a plotting program, more common. We'll work through enough examples in this and the next section (Data Interchange Between VisiCalc and PFS:Graph) to demonstrate the considerations involved in this interchange.

The examples in this section were done using an Apple II Plus microcomputer with 64K of memory, VisiCalc version VC-202B0-AP2, VisiTrend/Plot version 1.00, with printed/plotted output sent to an EPSON MX-100 printer via a Grappler Interface card. A one-line BASIC program contained in the Grappler manual has been used to print the graphs created by VisiTrend/Plot on the printer. (Trademark information is contained in the front matter of this book and product information is in Appendix C.)

CASE STUDY 1 VisiCalc to VisiTrend/Plot Plotting Quarterly Production Data

First, let's set up an example. Figure 3-1 displays a VisiCalc screen showing production figures that we have generated for each of the four quarters of the year.

Suppose we wish to have a graph prepared from this data, and that we ask an artist to prepare it, rather than use a computer graphing program. For that purpose, we've printed a report from VisiCalc of our data and made notes on it to our artist, who has produced the graph. This process is shown in Figure 3-2.

Notice that the artist knew to ignore the total and the underline above it. Blank areas of the report have also been ignored. Titles were reworked by the artist.

Now let's turn to VisiTrend/Plot and prepare a plot by passing the data from VisiCalc with a DIF formatted file. VisiTrend/Plot includes a "graph plotting system" that generates line, bar, area, pie, hi-lo, and scatter charts. It is a stand-alone product that allows users to enter data within the program, but that also can use data in a DIF format generated by programs which create DIF files.

When we want to communicate the data from our VisiCalc example to VisiTrend/Plot, we'll be responsible for creating a DIF file with the four data points. We'll ignore the blank rows, columns, and entries. We'll

	A	B	C	D
1				
2	PRODUCTION FORECAST			
3	BY QUARTER			
4				
5	QTR	PROD		
6	-----			
7	FIRST	15		
8	SECOND	19		
9	THIRD	21		
10	FOURTH	27		
11		=====		
12	TOTAL	82		
13				
14				
15				
16				
17				
18				
19				
20				

Figure 3-1

also ignore the underlines and the column total. In fact, for this example, we will communicate only numbers, not labels, via the DIF file.

At this point we need some basic information about plotting and VisiTrend/Plot. Two-dimensional plotting involves an X (horizontal) and a Y (vertical) coordinate, meaning that every data point plotted actually involves two discrete numeric values. VisiTrend/Plot plots in two dimensions, using the X-Y axes, where, for most applications, the X axis is a time axis. However, VisiTrend/Plot uses an unusual technique to generate the X value of the coordinate. The X value is assumed to be "time," and a starting time and an increment, computed from an item called the period, are used to derive X coordinates.

For example, if we start our data in 1900 and increment by one, our X coordinates become

1900
 1901
 1902
 1903
 .
 .
 .

with each year being assigned sequentially to our series of Y values. We reach our last year when we exhaust our Y values.

PRODUCTION FORECAST BY QUARTER	
QTR	PROD
FIRST	15
SECOND	19
THIRD	21
FOURTH	27
TOTAL	82

Artist -
 Can you please
 prepare a graph from
 this data? Label
 as you wish.
 Don

Don:
 How's
 This? A.

QUARTERLY PRODUCTION FORECAST - 1999

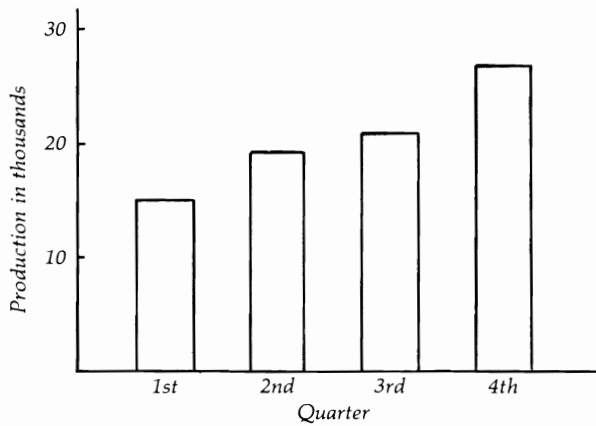


Figure 3-2

As another example, if we start in September of 1985 and increment monthly, we have X coordinates of

September, 1985
October, 1985
November, 1985
December, 1985
January, 1986
.
.
.

again, until we exhaust the Y values. Suppose we had data for 60 months in this example. Normally we would require 120 values to graph this, two values for each point to be plotted. However with VisiTrend/Plot we need less data. We need the 60 data values for the Y axis, but only three pieces of data for the X axis (starting month, starting year, and period).

Let's return to our VisiCalc data. It is quarterly, and we will assume that it starts in the first quarter of 1985. In VisiTrend/Plot terms it has a major start date of 1985, a minor start date of 1 (indicating the first quarter), and a period of 4 (indicating quarterly data).

Figure 3-3 illustrates how we'll create the desired DIF file from VisiCalc. The commands are shown in that figure, and we see that the newly created file is stored on a diskette. We've named the file

QTRC.DIF

(The naming of DIF files is discussed in Chapter 2, Using the DIF Format in a VisiCalc Context.)

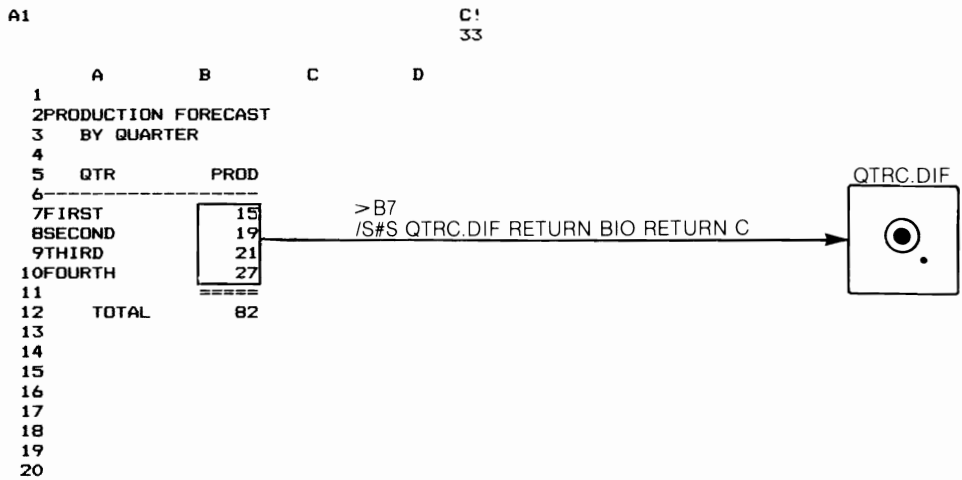


Figure 3-3

We now leave VisiCalc (/SQ...) and load VisiTrend/Plot.

Once in VisiTrend/Plot, we'll place our diskette with the DIF file in the disk drive, and use the LOAD option to load the DIF file. After selecting LOAD, VisiTrend/Plot displays a directory (a listing of the names of files on the data diskette). We select

QTRC.DIF

(the name of our quarterly forecast file from VisiCalc), and then respond to the prompts from VisiTrend/Plot as shown below.

VisiTrend/Plot Prompts	Our Response	Explanation
PERIODICITY	4 RETURN	Our data is quarterly, indicating a period of 4 within VisiTrend/Plot.
MAJOR START (YEAR)?	1985 RETURN	Our data begins in 1985.
MINOR START (PERIOD)?	1 RETURN	Our data begins in the first quarter of the year.

Notice that within VisiTrend/Plot we had to enter the date and period information as we loaded the DIF file. We cannot pass this data to VisiTrend/Plot in a file generated by VisiCalc. The *DIF Technical Specification* (Appendix A) specifies that a program which needs information not contained in the DIF file it reads should prompt the user for the data, as VisiTrend/Plot does here.

VisiTrend/Plot then reads the data from the DIF file and captures the data in an internal format which allows the program and user to create charts. The interchange of data is complete, for VisiTrend/Plot has successfully received the four data points. DIF has served its purpose here.

Figure 3-4 illustrates a variety of charts produced from these four values using VisiTrend/Plot. Notice that they contain labels similar to those in the VisiCalc screen of Figure 3-3. However, it's important to realize that these labels were entered through the labeling options or defaults of VisiTrend/Plot, and were *not* transferred through the DIF file.

Let's briefly summarize the steps we follow to create our data, exchange these few data points, and plot the data. We'll assume a one-disk drive computer system. A system with two disk drives will simplify this process since the data diskette could be left in place. A system with

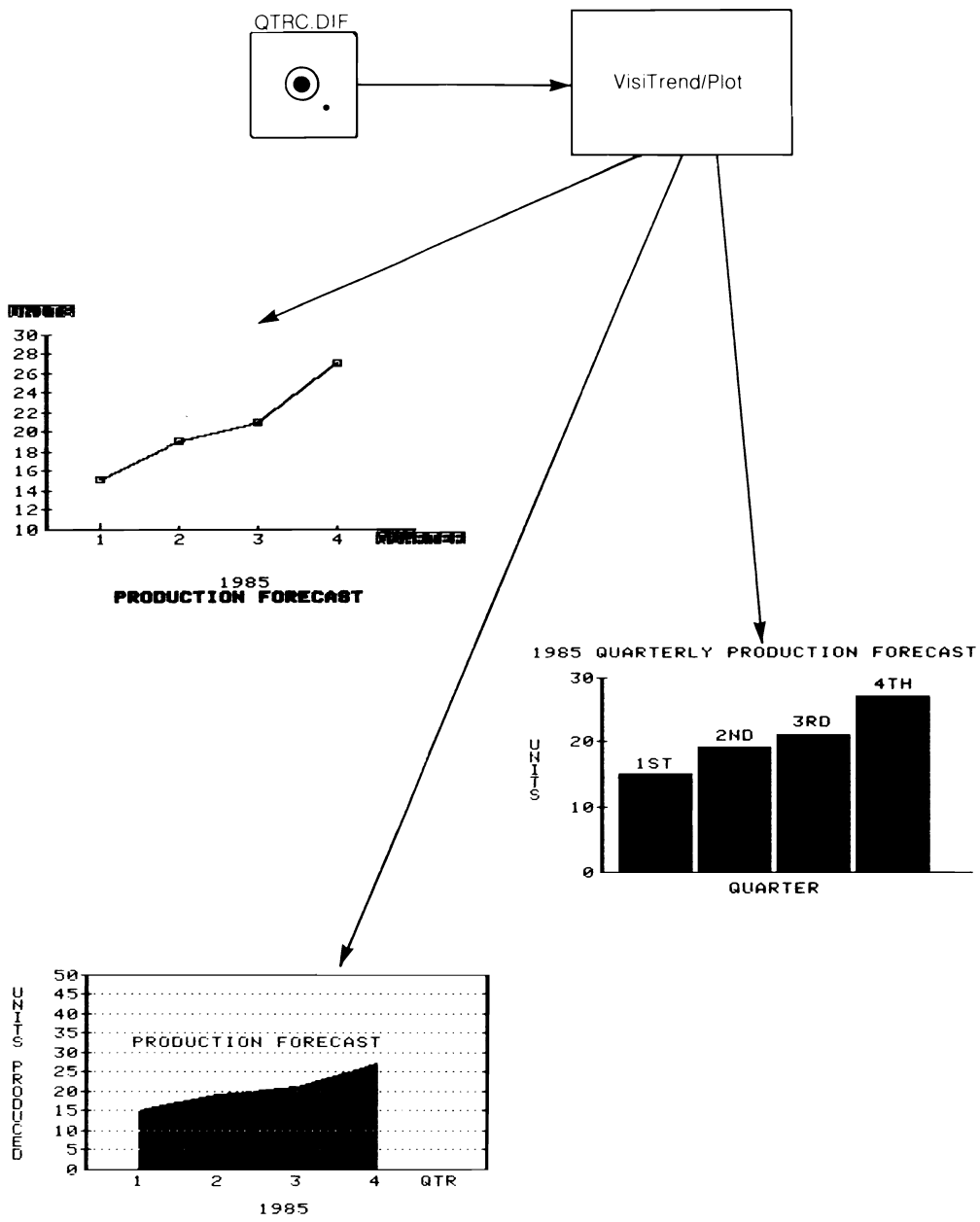


Figure 3-4

multiple drives attached to separate slots or a hard disk could further simplify this process since diskettes for several programs as well as data could be left in place.

- Step 1** Insert the VisiCalc diskette, run this program, and remove the diskette.
- Step 2** Generate the data as desired on the VisiCalc worksheet.
- Step 3** Insert a data diskette.
- Step 4** Save our data points to a DIF file on the data diskette, and remove the data diskette.
- Step 5** Insert the VisiTrend/Plot diskette.
- Step 6** Quit the VisiCalc program, causing VisiTrend/Plot to load and run.
- Step 7** Insert the data diskette and load the DIF file, supplying date and period information.
- Step 8** Generate and print the desired charts.

It's clear that this is a multiple step process. If we expect to use these four data points again within the VisiTrend/Plot environment, we should add an additional step.

- Step 9** Create a new file from VisiTrend/Plot in the VisiTrend/Plot format with the four data points and the date and period information.

This last step illustrates that the DIF format is for exchange, but it's not necessarily the best way to store data. Here, if we reload this DIF file into VisiTrend/Plot, we'll need to repeat part of Step 7, entering date and period information. But if we reload the file created by VisiTrend/Plot in Step 9, we will not need to re-enter the date and period. VisiTrend/Plot includes this data in its own file format.

VisiTrend/Plot can be used to create a DIF file from this data, and in doing so it will place this "extra" information (dates and period) into the DIF file using the optional aspects of the file. If this second DIF file is read into VisiTrend/Plot, the user will not need to re-enter the date and period information.

Let's examine another example of this interchange, as shown in Figure 3-5. We've built a quarterly production forecast based on three different assumptions about the economy, labeled ECON A, ECON B,

	A	B	C	D
1				
2	QUARTERLY PRODUCTION			
3	BASED ON THREE ECONOMIC FORECASTS			
4				
5	QTR	PRODUCTION		
6		ECON A	ECON B	ECON C
7	FIRST	15	20	30
8	SECOND	19	22	38
9	THIRD	21	23	42
10	FOURTH	27	29	41
11				
12	TOTAL	82	94	151
13				
14				
15				
16				
17				
18				
19				
20				

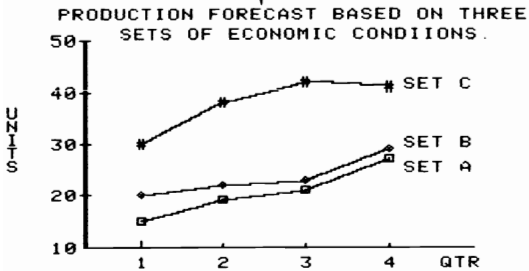


Figure 3-5

and ECON C. We then create a DIF file in column order from the three columns indicated in the VisiCalc screen in the figure. We have three columns of data. Within VisiTrend/Plot, each column becomes a separate series of data, and we can work with each separately, or in various combinations.

Also, when VisiTrend/Plot encounters a label in the first entry (as it does here with label ECON A in position B6), it uses the label as a series name. In our earlier example, we had a numeric value in our first field and the series was given a default name of SERIES0. Here however, our three strips of data will automatically be named ECON A, ECON B, and ECON C within VisiTrend/Plot. This convention (naming series by the first entries of the columns or rows) is one that has been adopted by VisiTrend/Plot, and is unrelated to the *DIF Technical Specification*.

Figure 3-5 also shows a graph created from this data.

Let's look at another example, as shown on the VisiCalc screen in Figure 3-6 where we have expanded our spreadsheet to include monthly as well as quarterly data. Suppose that we create a DIF file from the data area marked in column B.

We'll read this file into VisiTrend/Plot and produce a line chart from the data, as shown in the graph of Figure 3-6. Notice the spikes. The top of each is one of the quarterly totals. The bottoms of the spikes, where the line touches the X axis at zero, are the blank areas and underlines from VisiCalc column B.

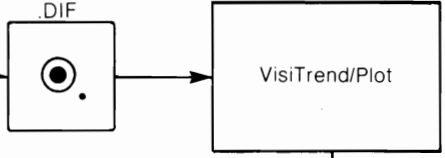
In most cases when VisiTrend/Plot encounters non-numeric data items in a DIF file, it converts them to the value zero. That's why we see the line touching the X axis in the valleys of the chart. (When VisiPlot encounters a label in the *first* entry a different action occurs, as described earlier in this example.) Also notice that the X axis default labels are all incorrect.

For this data exchange, the data must be manipulated considerably, either within VisiCalc or within VisiTrend/Plot, to eliminate all of the extraneous data from the plot. We'll see that this need for data manipulation occurs often in data interchange.

A1

C!
32

1	A	B	C	D
2	PRODUCTION FORECAST			
3	BY MONTH AND QTR.			
4				
5	MON/QTR	PROD		.DIF
6	-----			
7	JAN	5		
8	FEB	4		
9	MAR	6		
10	-----			
11	1ST QTR	15		
12				
13	APR	7		
14	MAY	6		
15	JUNE	8		
16	-----			
17	2ND QTR	21		
18				
19	JULY	9		
20	AUG	10		
21	SEPT	12		
22	-----			
23	3RD QTR	31		
24				
25	OCT	12		
26	NOV	14		
27	DEC	16		
28	-----			
29	4TH QTR	42		
30				
31	=====			
32	TOTAL	109		
33				
34				
35				



Values received within
VisiTrend/Plot

16 0 42 0 0 109



Figure 3-6

VisiCalc has powerful numeric manipulation features which allow us to semi-automate removal of the extra data. This important use of VisiCalc is demonstrated in Figure 3-7. In that figure we establish a separate area at the right of the spreadsheet of the previous example. Formulas are placed at entries E11, E12, E13, and E14 as below.

Entry	Formula at this entry
E11	{B11}
E12	{B17}
E13	{B23}
E14	{B29}

Some VisiCalc users may prefer to write "+B11" instead of "{B11}".

These four formulas automatically "copy" the quarterly subtotals to a compressed area of the sheet from which we can create the DIF file without including any extraneous data.

Also notice that

```
>E10 RET
/S#S QTRC.DIF
RET E14 RET C
```

appears above the column of quarterly data. These three lines of label data serve as directions for us to follow to create the DIF file from the four quarterly data points. VisiCalc can not automatically execute these "commands," but their appearance on the sheet is important documentation, for if we use this sheet repeatedly we will not need to rethink the steps necessary to recreate the DIF file each time.

CASE STUDY 2

VisiTrend/Plot to VisiCalc Manipulating Plot Data in VisiCalc

We'll continue with the example above, but this time move data from VisiTrend/Plot to VisiCalc, the reverse of Case Study 1.

Suppose that we have entered our monthly (not quarterly) data into VisiTrend/Plot and that we will move the data to VisiCalc so that we can compute quarterly totals or do other manipulation of the data. As a final step, in the next case study, we'll send the quarterly sums back to VisiTrend/Plot from VisiCalc.

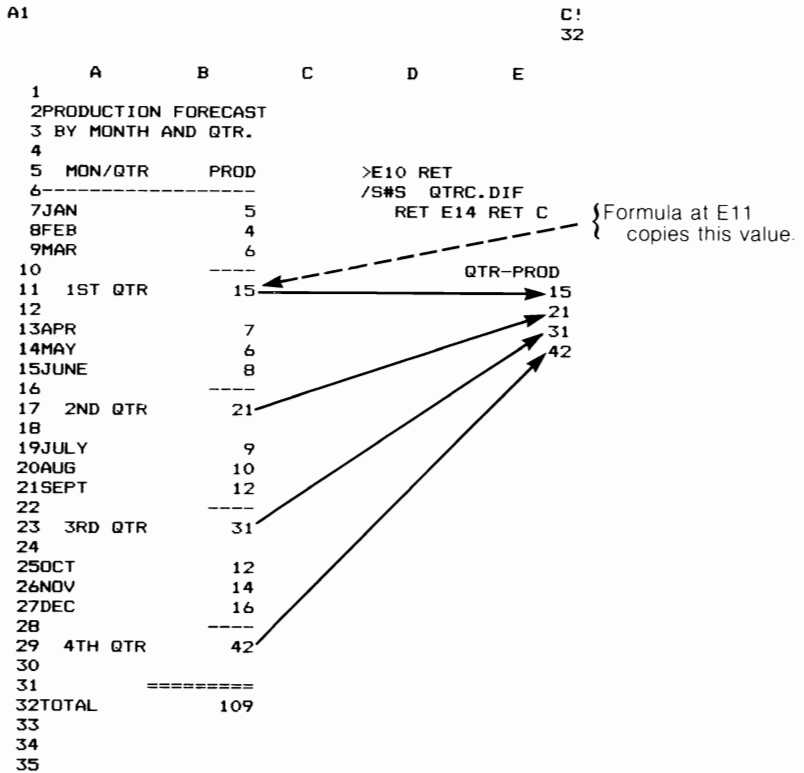


Figure 3-7

We'll assume that we have entered a series of monthly production numbers using the EDIT option of VisiTrend/Plot. We will use the VisiTrend/Plot PRINT option to list our data, as shown in Figure 3-8.

DATE	MO-PROD
1985 1	5
2	8
3	9
4	7
5	9
6	10
7	12
8	12
9	12
10	15
11	17
12	14

Figure 3-8

VisiTrend/Plot retains the major start date (1985), the minor start date (1 for January), and the period (monthly) in its file structure, as well as the production values of the second column of Figure 3-8. It does not store names of the months with the file.

We'll use the SAVE option of VisiTrend/Plot, select the DIF format, and save the file. When we create the DIF file, we'll get the two start dates and the period in the DIF file; however, they are there as optional fields. We will also get the 12 monthly values.

We then leave VisiTrend/Plot, enter VisiCalc, and load the file. VisiCalc, in reading the DIF file, ignores the optional fields with the start dates and period, and only places the 12 values on the worksheet.

Figure 3-9 illustrates a VisiCalc worksheet that will help us do this. On the top half of that figure we have the sheet *before* we load the file. Notice we load the file at the left of the worksheet, and as we do so the values at the right are automatically computed. This is demonstrated in the bottom of Figure 3-9 which shows the sheet *after* we have loaded the DIF file created from VisiTrend/Plot.

We have really divided the spreadsheet into two areas:

- A data communication area at the left of the sheet where we'll load our DIF file, the data from VisiTrend/Plot.
- A computation/forecasting area at the right where our data is redistributed automatically into appropriate entries of the sheet as illustrated by the arrows.

Before loading DIF file.

A1 C!
32

	A	B	C	D	E
1					
2	>B5	RET		PRODUCTION FORECAST	
3	/S#L	MO-VP.DIF	RET C	BY MONTH AND QTR.	
4					
5	JAN			MON/QTR	PRD
6	FEB			-----	-----
7	MAR			JAN	0
8	APR			FEB	0
9	MAY			MAR	0
10	JUNE			-----	-----
11	JULY			1ST QTR	0
12	AUG				
13	SEPT			APR	0
14	OCT			MAY	0
15	NOV			JUNE	0
16	DEC			-----	-----
17				2ND QTR	0
18					
19				JULY	0
20				AUG	0
21				SEPT	0
22				-----	-----
23				3RD QTR	0
24					
25				OCT	0
26				NOV	0
27				DEC	0
28				-----	-----
29				4TH QTR	0
30					
31				TOTAL	0
32				=====	=====
33					
34					
35					

DIF file loads here

After loading DIF file.

A1 C!
32

	A	B	C	D	E
1					
2	>B5	RET		PRODUCTION FORECAST	
3	/S#L	MO-VP.DIF	RET C	BY MONTH AND QTR.	
4					
5	JAN		5	MON/QTR	PRD
6	FEB		8	-----	-----
7	MAR		9	JAN	5
8	APR		7	FEB	8
9	MAY		9	MAR	9
10	JUNE		10	-----	-----
11	JULY		12	1ST QTR	22
12	AUG		12		
13	SEPT		12	APR	7
14	OCT		15	MAY	9
15	NOV		17	JUNE	10
16	DEC		14	-----	-----
17				2ND QTR	26
18					
19				JULY	12
20				AUG	12
21				SEPT	12
22				-----	-----
23				3RD QTR	36
24					
25				OCT	15
26				NOV	17
27				DEC	14
28				-----	-----
29				4TH QTR	46
30					
31				TOTAL	130
32				=====	=====
33					
34					
35					

Figure 3-9

CASE STUDY 3

VisiTrend/Plot to VisiCalc to VisiTrend/Plot

Let's complete this series of graphics case studies by taking data entered with VisiTrend/Plot, communicate it to VisiCalc so that we can manipulate it in a way not available in VisiTrend/Plot, then send resultant data back to VisiTrend/Plot for plotting.

The steps we'll need to follow for this process are

- Step 1** Run VisiTrend/Plot.
- Step 2** Enter data using the EDIT option and create charts as desired.
- Step 3** Create a DIF file from VisiTrend/Plot.
- Step 4** Quit VisiTrend/Plot.
- Step 5** Run VisiCalc.
- Step 6** Load the DIF file.
- Step 7** Manipulate data as necessary.
- Step 8** Create a second DIF file from the new data.
- Step 9** Quit VisiCalc.
- Step 10** Run VisiTrend/Plot (again).
- Step 11** Load the second DIF file.
- Step 12** Create charts as desired.

Let's look at the middle of this process (steps 6, 7, and 8) in Figure 3-10. That figure shows a VisiCalc worksheet, here divided into three areas that we've identified as STEP 6, STEP 7, and STEP 8 to correspond to the steps above. The areas are:

- Step 6 area** *receives* data via DIF from some outside source, here VisiTrend/Plot.
- Step 7 area** *manipulates* data using the capabilities of VisiCalc.
- Step 8 area** *sends* data via DIF to an outside program, here VisiTrend/Plot.

The "what if..." capabilities of VisiCalc are used in the lower section of the middle area (Step 7) of this sheet with results automatically carried forward to the right (Step 8) for later DIF file creation.

STEP 6

STEP 7

STEP 8

A1

C!
30

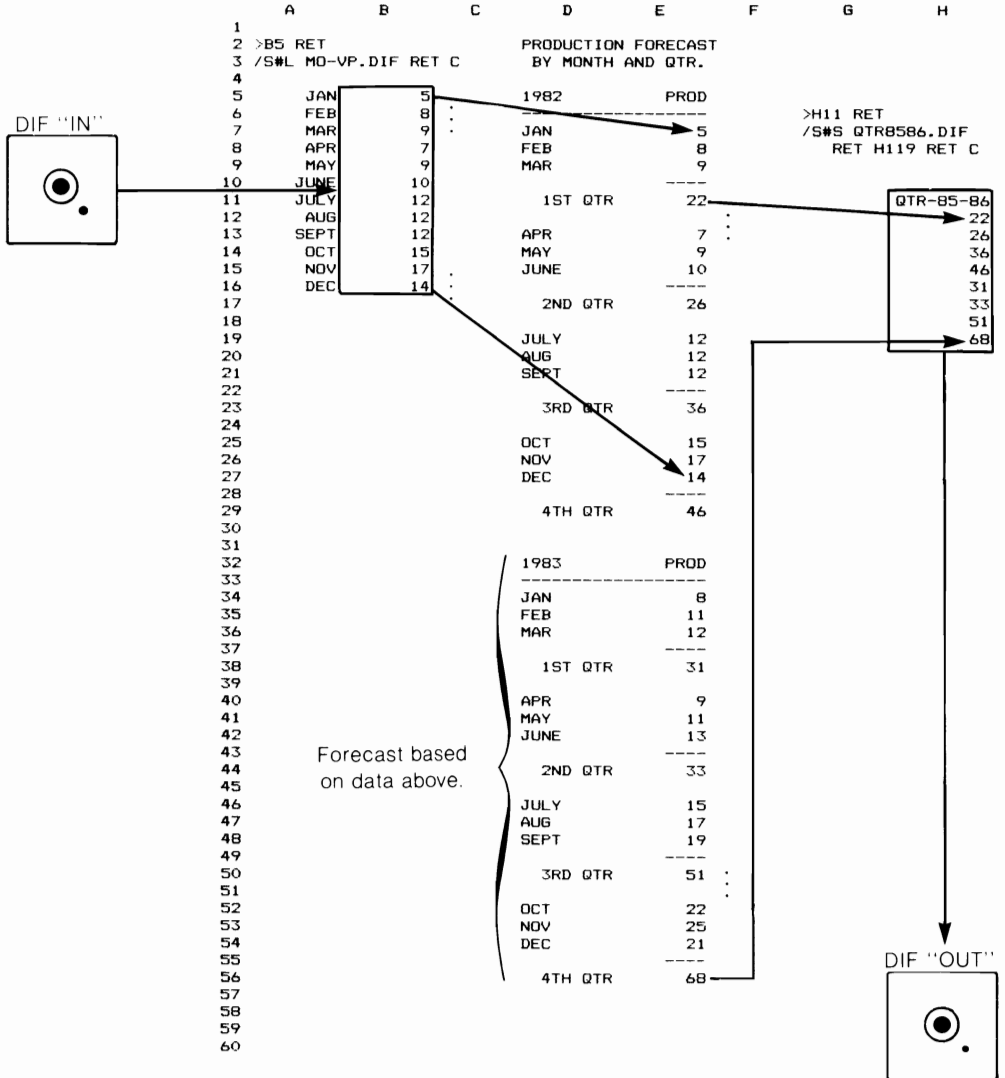


Figure 3-10

This is a powerful, utility-like use of VisiCalc. The capabilities of that product allow this numeric on-sheet copying to occur if sheets are set up as described. It is important to realize that this capability works with numbers, not with labels, and that the order of calculation may become an important factor in spreadsheets with this organization.

Additional Factors in the VisiCalc to VisiTrend/Plot Interchange of Data

There are additional considerations when data is exchanged between these two products. The items below, although specific for these two products, are included as illustrations of what occurs in any data interchange.

1. We have seen that VisiTrend/Plot creates DIF files that do not contain two coordinates for each data point. Instead, VisiTrend/Plot uses one-dimensional data. When the DIF file is created, we get the start dates (major and minor), increment, and then the Y coordinate data points.

When VisiCalc reads the DIF file, it ignores the dates and increment, and only reads the data points. This means VisiCalc only gets Y axis data.

2. Early versions of VisiTrend/Plot created DIF files incorrectly. (Refer to the Applications Programs section of the *DIF Technical Specification* reprinted in Appendix A.)

3. When we create a DIF file from VisiCalc (or from any program), our data must be a rectangle. This causes inconveniences in some cases, as shown in Figure 3-11. We want to send two vertical strips (series) of points to VisiTrend/Plot for plotting. If we save from

E10 to F22

as a DIF file, we get a rectangle of data, and all blank entries from E15 to E22 have the value 0.

Several steps later, when loading these two columns into VisiTrend/Plot, we get two series of numbers, each with 12 data points, even though we only want four points in the first. Both will be given the same period within VisiTrend/Plot.

For this reason, we'll need to add a step somewhere in our process. We can

- act within VisiCalc to create two DIF files (instead of one) as shown in Figure 3-12.
- act within VisiTrend/Plot (with the EDIT option) to delete the extra zeros from the end of the quarterly series and also change the period of one series.
- act within VisiTrend/Plot to ignore the zeros at the end of the series and only plot the first four values by rescaling the X axis.

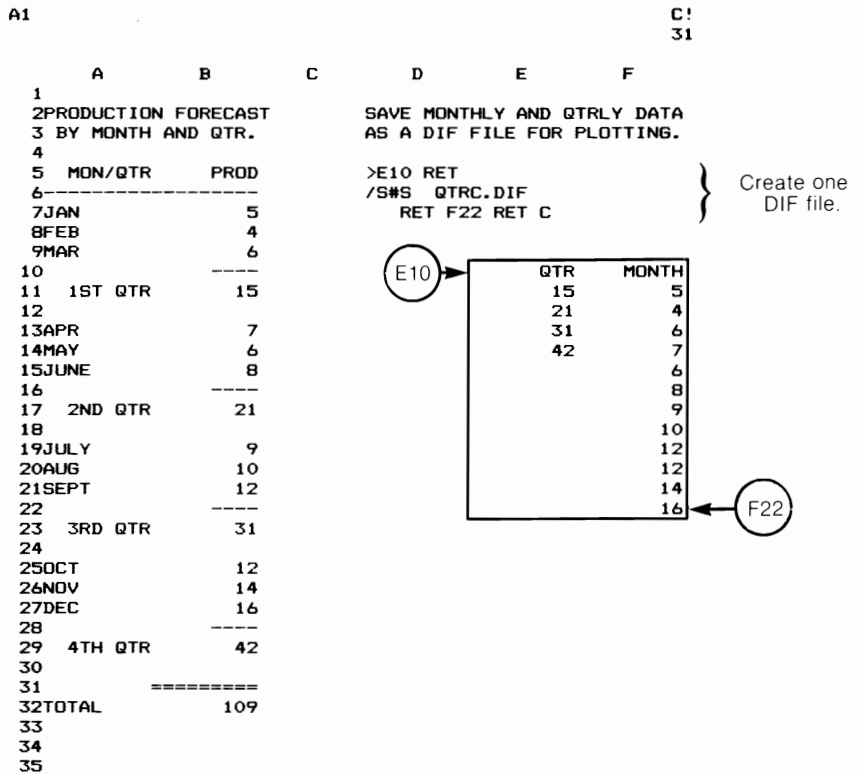


Figure 3-11

4. When we read a DIF file into VisiTrend/Plot that has been created from VisiCalc, we are asked for major (and minor) start dates and for the period. VisiCalc does not allow us to build these into the DIF file. We must be prepared to enter them when prompted to do so by VisiTrend/Plot, meaning they must be available. Making this interchange in one step may not be a problem; however, if we complete the interchange a week later, we may have forgotten this information.

For that reason, it is suggested that we document the interchange, for example by printing the data from VisiCalc. We'll return to this topic later in Chapter 5, Documenting Data Interchange.

5. At a later time, if we reload the same DIF file into VisiTrend/Plot, we'll again be presented with the same prompts, since VisiTrend/Plot does not store a file unless we direct it to. Therefore, if we expect to use

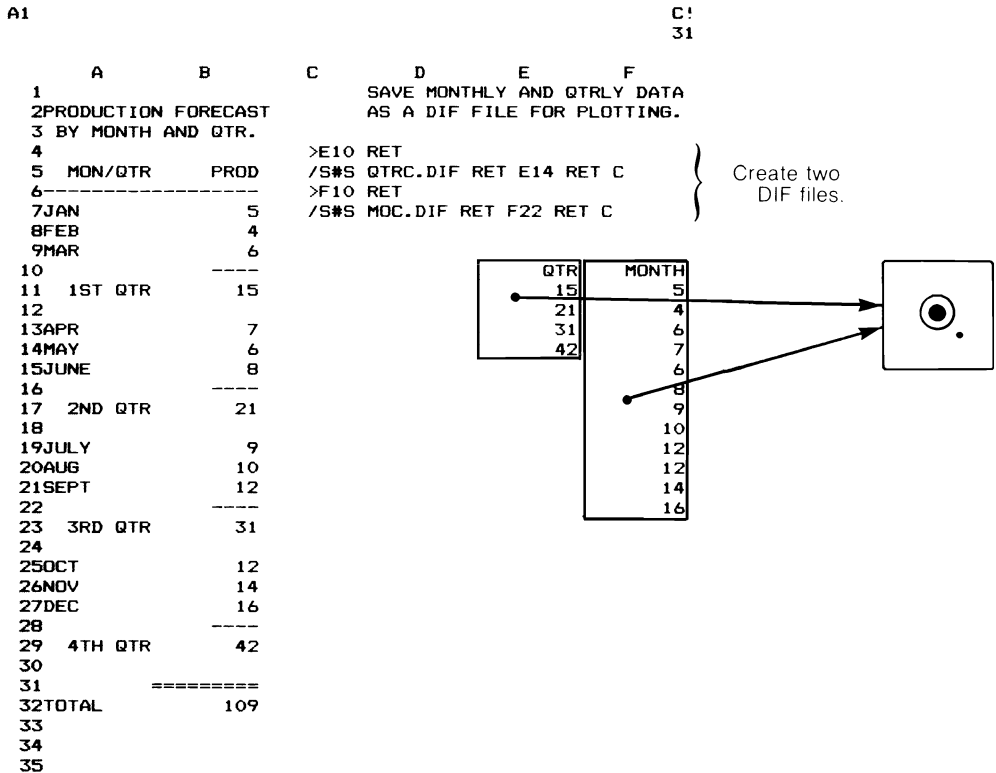


Figure 3-12

data repeatedly from the same DIF file, we should take the extra step to save the data as a VisiTrend/Plot file. When we need the data again, we'll reload the VisiTrend/Plot file, thereby eliminating the need to reenter the dates and period, and at the same time storing the data more efficiently.

6. DIF files created from the same set of numbers by VisiCalc and by VisiTrend/Plot will be different. Each data item will be the same in the DIF file, but VisiTrend/Plot will use optional entries of the DIF format to store start dates and the period. This is discussed in greater detail in Chapter 7, Limitations of a System Using the DIF Format.

7. The Kapor article and the VisiTrend/Plot manual by Ewing, both referenced in the Bibliography on the DIF Format, contain additional necessary information about this interchange.

DATA INTERCHANGE BETWEEN VISICALC AND PFS:GRAPH

PFS:Graph accepts data “from the keyboard or can retrieve data from VisiCalc or PFS files...(and allows users to organize) data into one of three types of graph—bar, line, or pie.”

In these case studies we'll begin with several of the same VisiCalc worksheets used in the last chapter. Our purpose is the same, that is, plotting VisiCalc data. We'll see that our *processes* will be similar, but our data will be quite different.

The case studies in this section were done using VisiCalc version VC-202B0-AP2; PFS:Graph, Rev. A; an Apple II Plus computer with 64K bytes of memory; and an EPSON MX-100 printer with a Grappler interface card. (Refer to the front matter for trademark information and to Appendix C for product information.)

CASE STUDY 4 VisiCalc to PFS:Graph

Let's begin with the production forecast of Figure 3-13, and again assume that we want a graph of the four quarterly values. For this example we'll produce a bar chart and a pie chart. PFS:Graph is able to use either numeric values, dates, or identifiers as X coordinates, where an identifier is a label such as FIRST, BOB, etc. Therefore, when we create the DIF file we can send the rectangle highlighted in Figure 3-13. This contains two columns, the first (column 1) with the X data coordinates (here identifiers), and the second (column 2) with the Y data (here numeric values).

Again, our purpose is not to provide a PFS:Graph tutorial, but rather to concentrate on the interchange; however, we'll need to have a basic understanding of this graph program to continue. This product stores full X and Y coordinates, where the Y coordinate is always a numeric value, and the X axis can be a value, an identifier (as in our example of Figure 3-13), or a date. We'll see examples of all types of X data in these case studies.

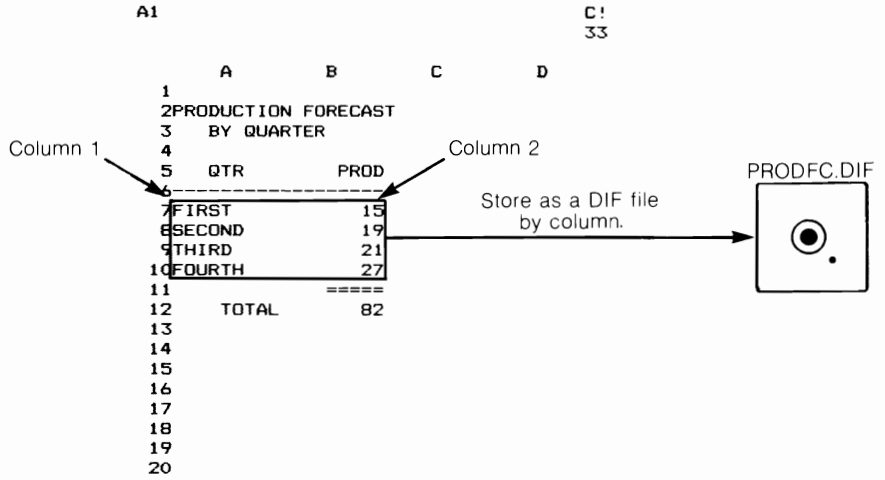


Figure 3-13

We now leave VisiCalc and run PFS:Graph.
Within PFS:Graph we select

GET/EDIT DATA

from the main menu, and then

GET VISICALC FILE

from the next menu. At this point we must supply responses for each of four prompts. The prompts and our responses are

Prompts	Our response	Notes
FILENAME:	PRODFC.DIF	
X DATA:	1	the number one
Y DATA:	2	
X DATA FORMAT:	I	the letter "eye"

Our responses include the filename, then numbers for the relative location of the row (or column) for our X and Y data, and finally the type of data in the X column (here I for identifiers).

The interchange is now complete. We have successfully entered data into PFS:Graph via the DIF format. Within PFS:Graph we can enhance graphs with titles, etc., and print them as illustrated in Figure 3-14. We have *not* had to enter the labels FIRST, SECOND, THIRD, and FOURTH since those were transmitted via the DIF file.

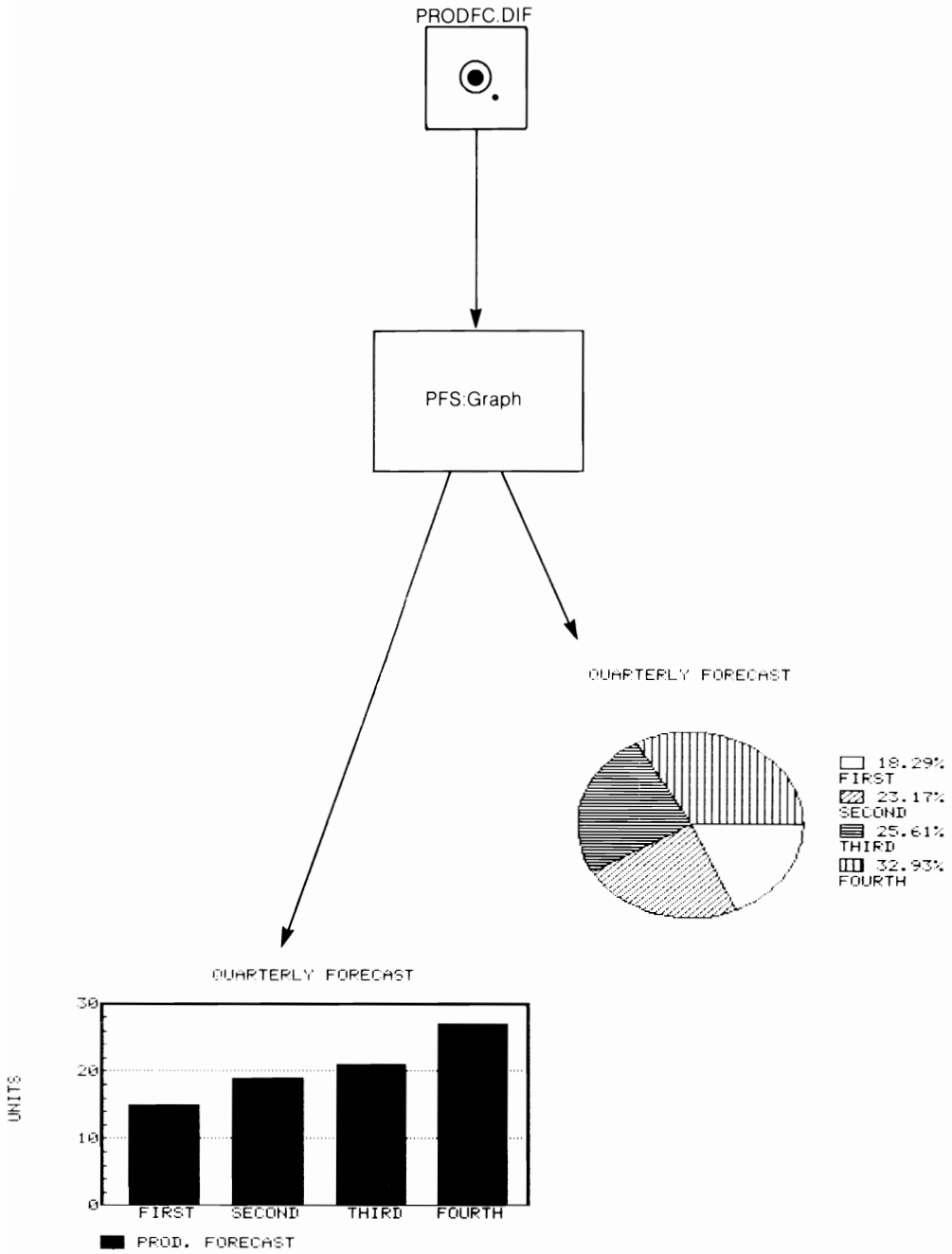


Figure 3-14

Let's stop for a moment to compare the process of the interchange from VisiCalc to VisiPlot with VisiCalc to PFS:Graph. We're comparing the interchange, not the two graphing products.

VisiCalc to VisiPlot

A one-dimensional set of numbers, or multiple one-dimensional sets of numbers, are exchanged via DIF files, with the user responsible for entering start dates and time period within VisiPlot before the interchange is complete.

If VisiPlot encounters non-numeric data, it is converted to the value zero (with some exceptions as described earlier in this chapter).

VisiCalc to PFS:Graph

Two-dimensional data is interchanged, with one row (or column) serving as the X axis and a second as the Y axis. Here the user is responsible for indicating which row (or column) is the X axis and which the Y axis, and for entering the data format (number, date, or identifier) of the X axis.

It is significant to recognize that although both products accept DIF files they require different data to successfully create a graph. They cannot each correctly work with some DIF files used as input to the other.

DIF is indeed the bridge, but the data crossing over it may be different for individual products even though those products have the same purpose, for example, graphing.

Let's work again with a problem from the last chapter, as shown in Figure 3-15. Here we'll create a DIF file from the two columns. We'll read this into PFS:Graph and provide the response to the prompts generated by the program. However, before the interchange is complete, we get the message below.

```
BAD Y DATA
4 DATA POINTS READ
```

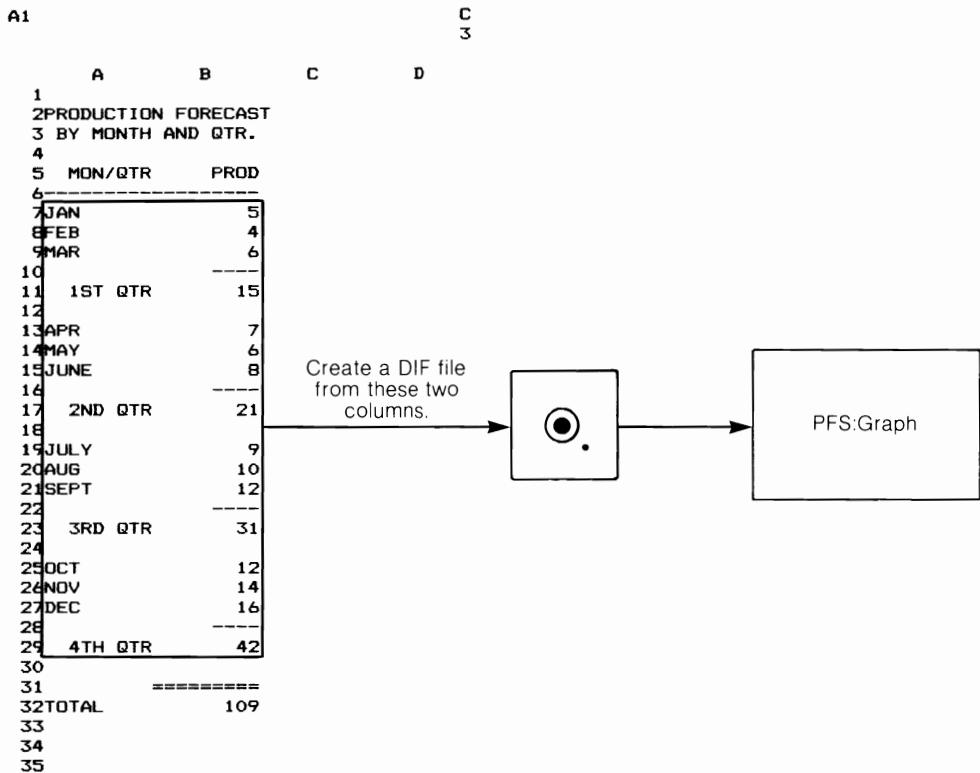


Figure 3-15

The program is telling us that data in the Y axis (column B in Figure 3-15) is invalid. The underline from the VisiCalc spreadsheet at location B10 is not a valid value for PFS:Graph. All Y data in PFS:Graph must be numeric.

We must return to VisiCalc and manipulate our data, for example by deleting the rows with the invalid data.

An alternative is to use VisiCalc to establish an area for data transfer as shown in Figure 3-16 and Figure 3-17, and as discussed at the end of Case Study 1. Notice that we now save two columns, not one as in a similar example in Case Study 1. We have entered the labels for the data in column D of those worksheets.

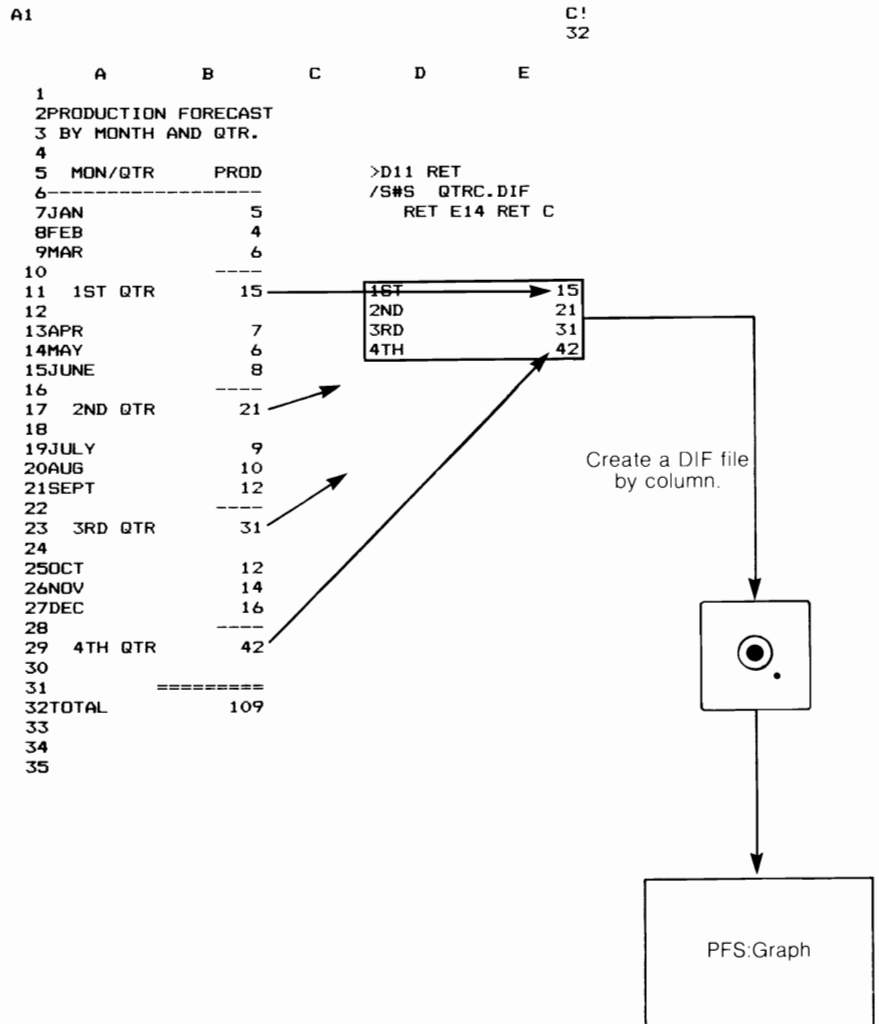


Figure 3-16

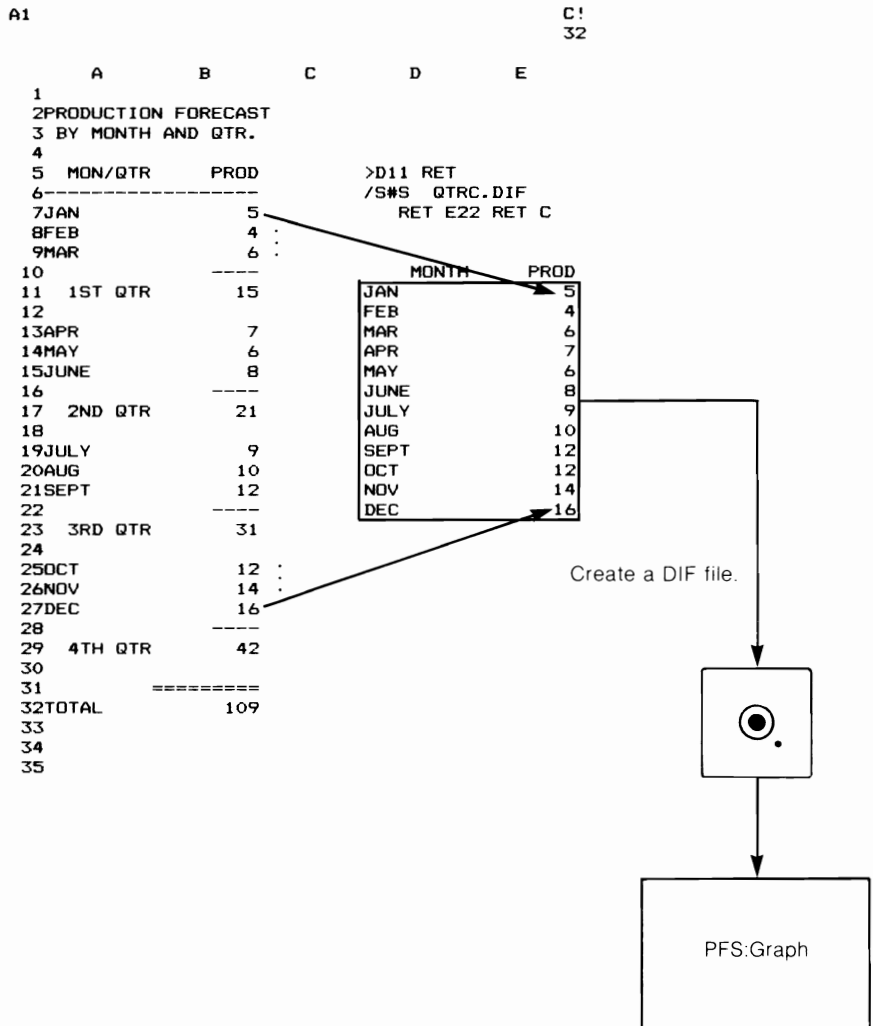


Figure 3-17

CASE STUDY 5 VisiCalc to PFS:Graph Exchanging Date Information

Let's continue with an example illustrating another way to send date information to PFS:Graph. In Figure 3-18 dates appear in column D of the VisiCalc sheet in the unusual format

MM=QQ=YY

that is, month, quarter, and year, each separated by an equal sign. PFS:Graph can accept date data consisting of day, month, quarter, and year in any order with fields separated by non-numeric characters. An equal sign has been used here so there is no confusion with the usual date format.

This date information was entered as label information within VisiCalc, that is

To enter	In VisiCalc type
1=1=85	"1=1=85
2=1=85	"2=1=85
.	.
.	.
.	.

Once we reach PFS:Graph we can manipulate our series of data points based on this date designation. For example, we could use all 24 points (ignoring the quarter) as shown in Figure 3-19, add data together cumulatively by month as shown in Figure 3-20 (again ignoring the quarter), compare quarters as shown in Figure 3-21 (ignoring the month), or look only at yearly data (ignoring the month and quarter), as shown in Figure 3-22. These are all created based on manipulation of the same data within PFS:Graph.

	A	B	C	D	E
1					
2	PRODUCTION FORECAST				
3	BY MONTH AND QTR.				
4					
5	MON/QTR	PROD		>D11 RET	
6	-----			/S#S QTRC.DIF	
7	JAN	5		RET E34 RET C	
8	FEB	4	:		
9	MAR	6	:		
10		----		MM=QQ=YY	PROD
11	1ST QTR	15		1=1=85	5
12				2=1=85	4
13	APR	7		3=1=85	6
14	MAY	6		4=2=85	7
15	JUNE	8		5=2=85	6
16		----		6=2=85	8
17	2ND QTR	21		7=3=85	9
18				8=3=85	10
19	JULY	9		9=3=85	12
20	AUG	10		10=4=85	12
21	SEPT	12		11=4=85	14
22		----		12=4=85	16
23	3RD QTR	31		1=1=86	8
24				2=1=86	7
25	OCT	12		3=1=86	9
26	NOV	14		4=2=86	11
27	DEC	16		5=2=86	11
28		----		6=2=86	14
29	4TH QTR	42		7=3=86	11
30				8=3=86	11
31	=====			9=3=86	14
32	1985 TOT	109		10=4=86	17
33				11=4=86	19
34				12=4=86	24
35					
36	JAN	8			
37	FEB	7			
38	MAR	9			
39		----			
40	1ST QTR	24			
41					
42	APR	11			
43	MAY	11			
44	JUNE	14			
45		----			
46	2ND QTR	36			
47					
48	JULY	11			
49	AUG	11			
50	SEPT	14			
51		----			
52	3RD QTR	36			
53			:		
54	OCT	17			
55	NOV	19			
56	DEC	24			
57		----			
58	4TH QTR	60			
59					
60	=====				
61	1986 TOT	156			
62					
63					
64					
65					

Create a DIF file.

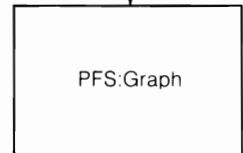
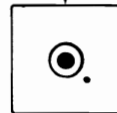


Figure 3-18

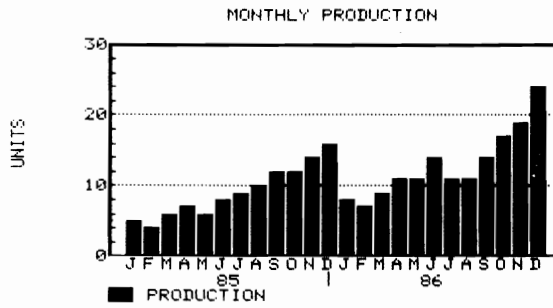


Figure 3-19

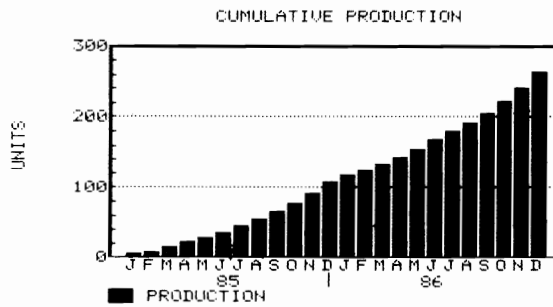


Figure 3-20

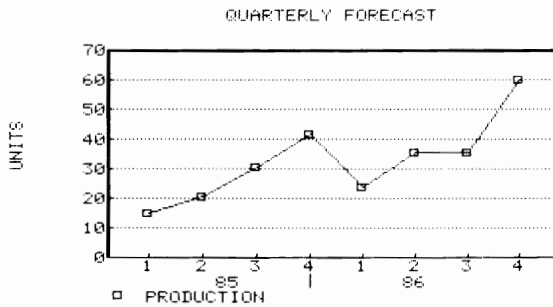


Figure 3-21

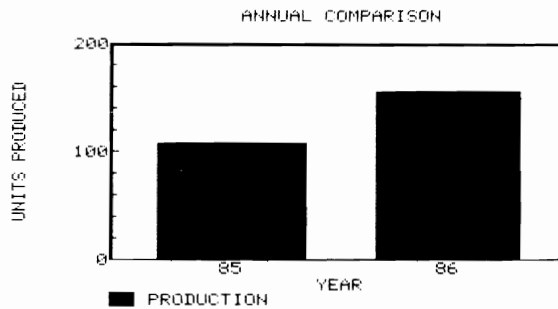


Figure 3-22

CASE STUDY 6

VisiCalc to PFS:Graph

Multiple DIF Files from the Same Data

In this section on PFS:Graph we'll look at an example which creates a DIF file containing more information than in the earlier case studies.

Suppose we have collected production information from a number of factories in different cities for the last six months, as shown in Figure 3-23, and that we want to manipulate this data and create a number of graphic comparisons.

A1									C1	
	A	B	C	D	E	F	G	H	31	
1										
2			MONTHLY PRODUCTION DATA							
3			COLLECTED: JUNE							
4										
5		FACTORY	JAN	FEB	MAR	APR	MAY	JUNE		
6		7ST. LOUIS	30	32	35	30	25	18		
7		8PITTSBURGH	80	80	82	81	78	80		
8		9ROCHESTER	48	57	60	70	82	93		
9		10POTTSVILLE	23	25	27	30	34	39		
10		11GREENSBURG	30	35	40	46	51	57		
11										
12										
13		TOTAL	211	229	244	257	270	287		
14		AVER	42	45	48	51	54	57		
15		HIGH	80	80	82	81	82	93		
16		LOW	23	25	27	30	25	18		
17										
18										
19										
20										

Figure 3-23

We could create a DIF file directly from Figure 3-23 at this point. In fact, we will create *multiple* DIF files.

Before we do so, let's enhance the VisiCalc worksheet to help ourselves in the process of exchanging data.

In Figure 3-24 we've added a number of rows to the top of the VisiCalc worksheet and one column at the left. On the top line we've added

>A5 RET /PP-H25 RET

which, when typed, will print the sheet. This will prove to be important documentation. It will help if we use this sheet repeatedly, but it cannot be automatically executed by VisiCalc. The line and the area printed from it are labeled A in the figure.

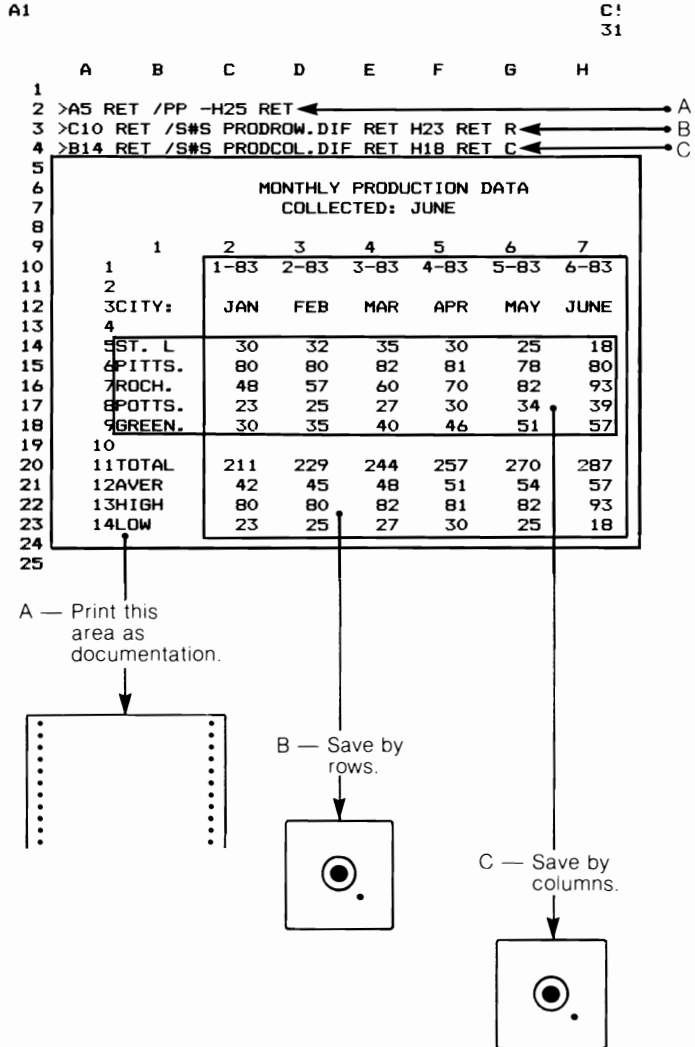


Figure 3-24

The next lines

```
>C10 RET /S#S PRODRROW.DIF RET H23 RET R
>B14 RET /S#S PRODCOL.DIF RET H18 RET C
```

create two DIF files, one by row (R), labeled B in Figure 3-24, and a second by column (C), labeled C in that figure. We'll see shortly why we created two files, not one.

Notice that we've added two rows.

```
1      2      3      4      ...
      1-83   2-83   3-83   ...
```

We'll soon explain the first of these. The second row, the dates, can be used later within PFS:Graph as we've illustrated in the last case study.

Finally notice a column of numbers

```
1
2
.
.
.
13
14
```

at the left.

This column, and the row of numbers (1, 2, ... 7) that we saw above, will be printed but they are not part of either of our DIF files. Once we enter PFS:Graph we'll need to supply the row (or column) numbers that correspond to the X and Y axis. (These are not the row numbers on the VisiCalc worksheet.)

Let's demonstrate by looking at Figure 3-25 where we'll plot St. Louis and Greensburg on the same graph with month as the X axis. We can see that row 3 will be the X axis, and that rows 5 and 9 will be the two sets of data for the Y axis. Now we can see the value of the sequential labels (1, 2, ... 13, 14) down the left of our printed documentation. They tell us the relative location of the data we want to use within our DIF file.

Similarly, Figure 3-26 illustrates this concept for the DIF file that we have saved by columns. The cities (in column 1) are the X axis and June (column 7) is the Y axis.

In these two figures, the printed documentation with extra rows and columns of labels that we produce as an intermediate step (in Figure 3-24) is helpful to us when we leave VisiCalc and enter the graphing program. Without it we will need to recall the relative positions of our data within the DIF file. If these extra rows and columns of labels interfere with the

DIF file in row order.



MONTHLY PRODUCTION DATA COLLECTED: JUNE							
	1	2	3	4	5	6	7
	1-83	2-83	3-83	4-83	5-83	6-83	6-83
3CITY:	JAN	FEB	MAR	APR	MAY	JUNE	
5ST. L	30	32	35	30	25	18	
6PITTS.	80	80	82	81	78	80	
7ROCH.	48	57	60	70	82	93	
8POTTS.	23	25	27	30	34	39	
9GREEN.	30	35	40	46	51	57	
11TOTAL	211	229	244	257	270	287	
12AVER	42	45	48	51	54	57	
13HIGH	80	80	82	81	82	93	
14LOW	23	25	27	30	25	18	

X Axis

Y Axis

Y Axis

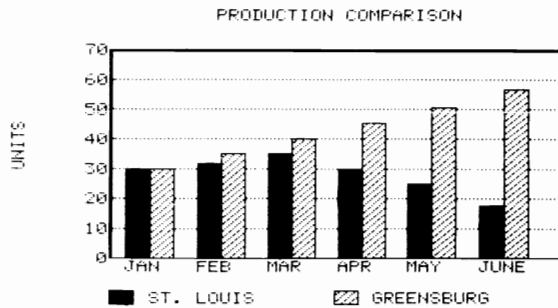
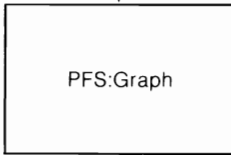
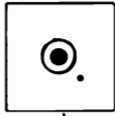


Figure 3-25

DIF file in column order.



X Axis

Y Axis

MONTHLY PRODUCTION DATA COLLECTED: JUNE						
1	2	3	4	5	6	7
	1-83	2-83	3-83	4-83	5-83	6-83
3CITY:	JAN	FEB	MAR	APR	MAY	JUNE
4						
5ST. L	30	32	35	30	25	18
6PITTS.	80	80	82	81	78	80
7ROCH.	48	57	60	70	82	93
8POTTS.	23	25	27	30	34	39
9GREEN.	30	35	40	46	51	57
10						
11TOTAL	211	229	244	257	270	287
12AVER	42	45	48	51	54	57
13HIGH	80	80	82	81	82	93
14LOW	23	25	27	30	25	18

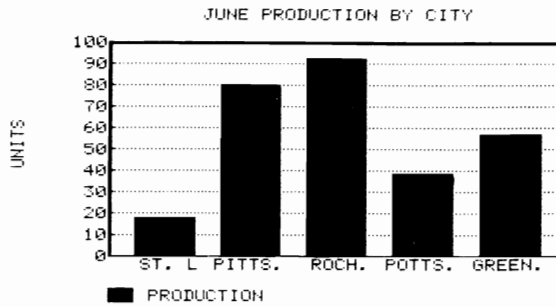
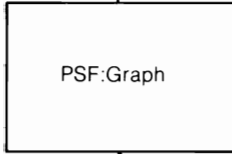


Figure 3-26

desired appearance of the VisiCalc sheet, we can establish a separate area on the sheet as described in earlier case studies.

It is important to realize that the need for two DIF files here was created by PFS:Graph. That program does not ask if our data was saved by row or column. Instead it asks for *relative* numbers of the strips of data we want for the X and Y axes. DIF does store strips, but it does so without any internal description of whether the data is in row or column order. A DIF file is really both row and column at the same time and we can access the same DIF file either way. We've seen this in Chapter 2, Using the DIF Format in a VisiCalc Context, where we learned to reload the same DIF file by row or by column. The file has neither row nor column orientation. That is something we impose on the data.

Within PFS:Graph the responsibility is ours for providing an orientation. We must remember how we've saved the file and then provide the location of the relative strips of data we want for each axis.

CASE STUDY 7 **VisiCalc to PFS:Graph** **Using Several DIF Files to Represent the Same Data**

In the final example of this section we'll take advantage again of the PFS:Graph ability to selectively choose rows (or columns) from a DIF file for graphing.

Suppose that we create a DIF file by rows from the indicated area of the large budget sheet of Figure 3-27. Now within PFS:Graph we can select the rows we desire for graphing. We do not select blank rows (such as row 2), rows with underlines (such as row 8), or rows with labels that we do not want to graph. All these rows are in the DIF file, but are ignored in the graphing. Instead, we could select rows 1 and 5 (to plot the change in Professional Salaries over our fiscal years), or rows 1, 28, and 31 (to compare several Benefits categories over time).

Thus the row/column selection feature of PFS:Graph is a significant way to simplify the problems associated with removing unwanted data; however, we must be able to identify the rows (or columns) we want without being able to see them within PFS:Graph.

Ten Year Fiscal Forecast Based On a
10 % Increase per Year

Code	Description	Fiscal Year 0 (000)	Fiscal Year+1 (000)	Fiscal Year+2 (000)	Fiscal Year+3 (000)	Fiscal Year+4 (000)	Fiscal Year+5 (000)	Fiscal Year+6 (000)	Fiscal Year+7 (000)	Fiscal Year+8 (000)	Fiscal Year+9 (000)
1110	Prof. Salr.	1853	1708	1878	2045	2271	2498	2747	3021	3323	3655
1120	P/T Prof.	28	30	33	36	39	42	46	50	55	60
1141	Consultant	99	108	118	129	141	155	170	187	205	225
	Sub.	1480	1846	2029	2230	2451	2695	2963	3258	3583	3940
142	P/T I	116	127	139	152	167	183	201	221	243	267
143	P/T II	54	59	64	70	77	84	92	101	111	122
	Sub.	170	186	203	222	244	267	293	322	354	389
144	P/T III	107	117	128	140	154	169	185	203	223	245
145	P/T IV	18	19	20	22	24	26	28	30	33	36
	Sub.	125	136	148	162	178	195	213	233	256	281
150	Hourly I	62	68	74	81	89	97	106	116	127	139
152	Hourly II	15	16	17	18	19	20	22	24	26	28
155	Hourly III	0	0	0	0	0	0	0	0	0	0
	Sub.	77	84	91	99	108	117	128	140	153	167
4200	Benefits										
210	Soc Sec	209	229	251	276	303	333	366	402	442	486
220	Retirat	112	123	135	148	162	178	195	214	235	258
232	L/T Dya	12	13	14	15	16	17	18	19	20	22
240	Wth In	51	56	61	67	73	80	88	96	105	115
260	Maj Med	15	16	17	18	19	20	22	24	26	28
270	Mka Com	27	29	31	34	37	40	44	48	52	57
	Sub.	426	466	509	558	610	668	733	803	880	966
310	Supplies I	44	48	52	57	62	68	74	81	89	97
320	Supplies II	62	68	74	81	89	97	106	116	127	139
	Sub.	106	116	126	138	151	165	180	197	216	236
408	Consultants	72	79	86	94	103	113	124	136	149	163
590	Cons. Trvl	12	13	14	15	16	17	18	19	20	22
	Sub.	84	92	100	109	119	130	142	155	169	185
430	Telephone	25	27	29	31	34	37	40	44	48	52
	Sub.	25	27	29	31	34	37	40	44	48	52
470	Travel I	48	52	57	62	68	74	81	89	97	106
515	Travel II	27	29	31	34	37	40	44	48	52	57
	Sub.	75	81	88	96	105	114	125	137	149	163
520	Hospitality	10	11	12	13	14	15	16	17	18	19
544	Recruiting	36	39	42	46	50	55	60	66	72	79
580	Advsrs I	0	0	0	0	0	0	0	0	0	0
585	Advsrs II	0	0	0	0	0	0	0	0	0	0
	Sub.	46	50	54	59	64	70	76	83	90	98
620	Equip. Maint	57	62	68	74	81	89	97	106	116	127
	Sub.	57	62	68	74	81	89	97	106	116	127
630	Repair	38	41	45	49	53	58	63	69	75	82
631	Maintenance	42	46	50	55	60	66	72	79	86	94
	Sub.	80	87	95	104	113	124	135	148	161	176
6755	Contingency	10	11	12	13	14	15	16	17	18	19
756	Conting756	10	11	12	13	14	15	16	17	18	19
	Sub.	20	22	24	26	28	30	32	34	36	38
88912	Equipment	120	132	145	159	174	191	210	231	254	279
88914	Equip700	104	114	125	137	150	165	181	199	218	239
	Sub.	224	246	270	296	324	356	391	430	472	518
=====											
TOTAL \$		3195	3501	3834	4204	4610	5057	5548	6090	6683	7336
=====											

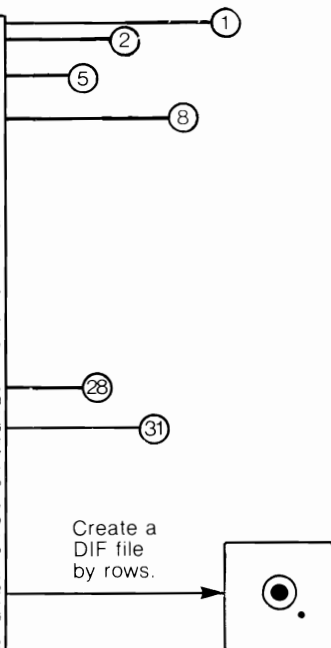


Figure 3-27

Additional Factors in the VisiCalc to PFS:Graph Interchange of Data

Below are a number of additional considerations in the interchange of data between these two products using DIF.

1. PFS:Graph does not create DIF files, and thus data we enter directly to PFS:Graph is not available to be sent to VisiCalc or other programs via the DIF format.
2. PFS:Graph contains useful data reduction capabilities that we can take advantage of if we rearrange our data (illustrated in Case Study 5) as required before creating the DIF file.
3. VisiCalc has a larger maximum and smaller minimum numeric value capability than PFS:Graph. Therefore we could have a valid value in a DIF file that is invalid within PFS:Graph.
4. PFS:Graph has a limit of 36 data points per graph after reduction, and a DIF file that does not reduce to 36 or fewer data points will cause an error. (The DIF file may be large, as in Figure 3-27, but when we plot one row against another from that budget data, we have only 10 points, well within the limits.)
5. PFS:Graph contains merge capabilities which include the ability to combine multiple DIF files into one set of data.

DATA INTERCHANGE BETWEEN DB MASTER AND THE EXECUTIVE SECRETARY

In this case study we'll move selected fields from the records of a data base to a word processing program where we'll generate form letters based on the data transmitted. The interchange will use a DIF file created from a DB Master data-base file and read by The Executive Secretary word processor.

The examples in this section were completed using an Apple II Plus microcomputer with 64K of memory, a SUP'R'TERMINAL 80-character board from M & R Enterprises, with printed output sent to an EPSON MX-100 printer via a Grappler Interface card. The data base used is DB Master, Version Three, revision #3.02 with DB Master Utility Pak #1, revision #1.07. The word processor is The Executive Secretary, release IV. (Trademark information is contained in the front matter of this book and product information is in Appendix C.)

CASE STUDY 8 DB Master to The Executive Secretary

We'll examine the process shown in Figure 3-28, that of linking the contents of the data base with the word processor. We see a listing of the data from a file that we've prepared with DB Master, our data-base program. We also see the letter that we want to send to each individual for whom we have a record in the data base. The letter will be the same for each individual except for the name and address, greeting, and manager's name for the carbon copy.

We'll assume that we have created our file format and entered all of our data into DB Master. Each of our records contains the following fields:

- Last name
- First name
- Street
- City
- State
- Zip
- Social Security Number
- Title
- Manager

For our letter we'll use all of the fields except the Social Security Number and Title.

With DB Master we are able to create and read transfer files by using a separately purchased diskette of modules called DB Master Utility

DB MASTER
records and fields in
our data base.

```

• LAST NAME - BEIL
• FIRST NAME - MARIAN
• STREET - 2198 MAHANTONGO
• CITY - ROCHESTER
• STATE - NY
• ZIP - 14618
• SSN - 123-45-6789
• TITLE - ARTIST
• MANAGER - STEVE JONES

• LAST NAME - BILL
• FIRST NAME - BILLY
• STREET - BROOKLAWN DR #9
• CITY - ROCHESTER
• STATE - NY
• ZIP - 14618
• SSN - 223-34-5566
• TITLE - AUTHOR
• MANAGER - DON BEIL

• LAST NAME - KENNEDY
• FIRST NAME - GABE
• STREET - 330 WILSON ST
• CITY - BRIGHTON
• STATE - NY
• ZIP - 14623
• SSN - 556-44-0033
• TITLE - ACTOR
• MANAGER - DON BEIL

• LAST NAME - PATRICK
• FIRST NAME - NOAH
• STREET - 330 WILSON ST
• CITY - BRIGHTON
• STATE - NY
• ZIP - 14618
• SSN - 444-55-6655
• TITLE - PHOTOGRAPHER
• MANAGER - NANCY JONES

• LAST NAME - SHEMELEY
• FIRST NAME - TOMMY
• STREET - FIREHOUSE #3
• CITY - WESTMONT
• STATE - NY
• ZIP - 14623
• SSN - 109-87-3311
• TITLE - ARTIST
• MANAGER - MARIAN BEIL

```

THE EXECUTIVE SECRETARY
letters printed for those
in the data base.

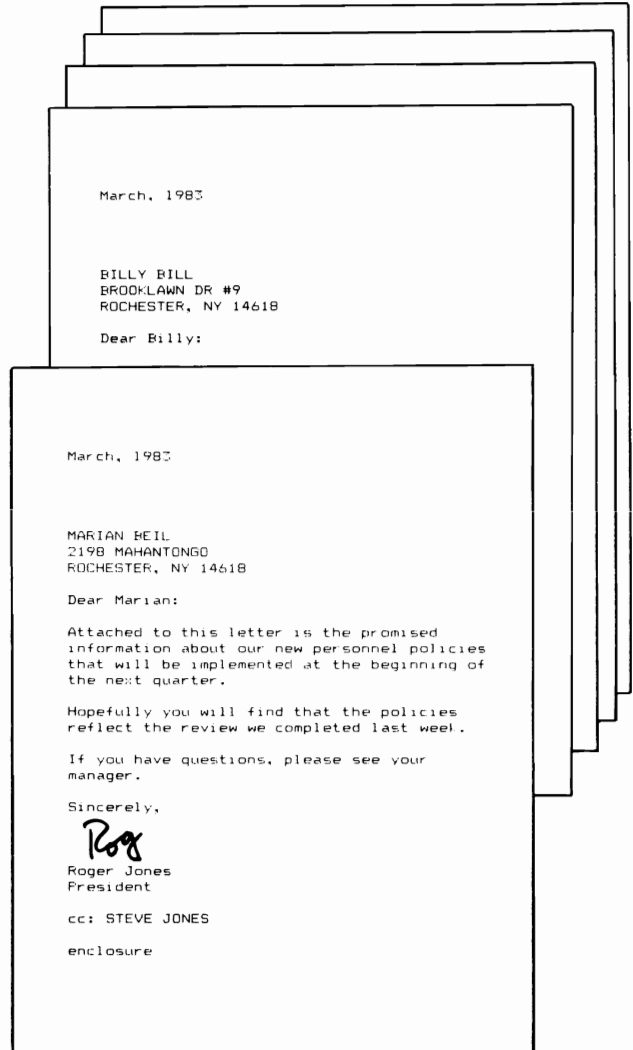


Figure 3-28

Pak #1. The idea of a separate utility illustrates a different approach to DIF files from what we have seen in the earlier case studies. The translation to and from DIF files is one of many different functions contained in this set of utilities.

We'll assume that we have successfully run Utility Pak #1 and that we have selected the appropriate choice, TRANSLATE FILE, from its menu.

From this point we are asked approximately a dozen questions designed to lead us through the process of DIF file creation. There are many decisions to be made before the file can be created. We'll go through the steps in general terms to understand some of the considerations in creating a DIF file from a data-base file.

First we're asked if we want to translate *to* a text file (create a DIF file from a DB Master file) or translate *from* a text file (pass data to an existing DB Master file from a DIF file). We'll choose the first.

We then have the option with one response of including all the fields in the new DIF file, with each field in the same order as in the original DB Master file. We will not do this, since for our example we will ignore a number of the fields and rearrange others. This selection capability is important because we can save time later by eliminating unneeded fields now.

Our next task is to establish the format for the data in the DIF file; that is, which fields should be translated for the DIF file, and in what order. We're presented with the names of all fields and asked to enter them in the order we want for our DIF file. The list at the right below shows how we'll set up these records. We've reversed the last and first name, and eliminated SSN (Social Security Number) and TITLE.

DB Master Record	DIF Record
1 LAST NAME	1 FIRST NAME
2 FIRST NAME	2 LAST NAME
3 STREET	3 STREET
4 CITY	4 CITY
5 STATE	5 STATE
6 ZIP	6 ZIP
7 SSN	7 MANAGER
8 TITLE	
9 MANAGER	

The format we've created can then be saved on a diskette. We'll do so under the name BENEFITS LETTER. Later (the same day, next week, next month,...) if we return to this step with this file, we'll have this format already available to us without going through the steps of creating it again.

Now we'll print this format, as shown in Figure 3-29. This will become important later in this process as documentation of the data layout in our DIF file.

```
.....  
•  
• TRANSLATOR FORMAT => BENEFITS LETTER  
•  
• FIELD #1 - FIRST NAME  
• FIELD #2 - LAST NAME  
• FIELD #3 - STREET  
• FIELD #4 - CITY  
• FIELD #5 - STATE  
• FIELD #6 - ZIP  
• FIELD #7 - MANAGER  
•  
•  
•
```

Figure 3-29

As with other DB Master actions, we can select the records we want to translate. We'll simply choose ALL RECORDS instead of developing a selection format. We'll then select MOVE ON TO THE NEXT STEP rather than printing, replacing, or choosing another format.

We then decide if we want to translate the file now or not. We'll do it now, inserting our EMPLOYEES Master file in one disk drive, and a diskette that we have *previously initialized* with The Executive Secretary. We'll enter a filename, here

EMP

that we will use later in The Executive Secretary. Utility Pak #1 automatically appends ".DIF" to the end of the filename.

We then have options of watching the file on the screen as it is created, limiting the number of records, and initializing the diskette. We'll choose 'no' for all of these options, and then the translation process runs until completion.

After leaving Utility Pak #1, we'll run The Executive Secretary and prepare our letter.

A sample of the letter, named EMPLETTER, is shown in Figure 3-30. The listing in that figure, printed from The Executive Secretary, includes line numbers at the left (at our request), and we notice that lines 1, 2, 9 through 14, and 31 all contain unusual markings. We will explain them in order.

```

.
. FILE: EMPLETTER
.
. 1 >DB EMP
. 2 >RN 1-
. 3
. 4 March, 1983
. 5
. 6
. 7
. 8
. 9 %01 %02J
.10 %03J
.11 %04, %05 %06J
.12 >KD 20, Enter first name then RETURN.
.13
.14 Dear %20:J
.15
.16 Attached to this letter is the promised information about our new personnel
.17 policies that will be implemented at the beginning of the next quarter.
.18
.19 Hopefully you will find that the policies reflect the review we
.20 completed last week.
.21
.22 If you have questions, please see your manager.
.23
.24 Sincerely,
.25
.26
.27
.28 Roger JonesJ
.29 President
.30
.31 cc: %07J
.32
.33 enclosure
.
.
```

Figure 3-30

Line Number	Explanation
1	<p>This line contains a command, >DB, indicating that we want to integrate a DB Master file in our letter, followed by the name of the file, EMP. This action connects our letter file with the DIF file we created earlier.</p> <p>The Executive Secretary can incorporate files like this from a variety of software packages.</p>
2	<p>This command, >RN, gives us control of which records are included by record number. Here we've typed</p> <p style="text-align: center;">>RN 1-</p> <p>indicating to The Executive Secretary that we want to start with record number 1 and proceed through the entire DIF file.</p>
9, 10, 11, and 31	<p>On these lines we've included items such as &01, &02, etc. These are signals to incorporate fields from the DIF file at these points in the letter. These characters indicate the relative location of the data field we want within the records of the DIF file.</p> <p>We know this relative order because we designed the record format when we created the file within Utility Pak #1. The first field (&01) was the FIRST NAME. The second field (&02) was LAST NAME, etc.</p> <p>Now we can see the importance of documenting the DIF file record format as we did when we printed the format in Figure 3-29.</p>

12 and 14

Notice in the letter of Figure 3-28 that the first name appears in full upper case (MARIAN) at one place in the letter (line 9) but that in the greeting it appears in upper and lower case (Marian). DB Master works with upper case characters, a hardware limitation of the Apple II Plus. Our DIF file therefore contains only upper case characters, although this is not a limitation of the DIF format.

We are using an 80-character board, a hardware addition to the Apple that includes the capability of permitting upper- and lower-case letters to be generated on our monitor. The Executive Secretary uses this feature although DB Master does not.

In our letter we've included the command of line 12 which stops the printer at this point, prints the message that we designed ("Enter first name then RETURN"), and accepts what we type, labeling it "20", a number we selected. Our typed response is then incorporated into the letter as desired, here on line 14. When we see the message on the screen we'll check on the printer for line 9, and then type the first name using the desired upper- and lower-case characters.

The Executive Secretary does include a command that is designed for situations like this. It allows us to convert words to lower case beginning with a character we select. For example

>LC 2

will convert all characters to lower case starting from the second character.

We've now successfully integrated data from our DB Master DIF file into a document generated by The Executive Secretary.

Figure 3-31 summarizes this complete process.

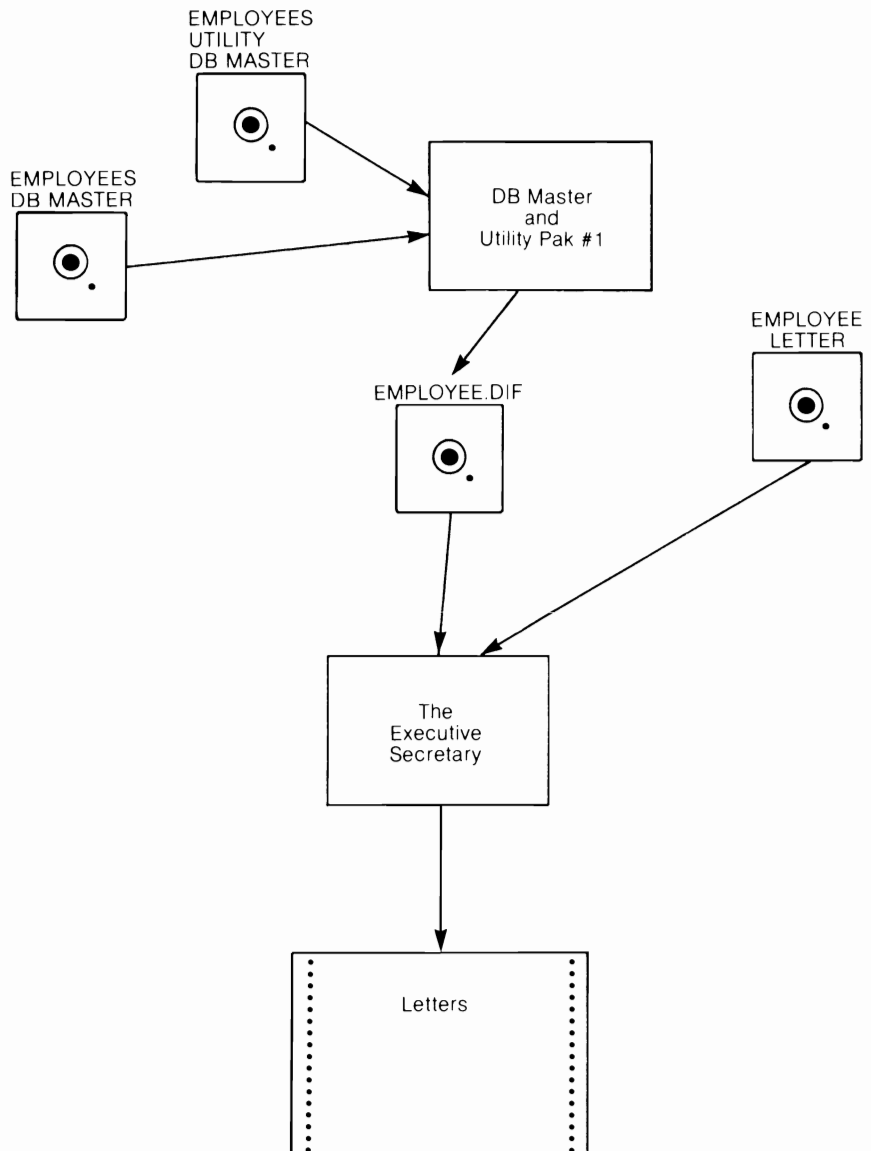


Figure 3-31

Additional Factors in the Use of DIF Files by DB Master and The Executive Secretary

There are a number of additional considerations when using products like these.

1. This exchange involves many diskettes. Those required include:
 - DB Master.
 - DB Master Utility Pak #1.
 - Employee master file (our DB Master data diskette).
 - Employee utility diskette (a requirement of DB Master).
 - Employee DIF file diskette (our data generated for the exchange by DB Master).
 - The Executive Secretary.
 - Employee letter (our letter developed within The Executive Secretary).

This means we'll be swapping diskettes to complete the transfer. If we're unsuccessful the first time (we generate the file incorrectly, forget the order of the fields in the DIF file, include incorrect commands in our letter, etc.), we'll increase the diskette swapping. As we've seen in other examples we need to work with care.

2. When attempting to complete this transaction, an error message was received with The Executive Secretary that was not explained in its documentation.

3. DB Master can read DIF files with Utility Pak #1 as well as create them; however, the DIF file must include field identification. If it does not, DB Master does *not* prompt for it, as VisiTrend/Plot does. This does not meet the requirement of the *DIF Technical Specification* which says:

If a reading program requires the information provided by an optional item, it should prompt the user to supply the missing information and not require the item itself.

Stoneware, the creator of DB Master, offers a BASIC program that performs this function. (Refer to the "DIF Translator—VisiCalc to DB Master" article in the Bibliography on the DIF Format.)

4. Since DIF files are text files, they can be read and modified by many word processors. A full knowledge of the DIF technical specifications allows a user to create or modify DIF files with a word processor. Chapter 6, A Tutorial on the DIF Format, and Appendix A, *DIF Technical Specification*, contain this information.

5. Both The Executive Secretary and DB Master have selection capabilities which allow users significant control over the contents of letters generated and of the content of the DIF file, respectively.

DATA INTERCHANGE FROM VISIWORD TO VISICALC USING LOADCALC

In this case study we'll transfer data from a word processing program to a spreadsheet program via a DIF file created by a utility program. The concepts apply to a wide variety of word processing and other applications.

This case study uses a prerelease copy of VisiWord, LoadCalc, and VisiCalc. The hardware used is an IBM Personal computer. (Refer to the front matter for trademark information and to Appendix C for product information.)

CASE STUDY 9 VisiWord to LoadCalc to VisiCalc

A memo was created with VisiWord and then printed, as shown in Figure 3-32. For our case study we'll create a DIF file from the data in the areas marked, without reentering that data.

VisiWord allows us to save data in several formats, but not in the DIF format. The DIF format is not an appropriate one for storage of the body of a word processing text, because the format was designed principally for rows and columns of numeric data; however, as in the memo of Figure 3-32 we may have a table of numeric values as part of a document or even as a complete document that we create with our word processor. In these cases, the ability to extract values and create a DIF file can provide us with the benefits we've seen in the other case studies.

For our case study, the document was saved as a print file from VisiWord by using the print-to-disk option provided by that word processor. This file, here named CUSTDISC.PRN (for customer discount file in print format), has the same format as the printed file, meaning that carriage returns and spacing within the file on the disk are the same as in the document we printed.

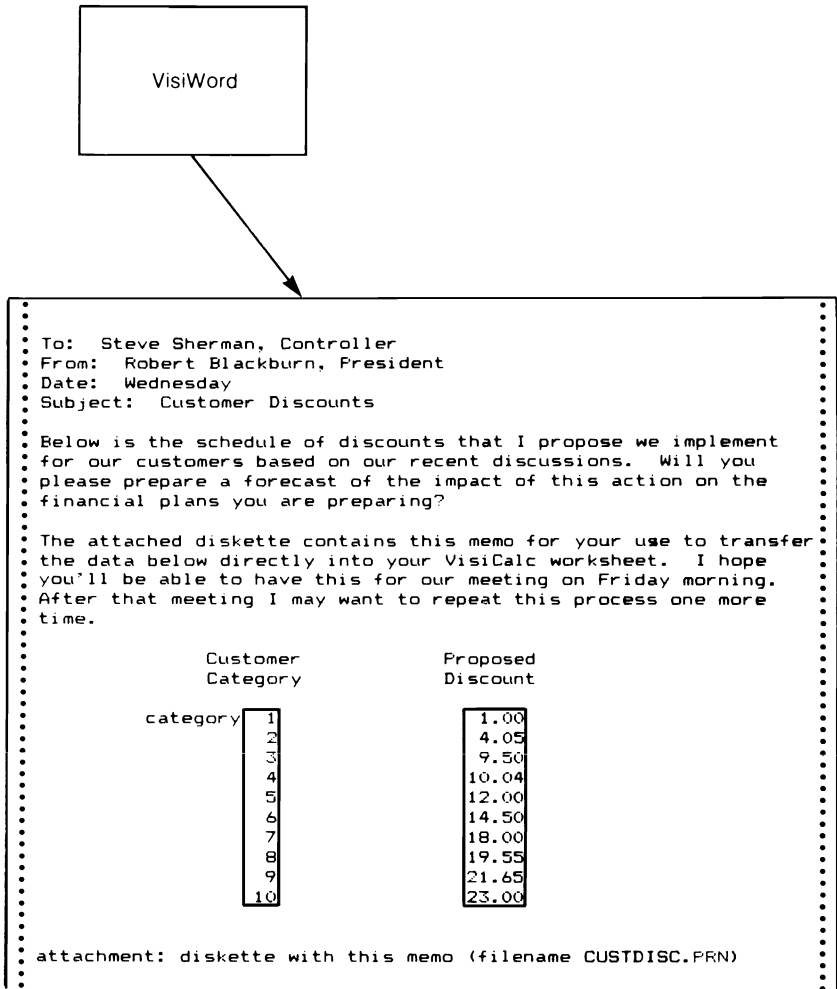


Figure 3-32

This process is summarized in Figure 3-33. To save the print file from VisiWord we'll follow the steps below, moving from menu to menu and responding to prompts as indicated.

- Step 1** Enter the VisiWord menu by pressing the E^{SC} key.
- Step 2** Enter the PRINT menu with a single P keystroke or by moving the cursor to PRINT and then pressing the RETURN key (the ENTER key on the IBM Personal Computer).
- Step 3** Indicate DISK (by pressing D or moving the cursor).
- Step 4** Supply a filename, either selecting from a list or typing the name. We'll type in CUSTDISC.PRN, with the suffix indicating a print file.

At this point we've saved a print file image of the memo.

We'll leave VisiWord by selecting EXIT from the main menu, and then START-NEW-PROGRAM from the next menu. As directed by VisiWord, we'll insert the new product, here LoadCalc, and press RETURN.

The full process of this case study is shown in Figure 3-34, and we see that LoadCalc is the middle step between VisiWord and VisiCalc. That figure shows three separate data diskettes, although all data files could be stored on the same diskette.

LoadCalc is a utility program that "converts text or prints files into DIF files.," an important utility function for those using programs with DIF format capabilities.

LoadCalc uses a command structure that parallels VisiCalc, for example:

Command	Means
/L	Load a file
/S	Save a file
etc.	

It also uses arrow keys in the same way that VisiCalc does to move the cursor on the screen and to scroll the diskette directory.

We'll run LoadCalc, leaving the word processor data diskette on which we've saved the CUSTDISC.PRN file in the second (B) disk drive. We load that full file into LoadCalc, using the /L command. Figure 3-35 shows the LoadCalc screen after we've scrolled over the document to place the text we want on the screen.

VisiWord Screen

.....!.C.....5.....6.....
To: Steve Sherman, Controller
From: Robert Blackburn, President
Date: Wednesday
Subject: Customer Discounts

Below is the schedule of discounts that I propose we implement for our customers based on our recent discussions. Will you please prepare a forecast of the impact of this action on the financial plans you are preparing?

The attached diskette contains this memo for your use to transfer the data below directly into your VisiCalc worksheet. I hope you'll be able to have this for our meeting on Friday morning. After that meeting I may want to repeat this process one more time.

Customer Category	Proposed Discount
category 1	1.00
2	4.05
3	9.50

VW: Press Esc to enter menu
Delete Copy Move Find Layout Windows Storage Print Options ?=help Exit

INSERT Line 23

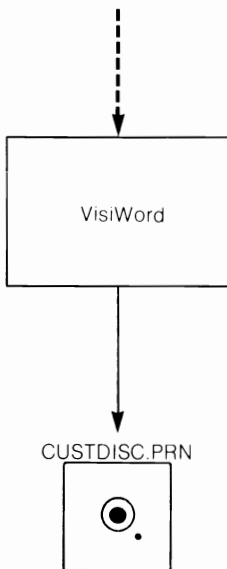


Figure 3-33

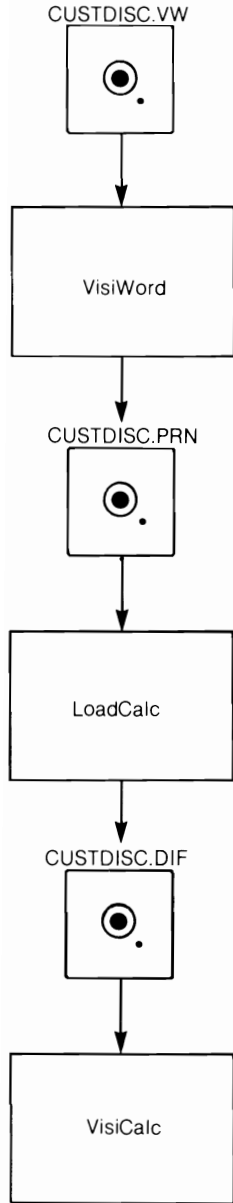


Figure 3-34

To: Steve Sherman, Controller
From: Robert Blackburn, President
Date: Wednesday
Subject: Customer Discounts

Below is the schedule of discounts that I propose we implement for our customers based on our recent discussions. Will you please prepare a forecast of the impact of this action on the financial plans you are preparing?

The attached diskette contains this memo for your use to transfer the data below directly into your VisiCalc worksheet. I hope you'll be able to have this for our meeting on Friday morning. After that meeting I may want to repeat this process one more time.

Customer Category	Proposed Discount
category 1	1.00
2	4.05
3	9.50
4	10.04
5	12.00
6	14.50
7	18.00
8	19.55
9	21.65
10	23.00

attachment: diskette with this memo (filename CUSTDISC.PRN)

please prepare a forecast of the impact of this action on the financial plans you are preparing?

The attached diskette contains this memo for your use to transfer the data below directly into your VisiCalc worksheet. I hope you'll be able to have this for our meeting on Friday morning. After that meeting I may want to repeat this process one more time.

Customer Category	Proposed Discount
category 1	1.00
2	4.05
3	9.50
4	10.04
5	12.00
6	14.50
7	18.00
8	19.55
9	21.65
10	23.00

attachment: diskette with this memo (filename CUSTDISC.PRN)
End of Data

Figure 3-35

Step 6
SAVE in DIF format, select first line
<A>

Step 8
< B >

please prepare a forecast of the impact of this action on the financial plans you are preparing?
The attached diskette contains this memo for your use to transfer the data below directly into your VisiCalc worksheet. I hope you'll be able to have this for our meeting on Friday morning. After that meeting I may want to repeat this process one more time.

Customer Category	Proposed Discount
category 1	1.00
2	4.05
3	9.50
4	10.04
5	12.00
6	14.50
7	18.00
8	19.55
9	21.65
10	23.00

Step 8
Step 10
Step 11

attachment: diskette with this memo (filename CUSTDISC.PRN)
End of Data

Figure 3-36

Our next step in LoadCalc is to mark the areas we want sent to the DIF file. We do so by identifying columns and then selecting rows (really lines here). Together, these delineate the area from which we'll create the file. This has some conceptual similarities to creating a DIF file from a VisiCalc spreadsheet although the details differ. Instead of an upper left corner and a lower right corner, we'll specify columns, and then indicate the top and bottom rows of the area to be saved.

The exact steps to follow are summarized below, with some of the step numbers shown on the screen of Figure 3-36.

1. Run LoadCalc.
2. Load the memo document (the print-to-disk text file), by typing
/L B:CUSTDISC.PRN RETURN
3. Scroll the screen until the area we want is in view, as shown in Figure 3-36.
4. Change the column width to three. Unlike VisiCalc, this leaves the screen completely intact, and only changes the width of the cursor.

/C 3 RETURN

5. Move the shortened cursor to a location we want to define as a column. In LoadCalc the cursor moves horizontally one character at a time (instead of the full-width cursor move of VisiCalc). We'll move the cursor to any line in our first column of data, with the cursor positioned so that the value is right justified in the cursor.

6. When the cursor is positioned, we'll mark this column by pressing the RETURN key. LoadCalc places an indication above the column selected, the {A} we see in the shaded area of Figure 3-36.

RETURN

7. Change the column width to five. Again only the cursor width changes.

/C 5 RETURN

8. Move the cursor to the second column (as shown in Figure 3-36) and mark that column. Notice that the {B} mark at the top of this column is five characters wide, and that the {A} mark is three columns wide. Each indicates the column width selected.

9. Begin the command to save the DIF file.

/S RETURN

10. Respond to a prompt to move the cursor to the first line we want saved. The cursor should be anywhere on the line labeled "Step 10" in Figure 3-36.

RETURN

11. Respond to a second prompt to move the cursor to the last line of the area we want saved, the line labeled "Step 11" in Figure 3-36.

RETURN

12. Respond to a prompt to save by row, column, or RETURN. We'll choose RETURN, which defaults to row order.

RETURN

13. We provide a filename, scrolling the directory if desired. We'll use CUSTDISC.DIF for this DIF file of discount values. If a file exists with this name we can overwrite it if desired or provide another name, as in VisiCalc.

CUSTDISC.DIF RETURN

This completes the job, and at this point LoadCalc creates a DIF file from the areas defined by the actions above. We quit LoadCalc by typing

/Q

We can now use this DIF file as input to a program that reads DIF files.

For example, we'll read the file onto an area of a VisiCalc sheet where it is used as a table with the @LOOKUP function. In Figure 3-37, we see a VisiCalc screen at the top before entering the DIF file. We've placed instructions on the sheet to make it easier to load the DIF file. When we follow the directions we get the screen at the bottom of Figure 3-37, with our data in place.

We have now successfully completed the interchange.

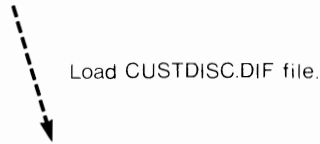
M27

C
34

20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

L M N O P Q R S

>M28
/S#L B:CUSTDISC ENTER ENTER
Rob's discount proposal.



Load CUSTDISC.DIF file.

M28 (V) 1

C
33

20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

L M N O P Q R S

>M28
/S#L B:CUSTDISC ENTER ENTER
Rob's discount proposal.

1	1
2	4.05
3	9.5
4	10.04
5	12
6	14.5
7	18
8	19.55
9	21.65
10	23

Figure 3-37

Additional Capabilities and Considerations in Using LoadCalc or VisiWord

Included among the other capabilities of LoadCalc are the abilities to:

- Use text files created by communications programs from data available on networks, mainframe computers, or other outside sources.
- Save fields as numbers or labels as appropriate.
- Mark multiple columns and mark the same column more than once. The order in which columns are marked determines their order in the DIF file.
- Reset marked columns and begin the process again to create additional DIF files from the same or new data.
 - Provide limited assistance with a help command (/H).
 - Enter labels for columns if desired.
 - Convert fractions such as 12 1/4 to decimal equivalents, here 12.25. This makes the product useful with stock market prices or similar data taken from the financial portion of a network.
 - Keep (the /K command) and get (the /G command) the definitions of column positions, types, and labels, so that once entered they don't need to be reentered when the same retrieval occurs.

Since LoadCalc does not allow us to select rows of data in the same way we select columns, we may need to create multiple DIF files, repeating the steps above for each. For example, in a lengthy file, if we mark our columns and then want lines 8-12, 15, 34, 40-55, and 67, we would need to create 5 separate DIF files. We cannot create this as a single file in one step.

When using VisiWord in a file transfer context, we should know that VisiWord *can* accept files from VisiCalc if those files are created using the Print command (/PF) of VisiCalc (rather than the DIF file feature of the Storage command). Additionally, DIF files can be read, displayed, and manipulated as is by VisiWord, that is, VisiWord cannot gracefully extract the data values from a DIF file, but can perform some processing on the file. We'll discuss this again in Chapter 6, A Tutorial on the DIF Format.

DATA INTERCHANGE FROM COMPUSERVE TO VISICALC USING MAINLINE

In this case study we'll use MAINLINE to retrieve and create a DIF file from selected data on a financial data base on the CompuServe network. We'll then read the DIF file into a VisiCalc spreadsheet.

The examples in this section were completed using an Apple II Plus microcomputer with 64K of memory and two disk drives, and a Hayes Micromodem II. The software used was a prerelease copy of MAINLINE from the Gregg Corporation, and VisiCalc, with securities data retrieved from the VALUE data base on the CompuServe network. (Trademark information is contained in the front matter of this book and product information is contained in Appendix C.)

CASE STUDY 10 CompuServe to MAINLINE to VisiCalc

MAINLINE is a communications and data retrieval system composed of software on both the user's microcomputer and on the mainframe from which data is retrieved. From the user's perspective there are two principal tasks, both on the microcomputer. The user must:

- Create a login procedure.
- Create data retrieval requests.

The login procedures allow automatic login to a distant system, with separate procedures for each system. (A typical login procedure is shown in Figure 3-38.) A user of a single remote system will need only one login template, and once created may not need to revise that template unless there are changes in the system prompts, telephone number, etc., or unless the user wants to change a password.

For that reason we'll discuss that procedure only briefly with several observations.

- The procedure includes different sections such as the required SETTINGS, LOGIN, and LOGOUT sections shown here, each containing commands.
- In the LOGIN section of Figure 3-38 we see commands to dial a telephone number, send control characters, wait for a prompt (ID:) from

```
1.SETTINGS SECTION
2.
3.DUPLEX FULL
4.DEBUG 0
5.
6.LOGIN SECTION
7.
8.DIAL 5551212
9.SEND ""^C"
10.MATCH "ID:"
11.SEND "HI MON<CR>"
12.MATCH "PASSWORD:"
13.NOECHO "SONNY<CR>"
14.
15.LOGOUT SECTION
16.
17.SEND "BYE<CR>"
```

Figure 3-38

the system, send identification information, wait for another prompt (PASSWORD:), and send a password without displaying it (NOECHO).

- Additional commands (not shown in Figure 3-38) allow us to identify the specific hardware communication device and line speed, and include time delays of several types. For example on a telephone system that requires users to dial 9 to get an outside line before dialing the rest of the number, we can build in a several second delay in the DIAL command so that MAINLINE automatically waits after dialing the 9 to allow an outside line to be reached. Other commands are also available.

- MAINLINE provides a File Utility module that includes an editor for creating and revising user files such as this login procedure.

The login template may be complicated for a neophyte user to create; however, once working correctly it is automatically executed by the software. This complication is due primarily to the nature of network login procedures.

For our case study we've created a login file similar to the one shown in Figure 3-38. That procedure contains a hypothetical telephone number, identification code, and password used to access the CompuServe network.

CompuServe provides a broad variety of services, including access to financial databases such as VALUE, the one used in this case study. VALUE is "an interactive securities information system, (that) provides fast access to a wide variety of data on stocks, bonds and options..."

Along with other applications, MAINLINE can be used as retrieval software with the VALUE database on CompuServe, and delivers data to the microcomputer in the DIF format. The Gregg Corporation has installed its "cooperating" software on the CompuServe mainframe, which works with MAINLINE on the microcomputer. The mainframe software is invisible to the microcomputer user once the login procedure is successfully completed.

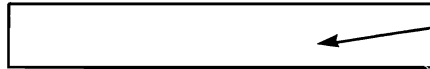
On the microcomputer we boot MAINLINE (a system split into two diskettes plus a third data diskette) and choose from the menu items below.

Menu Item	Explanation
1. Request Builder	<p>Create a request to retrieve information from the mainframe, for example</p> <ul style="list-style-type: none"> • provide today's high, low, and closing price for a series of stocks. <p>This example illustrates the three basic dimensions of the request we'll build for this case study.</p> <ul style="list-style-type: none"> • <i>Time</i>. Eight years of pricing history and 13 years of dividend history are available. • <i>Name</i> of securities. Over 42,000 stocks, bonds, and options are available. • <i>Item</i> data about each security. Over 60 fields are in the data dictionary for each security, where one item is closing price, another is high price, etc. <p>We'll discuss this section in more detail below.</p>
2. Data Retrieval	<p>This option automatically executes our login file, communicates our request for data to the mainframe, returns data to us in the DIF format, and logs off at our request.</p>
3. Terminal	<p>These two options allow the microcomputer to function as a timesharing terminal and to move files of data between the mainframe and the microcomputer, other than through the data retrieval module. Neither will be discussed here.</p>
4. File Mover	
5. File Utility	<p>Used to create and maintain files on the microcomputer. We used this above to create our login file, and will not discuss it again.</p>

We'll begin our retrieval process with the VisiCalc template of Figure 3-39 named TODAY.VC, and demonstrate how to use MAINLINE to obtain today's high, low, and closing price for each of the two stocks listed. Our template contains simple formulas to compute the market value (in the column labeled MARKET) by multiplying the closing price by the number of shares held. We have built the sheet so that values retrieved in the DIF format will be placed in the marked area at the top of the sheet.

We've established simple formulas in the report area of this sheet to copy these retrieved values into the desired areas below. This is the same concept we've used in previous case studies. We'll briefly explain the technique again later in this case study, and also review the purpose of the top two lines on the screen.

```
>A4
/S#L TODAY.DIF,D2 RETURN C
```



Load DIF
file here.

PORTFOLIO OF COMPUTER STOCKS

STOCK	SHARES HELD	-----TODAY-----			
		HIGH	LOW	CLOSE	MARKET
DEC	40	0	0	0	0.00
IBM	50	0	0	0	0.00
					=====
					0.00

Figure 3-39

MAINLINE will create the DIF file on a diskette for us, and then it is our responsibility to load the DIF file onto the VisiCalc template.

We'll now concentrate on the Request Builder section of the MAINLINE software and demonstrate how it can be used to update the portfolio of stocks that we're keeping on this VisiCalc template.

After entering MAINLINE we'll select the Request Builder module. We're prompted for information about our request. Figure 3-40 shows the MAINLINE screen after all of the following steps have been completed.

Prompt	Our Response	Explanation
SYSTEM:	CPS	We'll retrieve data from the CompuServe (CPS) system (SYS).
DATE BASE:	SDB	Use the VALUE securities data base (SDB) as the data base (DB).
NAME LOCATION P)AGE R)OW C)OL	R	Our stock names will appear on the rows (ROW) of the request.
ITEM LOCATION P)AGE R)OW C)OL	C	Our items (high, low, close) will appear in the columns (COL) of the request.
TIME LOCATION P)AGE R)OW C)OL	P	One time (TIME) will apply to this entire "page" (PAGE) of information.

```

                                REQUEST BUILDER
SYS  CPS      DB  SDB      MEM  92%
NAME ROW      ITEM COL      TIME PAGE
COMMANDS  B C D E G H I L Q S

PAGE: B*

                                A      B      C
                                HIGH   LOW   CLOSE

1 DEC
2 IBM
3
4
5
6
7
8
9
10
11
12
13
14
15

```

Figure 3-40

We now must enter the time (date) that we desire for this data retrieval. MAINLINE allows wide flexibility in the date format with a large number of options available. For example we can request a specific date, a most recent date, an earliest date, a date range, a relative date, or a computed date, with each specified in a variety of methods. For our example we'll use

B*

which means the most recent business date (Monday through Friday).

We then enter the words HIGH, LOW, and CLOSE as "items" across the columns, and DEC and IBM as "names" down the columns.

Entering these labels occurs in a manner similar to entering VisiCalc data, with a wide cursor moving across the top or down the side of the screen.

We see the full template for our modest example on the screen of Figure 3-40. As with VisiCalc the MAINLINE sheet is 63 columns by 254 rows, although for this example we're using only a small part of the largest potential sheet. MAINLINE scrolls over the sheet, as does VisiCalc.

At this point we've completed our MAINLINE request and we want to save this entire template to a diskette. To do so we enter a

/ (slash)

as in VisiCalc and then have single letter prompts of

B C D E G H I L Q S

with many parallels to VisiCalc. For example

Command	Means
B	Blank the cursor location.
D	Delete a row or column.
S	Save the template.

There are many other MAINLINE similarities to VisiCalc, which can serve to minimize learning time for users familiar with VisiCalc.

We'll type S, for save, and then type the desired filename or use the right arrow key to scroll the names on the diskette. We'll name this file TODAY.ML, and enter the filename by typing it.

It's important to understand that the request we've just built was done off-line, that is while working completely on the microcomputer and not on the mainframe. In addition we've saved this request template,

meaning that if we wish to use it tomorrow we can do so with no changes. Should we wish to make changes, MAINLINE contains editing capabilities that allow changes to request specifications.

One way to forward this request to CompuServe is to leave the request module builder and enter the data retrieval module. We'll do so.

As we enter the data retrieval module we're prompted for a filename of the request file. We type

```
TODAY.ML
```

(or scroll the directory until we reach that name). We're then prompted for a filename for the output DIF file. We'll respond with

```
TODAY.DIF
```

The retrieval module now automatically begins to execute our request file. It links with the appropriate login file for this data base, transmits that file, transmits our data request, and saves the retrieved data in a DIF file on our data diskette. At several steps in this process we select from various options (for example we can switch data diskettes if we wish, etc.), and receive a variety of status data on the processing (for example size of the DIF file).

At the completion of the data retrieval, we'll exit from MAINLINE, leaving our data diskette in disk drive 2.

We then run VisiCalc and type:

```
/SL TODAY.VC,D2 RETURN
```

to load the VisiCalc sheet shown at the top of Figure 3-41. We'll then follow the two lines of directions we've placed at the top of that worksheet, causing the DIF file created by MAINLINE to load in the worksheet starting at our predetermined location of A4.

Once loaded, the six data points from the DIF file are copied to the desired locations automatically as indicated in the lower half of Figure 3-41. We have placed single coordinate references in the template to copy these values.

We can see that the current market value has been computed for the shares we hold of these two stocks and that the DIF format was used successfully by MAINLINE to interchange data from the CompuServe network to our microcomputer.

```
>A4
/S#L TODAY.DIF,D2 RETURN C
```

PORTFOLIO OF COMPUTER STOCKS

STOCK	SHARES HELD	-----TODAY-----			
		HIGH	LOW	CLOSE	MARKET
DEC	40	0	0	0	0.00
IBM	50	0	0	0	0.00
					=====
					0.00

Load TODAY.DIF.

```
>A4
/S#L TODAY.DIF,D2 RETURN C
```

B*	HIGH	LOW	CLOSE
DEC	120.875	117.625	119
IBM	98.5	97.125	97.375

} Loads from MAINLINE DIF file.

PORTFOLIO OF COMPUTER STOCKS

STOCK	SHARES HELD	-----TODAY-----			
		HIGH	LOW	CLOSE	MARKET
DEC	40	120.875	117.625	119	4760.00
IBM	50	98.5	97.125	97.375	4868.75
					=====
					9628.75

Figure 3-41

Additional Capabilities and Considerations in Using MAINLINE

In addition to the example discussed in this case study, there are a number of other considerations in using MAINLINE.

- It can be used with other mainframes and data bases where software is installed on the mainframe.
- The MAINLINE system includes a diskette with data dictionaries containing names of data items present on the mainframe data base. This allows for local error checking while creating the data request.
- MAINLINE can return values without headers, or it can return a row of labels at the top and a column of labels at the left like those shown at the bottom of Figure 3-41.

DATA INTERCHANGE FROM DIF FILES TO 1-2-3

In this section, we'll discuss the value of DIF files when used with "integrated" software packages. We'll list a number of potential uses for DIF files for this kind of software, and demonstrate file exchange with one of these products.

The examples in this case study were completed using 1-2-3 (version 1.00) on an IBM Personal Computer. (Trademark information is contained in the front matter of this book and product information is contained in Appendix C.)

What's Integrated Software?

Integrated software products that are currently reaching the micro-computer market combine a number of formerly separate functions into a single product. For example, 1-2-3 combines spreadsheet, business graphics, and data-base functions. A common user interface exists to all functions. This means that an individual with a need for all three capabilities can obtain them from the same product, and does not need to learn to work with separate products that most likely function differently.

As significant as the common interface is, these products are also important because they share the same data. This means that there's no need to transfer data from a spreadsheet to a separate data base since both applications use common data.

What's the Value of the DIF Format with Integrated Software?

With integration of functionality and of shared data, the DIF format may be of little significance to a user whose computing needs are met fully by that product, although for such a user, if the product does read and write DIF files, there may still be uses for the format. For example, Chapter 2, *Using the DIF Format in a VisiCalc Context*, contains suggestions for using DIF files for an individual who uses only VisiCalc. Similarly, DIF may prove to be useful in providing some data manipulation or file management capabilities not otherwise available with the integrated product.

However, DIF can be of significant value for the individual who uses software other than the single integrated package, or who needs to access data other than that entered into or generated by the integrated package.

Let's examine a number of instances of the potential value of DIF files when used with 1-2-3, which does read and write DIF files

- The data we need may be in the DIF format. We may have retrieved it from a network, such as the example we've seen earlier using MAINLINE to obtain financial data from CompuServe. Similarly, we may have obtained data from an I. P. Sharp data base using MICROMAGIC and MICROCOMM, or from the Dow Jones News/Retrieval network using the Investor's Interface. All of these products return data in the DIF format, and the communications capabilities of these products and their ability to build requests in a format selected by the user are not found in 1-2-3.

- Data that we need in 1-2-3 may already have been entered in another program. If that program is capable of writing DIF files, we can access that data without the need to re-enter it in 1-2-3.

- We may have been using a program that competes with 1-2-3 in one of its areas of functionality, and may want to move our applications to 1-2-3. Although 1-2-3 can accept data directly from some products without going through the DIF format, it cannot do that for all products. Therefore, we may find ourselves with a significant number of applications that we want to move to 1-2-3. If the first program can generate DIF files, it will simplify this procedure. Similarly, we may want to move from 1-2-3 to another product.

- Although 1-2-3 can read text files, there may be instances where our text file needs processing before it can be easily used by 1-2-3. For example, we may have variable length records, or records with unusual data types or formats. In such a case, we may want to use a product like LoadCalc to manipulate the text file and create a DIF file from it. That DIF file can then be read by 1-2-3.

- In other instances, we can use 1-2-3 as a utility program for converting a text file into a DIF file that we need for another program. We may need to perform date arithmetic, or sorting, or selecting, or some other action on data in a text file, and then read the manipulated data into a program that reads DIF files. 1-2-3 can be used for this utility purpose.

- 1-2-3 may not contain a capability required. For example, we may need to perform some statistical computation contained in speedSTAT, but not in 1-2-3. DIF can be the required bridge between the two products.

- Data we need may be on a mainframe, and a program to retrieve it and generate a DIF file may exist and be in regular use. Rather than re-write the program, we can use the program as is, and use the DIF file capabilities of 1-2-3 to access the data.

CASE STUDY 11 DIF Files to 1-2-3

For this case study, we'll begin with a DIF file, called PRODMEMO.DIF, and translate the file so that the data can be used in 1-2-3. A separate translation step is used in this process, so that the file is first transferred into a 1-2-3 file, which can then be read into 1-2-3. 1-2-3 files are called worksheet files, and are given a filename extension of ".WKS" to indicate their usage. We'll therefore be going from a ".DIF" file to a ".WKS" file. Our DIF file already exists, although for our purposes, its contents are not important.

When we run 1-2-3, we typically see an introductory screen, and then the screen shown in Figure 3-42. On this screen, we see a menu, with choices indicated on the line that contains

1-2-3 File-Mgr Disk-Mgr GRAPH Translate PC-DOS

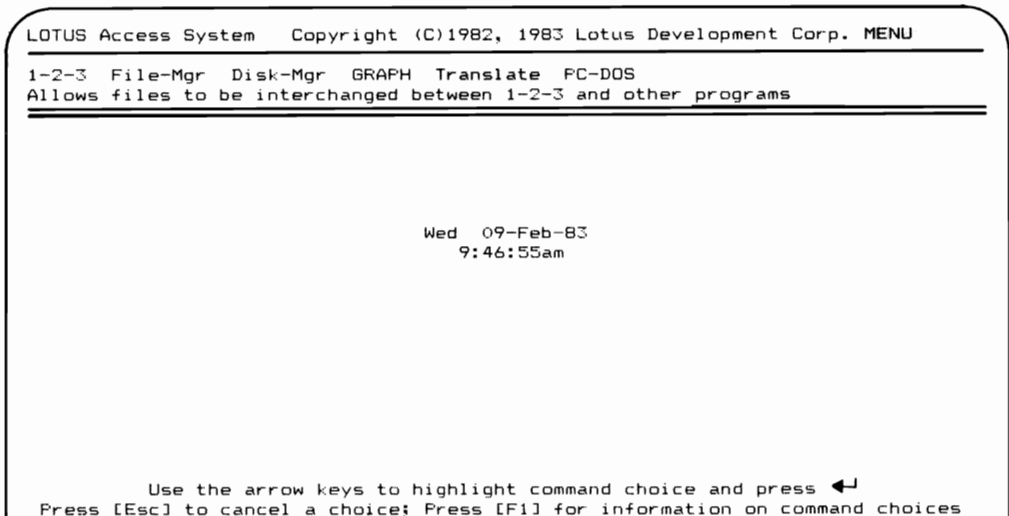


Figure 3-42

Notice that the choice Translate is highlighted on that line. This highlight, the cursor, moves from item to item as we press the left and right arrow keys. As we do so, the line beneath the menu displays additional information about the highlighted selection. Here that line displays:

Allows files to be interchanged between 1-2-3 and other programs

This is the action we desire, and we'll press the RETURN key (called the ENTER key on the IBM Personal Computer) to initiate this action. In all of the 1-2-3 menus, each item begins with a unique letter. By typing that letter from the menu (for example, by typing "T" here for Translate), we will enter that particular function without the need to move the cursor and then press the RETURN key.

We are then prompted to insert the LOTUS GRAPH Program diskette into drive A. We'll remove the System Disk with which we began, insert the required GRAPH diskette which contains the transfer utilities, press RETURN, and receive the screen shown in Figure 3-43. Notice that translations are provided from: VC (VisiCalc) files to WKS files (that is, 1-2-3 can receive VisiCalc files with formulas, and convert them directly into 1-2-3 worksheet files); from DIF files to WKS files; and from WKS files to DIF files. In addition, data can also be transferred from dBASE II to 1-2-3 in a multi-step operation. We'll discuss only the DIF transfers here.

```
LOTUS File Translation System (C)1982,1983 LOTUS Development Corp  MENU
-----
VC to WKS   DIF to WKS   WKS to DIF   Quit
Translate .DIF data file to 1-2-3 worksheet file.
-----
```

Figure 3-43

1-2-3 includes extensive "help" facilities (press the F1 function key at the point at which help is needed). If we press the F1 key here, the screen clears, and we see the screen of Figure 3-44, explaining the translation system. (Printed with permission of LOTUS Development Corp.)

```

LOTUS File Translation System (C)1982,1983 LOTUS Development Corp  MENU
VC to WKS  DIF to WKS  WKS to DIF  Quit
Translate .DIF data file to 1-2-3 worksheet file.
===== 319 =====
                                The Translate Functions
1. Choose Translate function.
2. Indicate which disk has the file to be Translated.
3. Select the file using [Space]. Press ← when done.
4. Choose a disk to store the Translated file.
5. Confirm (Yes), change file selection (No) or cancel (Quit) the Translation.

Translate converts VisiCalc<TM> spreadsheets and .DIF files to 1-2-3 worksheets,
and 1-2-3 worksheets to .DIF files. See the User Manual for dBASE-II<TM> files.

When Translate encounters a VisiCalc formula that it cannot translate --
@LOOKUP, @AND, @OR, @NOT -- it enters the formula as a label in the proper
cell. Translate provides a listing of all formulas that it cannot translate.
When you /File Retrieve the translated worksheet file, edit these labels into
formulas with the 1-2-3 equivalents -- @HLOOKUP, @VLOOKUP, #AND#, #OR#, #NOT#.

Before translating files, use the                               || =>
File-manager Rename function (if                               1-2-3 worksheet files:      .WKS
necessary) to provide correct                                 VisiCalc spreadsheet files:  .VC
filename extensions.                                       Files in DIF format:        .DIF

```

Figure 3-44

We'll now continue with our translation process, following prompts provided by 1-2-3 to select a source disk drive (we'll use drive B), select the DIF file from a list of files with ".DIF" as a suffix presented by 1-2-3 (any DIF file to be transferred must have a filename that ends with this extension), select the destination drive (we'll use B again), select the translation order (rowwise or columnwise), and complete the transfer. For all of these actions, we follow clear prompts that appear on the screen.

We'll follow directions to leave the translate function and return to 1-2-3. When we do so, we see the screen of Figure 3-45, on which we have moved the highlight to File. We'll select that choice (File), and then Retrieve from the next menu, as indicated in Figure 3-46.

We enter the filename, at which point the screen is cleared; and data from the DIF file, which has now been transferred into 1-2-3 format, is loaded onto the 1-2-3 worksheet, as shown in Figure 3-47.

If we wish to combine this data with other data that is already on the current worksheet file, we can do so by using the Combine option of the

```

A1:
Worksheet Range Copy Move File Print Graph Data Quit
Retrieve, Save, Combine, Xtract, Erase, List, Import, Disk
      A         B         C         D         E         F         G         H
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

```

Figure 3-45

```

A1:
Retrieve Save Combine Xtract Erase List Import Disk
Erase the worksheet and read a worksheet file
      A         B         C         D         E         F         G         H
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

```

Figure 3-46

```
A1: "Jan"                                READY

      A      B      C      D      E      F      G      H
1     Jan   900000
2     Feb   800000
3     Mar   865000
4     Apr   920000
5     May   970000
6     Jun  1030000
7     Jul  1150000
8     Aug  1150000
9     Sep  1000000
10    Oct  1000000
11    Nov   950000
12    Dec   950000
13
14
15
16
17
18
19
20
```

Figure 3-47

File command, rather than the File Retrieve option used above. The Combine selection allows us to specify the desired location at which the data should be loaded. It also allows us to control what occurs should data overlap. For example, we can specify that incoming data should replace existing data, be added to existing data, or be subtracted from existing data.

Additional Considerations When Using DIF Files with 1-2-3

If we retrieve a transferred file as above, manipulate the data within 1-2-3, and now want to create a DIF file from the new data, we must add one step to the process. We must save the file as a WKS file using the Save option of the File command. The WKS to DIF option of the transfer utility works only with WKS files that have been saved with the File command (it does not work with files created by the transfer utility from the DIF file).

When creating a DIF file from a worksheet file, we do not have options to select the portions of the data that are translated; instead, the full sheet is translated. This means that we might need to perform manipulations on the file within 1-2-3 before we create a file to be transferred.

1-2-3 is an extremely powerful program. Its use of DIF files provides access to its extensive capabilities for those of us with DIF requirements.

DATA INTERCHANGE FROM DIF FILES TO TK!SOLVER

In this case study, we'll move data from a number of DIF files to TK!Solver. The example below was completed using an IBM Personal Computer and TK!Solver, version TK-1(1E)/IBMPC. (Trademark information is contained in the front matter of this book, and product information is contained in Appendix C.)

CASE STUDY 12 DIF Files to TK!Solver

For this example, we'll be working with the data shown in Figure 3-48. This figure contains data listing the names and annual salaries of three groups of employees. We'll assume that this data has come from a large personnel file maintained on a mainframe computer, and that we're in the process of computing salary increments for our employees.

Professional		Part-Time		Other	
Sue	75000	Bobby	12000	Nancy	32000
Tony	37000	Sammy	20500	Mike	26000
Tom	16000	Chris	18000	Tom	19000
Carol	19000	Barb	5000	Marsha	30500
Nancy	20000	Marty	7500		
		Wendy	10400		
		Fernando	8000		
		Patrisha	11000		

Figure 3-48

Although the lists of data shown are very short, TK!Solver can handle lists of thousands of data items. We'll begin with three DIF files, one for each classification of employee, and each containing data saved by column. Thus each file really contains two lists, one of names, and a second of salaries.

Our problem is to decide how much to increase the salaries for each category while remaining within an overall constraint. For example, we may want to increase salaries by 10%, but want to increase the salaries for each grouping by a different percentage amount. To do so we need to know the total salaries paid to each category of employee, for with this knowledge we can then play with various increments for each group while remaining at an overall 10% increase.

TK!Solver is a powerful tool for this type of problem, and for many others. It "can solve a broad range of problems using models, sets of equations with variables of known and unknown values." When using

TK!Solver, we enter formulas, from which TK!Solver extracts and lists variables. We then enter known values and TK!Solver will find values of the remaining variables for which it can solve.

Figure 3-49 is the basis for our case study. Notice that the screen is divided horizontally into two parts, with VARIABLE SHEET as the title on the top, and RULE SHEET as the bottom.

```
(Br) Rule: 192/!
===== VARIABLE SHEET =====
St Input      Name      Output    Unit      Comment
-----
              dolprof
              dolpt
              dolothr
              dolgr
              pergr
              perprof
              perpt
              perothr
===== RULE SHEET =====
S Rule
- ----
* dolprof = sum('salprof)
* dolpt   = sum('salpt)
* dolothr = sum('salothr)

* dolgr   = dolprof + dolpt + dolothr

* pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr
```

Figure 3-49

Let's start with an explanation of the RULE SHEET, which contains a series of five formulas that we have entered. The first says

$$\text{dolprof} = \text{sum}(\text{'salprof})$$

which means that we want the variable *dolprof* (a name we've invented to mean the total current dollar salary for all of our Professional employees) to be the sum (this is a built-in function provided with TK!Solver) of all of the salaries of our Professional employees (indicated by *salprof*, the name of this list of values).

We enter our formula by typing the line character-by-character, using editing features as needed. When we're done with this first rule (we press ENTER or an arrow key as in VisiCalc), we move down the sheet and enter the second and third rules as shown, indicating that we want to obtain the total dollar amounts paid to our Part-time and Other employees respectively.

The next rule,

$$\text{dolgr} = \text{dolprof} + \text{dolpt} + \text{dolothr}$$

is a formula for computing the total amount of the salaries of all of our employees (the gross dollar total).

Finally, we see the formula

$$\text{pergr} * \text{dolgr} = \text{perprof} * \text{dolprof} + \text{perpt} * \text{dolpt} + \text{perothr} * \text{dolothr}$$

which indicates that the gross dollar increment must equal the total of the dollar increases that we give to each of our employee groups (that is, the proposed percent increase times the current total salary dollars for each group).

As an example, we may decide that we want to give an overall increase of 10%, but to give 12% to our Part-time personnel and 14% to our Other employees. If we do so, what percent increase should we give our Professional employees to be certain that we give the overall desired 10% increase?

We'll use the calculating power of TK!Solver to solve this problem for us.

Let's make several observations about the formulas above. First, we selected the variable names used and the formulas relating them. Although most of the formulas contain a single variable on the left of the equal sign, notice that the last formula does not. TK!Solver is capable of solving formulas in which the variable we're solving for is not necessarily isolated at the left of the equal sign. This is an extremely powerful capability, allowing us great flexibility in problem definition.

Let's turn our attention to the top half of the screen containing the VARIABLE SHEET. Notice that all of the variable names appear there (with the exception of names of the lists of salaries). The names shown were extracted from our formulas and placed there by TK!Solver as we entered our rules on the sheet below.

We'll need to use another sheet, the LIST SHEET, to associate the values from our DIF files with the names we've used with the sum functions in the first three rules. We'll use appropriate commands to revise the windows to display the LIST SHEET on a split screen with the RULE SHEET as shown in Figure 3-50.

DIF files used with TK!Solver are associated with the LIST SHEET. While on this sheet, and with our diskette with DIF files SALPROF.DIF, SALPT.DIF, and SALOTHR.DIF in place, we'll use the DIF Load option by typing

/S#L SALPROF Enter

When we do so, the two digits

5
5

appear under the column labeled Elements on the LIST SHEET as shown in Figure 3-51. Each row of this sheet represents a separate list of values, indicating that we have read two lists here, each containing five elements. Notice in Figure 3-48 that our data for Professional employees contained two columns, the first with five names, and the second with five salaries.

```

(1n) Name: 193/!

===== RULE SHEET =====
S Rule
-----
* dolprof = sum('salprof)
* dolpt   = sum('salpt)
* dolothr = sum('salothr)

* dolgr   = dolprof + dolpt + dolothr

* pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr

===== LIST SHEET =====
Name      Elements  Unit      Comment
-----

```

Figure 3-50

```

(1n) Name: 192/!

===== RULE SHEET =====
S Rule
-----
* dolprof = sum('salprof)
* dolpt   = sum('salpt)
* dolothr = sum('salothr)

* dolgr   = dolprof + dolpt + dolothr

* pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr

===== LIST SHEET =====
Name      Elements  Unit      Comment
-----
          5
          5
          8
          8
          4
          4

```

Figure 3-51

We'll follow similar steps to load the other two DIF files containing information about our Part-time and Other employees. Figure 3-51 shows the number of elements in these lists. Although we will not do so here, we could use TK!Solver to examine the contents of these lists to verify that they contain the data from the DIF files.

We must now add appropriate names to these lists, which we do by moving the cursor to the Name column of the LIST SHEET and typing the names we desire, as we have done in Figure 3-52. Notice that we have supplied names only for the second list of each pair, the list of salaries, while ignoring the first half of each pair, the list of names, which is not of interest to us.

```
(6n) Name: salothr192/!
===== RULE SHEET =====
S Rule
-----
* dolprof = sum('salprof)
* dolpt   = sum('salpt)
* dolothr = sum('salothr)

* dolgr   = dolprof + dolpt + dolothr

* pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr
===== LIST SHEET =====
Name      Elements  Unit  Comment
-----
salprof   5
          5
          8
salpt     8
          4
salothr   4
```

Figure 3-52

We have now completed the reading of the DIF files, and by providing them with names, we have established a method of associating the data with the variables used in the formulas on the RULE SHEET.

We'll now return to the VARIABLE SHEET and RULE SHEET shown in Figure 3-53. This screen shows these two sheets after we have pressed the

!

key. It is used in TK!Solver as the Action Command, and requests TK!Solver to solve the model we have created.

Notice on the VARIABLE SHEET that there are columns entitled Input and Output. We'll work with TK!Solver by providing values in the input column for the known variables, and then TK!Solver will attempt to solve for and display the remaining values as output.

In Figure 3-53, we see that this software has computed and displayed values for Output for four of the variables listed. (Recall that these variables are the sum of the salaries for our Professional, Part-time, and Other employees, and then the total of these three sums.)

At this point the software only has sufficient data to solve for these four variables.

```
(5i) Input: 192/
===== VARIABLE SHEET =====
St Input      Name      Output    Unit      Comment
-----
              dolprof  167000
              dolpt   92400
              dolothr 107500
              dolgr   366900
              pergr
              perprof
              perpt
              perothr
===== RULE SHEET =====
S Rule
-----
dolprof = sum('salprof)
dolpt   = sum('salpt)
dolothr = sum('salothr)

dolgr   = dolprof + dolpt + dolothr

* pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr
```

Figure 3-53

We'll now move to the VARIABLE SHEET and position the cursor over the input areas for the variables

pergr
perpt
perothr

and enter the values 10 (indicating a 10% increase desired overall for our employees), 12 (indicating the desired increase for our Part-time employees), and 14 (our desired increase for our Other category employees), as shown in Figure 3-54.

At this point, we initiate another request for the Action Command (press !), and the screen changes as shown in Figure 3-55, with a solution for the necessary percent increase in Professional salaries to ensure that all conditions established in our model are satisfied. We could continue with this sheet, revising any three of the percents, and requesting TK!Solver to find the fourth.

This example demonstrates one of the many ways that DIF files can be used with TK!Solver, and concludes all of our case studies.

```

(Bi) Input: 14 192/!

===== VARIABLE SHEET =====
St Input      Name      Output      Unit      Comment
-----
              dolprof  167000
              dolpt   92400
              dolothr 107500
              dolgr   366900
10            pergr
12            perprof
14            perpt
              perothr
===== RULE SHEET =====
S Rule
-----
dolprof = sum('salprof)
dolpt   = sum('salpt)
dolothr = sum('salothr)

dolgr   = dolprof + dolpt + dolothr

* pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr

```

Figure 3-54

```

(Bi) Input: 14 192/!

===== VARIABLE SHEET =====
St Input      Name      Output      Unit      Comment
-----
              dolprof  167000
              dolpt   92400
              dolothr 107500
              dolgr   366900
10            pergr
12            perprof  6.3185629 ←
14            perpt
              perothr
===== RULE SHEET =====
S Rule
-----
dolprof = sum('salprof)
dolpt   = sum('salpt)
dolothr = sum('salothr)

dolgr   = dolprof + dolpt + dolothr

pergr * dolgr = perprof*dolprof + perpt*dolpt + perothr*dolothr

```

Figure 3-55

Chapter 4

Guidelines for Using DIF Files

In this chapter, a series of guidelines is presented for users of the DIF format as a data interchange. Our purpose is to ensure that the tasks of data exchange between different software occur as smoothly as possible.

In addition to the information below, Chapter 2, *Using the DIF Format in a VisiCalc Context*, contains many suggestions for VisiCalc users who make use of DIF files.

GUIDELINES

Guideline	Explanation
<p>The DIF format works; however, we may need to be patient and be willing to experiment to make it work for us.</p>	<p>The data exchanges in Chapter 3, Case Studies, demonstrate the wide variety of situations in which DIF files can be used to successfully exchange data between otherwise incompatible software.</p> <p>However, even in programs designed specifically with exchange features that use DIF, or other interchange formats, the user must be prepared to face some frustration in exchanging data. For example, a VisiCalc to VisiFile transfer required four regenerations of the DIF file of the VisiCalc data before the file was accepted by VisiFile.</p> <p>Although two of these occurred because of an uninformed user, in both cases error messages generated were not in the error message section of the VisiFile product documentation. The other errors were inadvertent, although because of the flexibility of the DIF format, and of the two products, they could occur easily.</p>
<p>Know the programs used to access the data files.</p>	<p>The interface places responsibility on the user to be knowledgeable. In addition, the transfer does not occur without the full care and attention of the individual performing the exchange.</p> <p>Users should study carefully the sections of the documentation which detail the DIF file interchange for any products used.</p>

“Clean” the file in the program in which it’s easiest to do so.

The “Error Messages” section of the documentation, where available, can be very helpful in preventing errors on our part. For example, the PFS:Graph “Messages” section lists 28 messages, including eight which reference the DIF format. For each, corrective action is provided, an excellent source of information that can help us to prevent errors and to understand the process of interchange as applied to that product.

For many interchanges, the data will need processing to prepare it for use in another program. A VisiCalc worksheet may contain blank entries, subtotals, underlines, etc., that we do not want in the application program for which we’re creating the DIF file. This means the data must be “cleaned” at some point, either before creating the DIF file, as an intermediate step before using the file, or after we have accessed the file in the second application program.

As another VisiCalc example, we may have data displayed on the screen with a format that presents the data in a different way from its true value. For example an entry displaying

14

may in fact be

14.284

displayed with an integer format (/FI or /FGI). When the DIF file is created, the value to full precision is saved, not the displayed integer. If we

Guideline

Explanation

want the integer value then we must manipulate the data, for example with the Integer built-in function (@INT) of VisiCalc.

The decision regarding when this processing occurs might be based on which method involves the least work. A user with greater familiarity with one of the two programs used may choose to use that program for the editing function because of the knowledge regarding that program.

VisiCalc can be used to automate this process if the DIF file is mostly numeric data. With it, we can manipulate data before creating a DIF file, or after a DIF file is read onto the VisiCalc worksheet. A technique to perform this task with input and output areas is presented in Chapter 3, Case Studies, Data Interchange Between VisiCalc and VisiTrend/Plot, in Case Study 3.

Briefly, the advice there is to establish separate areas on the VisiCalc worksheet that send or receive values from a DIF file. These tightly packed areas eliminate extraneous data (underlines, blank areas, etc.), and can therefore eliminate potential data problems.

If one of the programs is user generated, for example a BASIC or Pascal program, and the other is a commercial program with editing capabilities then the commercial product may be

Document the file interchange process.

When creating the DIF file, also prepare a printed listing of the data and other information about the file.

chosen for this function since it may significantly reduce the user programming needed.

An additional consideration is the size of the file. If data are being sent from a large file on a central mainframe computer to a personal computer, then we may be forced to reduce the file size by removing unneeded records and fields while in the mainframe environment. This process may occur as part of the DIF file creation, or may be a separate step taken before creating the DIF file.

In applications where exchange will occur regularly, prepare detailed documentation to be followed rigidly while creating and using the DIF file. This will help to ensure that all of the steps necessary to share data will occur, and will occur in the proper sequence.

It is important to recognize that data interchange is a set of procedures and not only a data format.

When processing DIF files it is often necessary to know explicit details of the file, for example filename, number of fields, type of data, field widths, etc. Often it is also desirable to have a printed copy of the data. Adequately prepared documentation will meet this need. Without it we may find ourselves unable to successfully complete the interchange.

Sample documentation and additional considerations regarding this and the

Guideline	Explanation
<p>Be ready to prepare multiple DIF files to make the same data transportable to multiple programs.</p>	<p>previous guideline appear in Chapter 5, Documenting Data Interchange.</p> <p>Various software has different expectations from the information in the DIF file. A DIF file that successfully interchanges data with one plotting package may not work with another.</p> <p>For example, VisiTrend/Plot usually expects data for the Y axis alone, while PFS:Graph must have X axis and Y axis data. Each program has different requirements for the data it accepts.</p> <p>Although it may be unusual for us to use more than one plotting package, even within the same package we must often prepare more than one DIF file to transfer data. For example, PFS:Graph accepts data in either row or column order, but cannot distinguish between the two. If we want graphs prepared from rows, we must create a DIF file in row order, and if we want graphs prepared from columns of the same data we must create a second DIF file in column order.</p>
<p>Use the ".DIF" suffix on the filename of all DIF files created.</p>	<p>Some commercial software will not read a file as a DIF file unless its name ends with these four characters. Some, but not all, application programs append this automatically to files created as DIF files. For example, VisiCalc on the IBM Personal Computer does add this, but on the Apple II Plus 16 sector version it does not. The user must do this to transfer files to some other programs.</p>

Use an "R" or "C" before the ".DIF" in the filename of a DIF file.

An "R" or "C" as part of the filename clearly identifies the orientation of the DIF file as *Row* or *Column* order. For example a DIF file with sales information created in column order could be named

SALESC.DIF

Use short filenames.

On the Apple II, VisiCalc allows up to 30 characters in the filename. However, other products that read DIF files on the Apple II have requirements for shorter filenames, for example 10 characters. This means that a transfer cannot be accomplished if the filename exceeds 10 characters.

Prepare backup copies of the files before a transfer is attempted.

The processes of manipulating data that occur in passing data between programs can inadvertently result in the destruction of the file. Establish procedures to ensure that backup copies of all files are available. Develop the habit of using the program that creates the DIF file (or the system copy utility) to create a second, backup copy of each file as it is created.

Identify backup files with a consistent naming scheme.

The prefix

BU

might be appended to the beginning of all files that are backup copies. A file named

SALESR.DIF

could be backed up to a file named

BUSLSR.DIF

Before loading a DIF file onto an area of a spreadsheet already in memory, save the non-DIF worksheet for backup.

Suppose we have an empty vertical area in a worksheet in memory that we have prepared to receive a DIF file, and that we load the file by row when we need to load by column. This can

Guideline	Explanation
<p>When the data transfer is complete, save the data again in the file format of the receiving program.</p>	<p>happen easily with DIF files, and may cause the file to load horizontally and destroy an area of the spreadsheet already in memory.</p> <p>The DIF format serves as a format for data exchange, but may not be the most efficient storage arrangement for the data once it has been transferred.</p> <p>For example, if VisiTrend/Plot is reading a DIF file and does not encounter information it needs, it requests the user to supply that information. If we read the same DIF file repeatedly into VisiPlot, we'll need to supply that data repeatedly, an inefficient, and potentially error-prone activity. Instead, after reading the DIF file once and supplying needed information, we should use the data management capabilities of VisiTrend/Plot to save the data in the VisiTrend/Plot file format.</p>
<p>VisiCalc, and other programs, can be helpful in troubleshooting data interchange using DIF files.</p>	<p>If we are trying to exchange data between two programs, neither of which is VisiCalc, with a DIF file and have been unsuccessful, we can try to read the file into VisiCalc. This is an easy way to verify whether we do indeed have a valid DIF file. In addition, we can view the data on the VisiCalc screen, providing us with an opportunity to verify the file contents.</p> <p>If we have incorrect contents, we may be able to correct them while in VisiCalc and create the desired DIF file from VisiCalc.</p>

Since DIF files are text files, they can be read by many word processing programs. This gives us access to the file, and if we are familiar with the technical specifications of DIF files we can use the word processor to modify the file if needed, or even to create a DIF file from scratch.

CONCLUSION

The interchange of data, although conceptually easily achieved with a potential standard like the DIF format, is in fact a *process* of which the data format is only one part. Procedures need to be established, knowledge developed, and user care and attention must be devoted to the process to ensure that data can be successfully interchanged.

As a final guideline we'll summarize much of the information in this chapter.

Develop a plan of attack when attempting data interchanging between two products for the first time. Such a plan might follow these steps:

1. Read all sections of the documentation of each product dealing with DIF, looking for "DIF" or "Data Interchange Format" in the index and table of contents of the documentation.
2. Read the error message section of the product documentation (if present) to learn of potential DIF problems.
3. If the documentation contains sample exercises regarding the exchange, or practice data on an enclosed diskette, perform the exchange with that data to gain experience with a proven example.
4. If your exchange is a complicated one, involving field and record selection, try to simplify your problem and establish the exchange without all of the details. After successful contact between the programs with part of your data, expand to your complete problem.
5. If there is a difficulty, determine if the problem is in creating the DIF file or reading it. VisiCalc can be helpful since it allows relatively easy access to DIF files and displays the data from the file on the screen.

6. If problems persist, start from scratch, including reinitializing diskettes (where used), and regenerating the DIF file.
7. If necessary, call your product vendor or the software producer for help. The DIF-format aspects of many products are relatively esoteric, and often the software producer is able to answer questions more readily than your vendor.
8. Once exchange has been successful, take time to document it as suggested in Chapter 5, Documenting Data Interchange.

Chapter 5

Documenting Data Interchange

Documentation of user activities can be extremely helpful in the successful completion of data exchange. Because this interchange may often involve two distinct software packages, each with a different user interface, the potential for a user misstep may be high. Users may not be fully knowledgeable about both programs, or may use the application programs infrequently. The software may be used in an environment where many individuals may share responsibility for this process. Even where one individual does all of the interchange, well documented procedures can be helpful by “automating” the process and reducing the likelihood for errors or frustration.

Documentation must serve the user, and should be simple, direct, and yet complete. The level of documentation will depend on many factors, varying from no documentation, for example when data will be interchanged only once, to sophisticated detailed instruction sets where a variety of users and many file exchanges are involved.

With larger computers, the flow of data from program to program may be a multistep, complicated series of procedures. For example, we may run a program, generate an output file, read this file into a second program, generate another output file, etc. On these large systems, procedures can be automated so that an operator’s only action may be to start the procedure—then the system assumes full control.

However, microcomputers, where much of the DIF activity occurs, currently function quite differently. The user has a major responsibility for overseeing and controlling the procedures. Most application software expects a user to select options, step through menus, identify files, etc. A user must be continually alert throughout the process.

For this reason, well documented procedures are often extremely important since they can assist to semi-automate the procedures by providing a support, or prop, on which a user can lean to ensure that actions and decisions are correct. The establishment of formal proceedings and the preparation of documentation to reflect them are important steps in the data interchange process.

This chapter suggests ideas and formats which can be used in developing documentation from the perspective of data interchange with DIF files.

DOCUMENTATION WITHIN A VISICALC ENVIRONMENT

We'll start by looking at an example which will use DIF files entirely within a VisiCalc context. Suppose that we want to use DIF files to "pound" values into place within one area of the VisiCalc worksheet. (The section "Pounding Values in Place" of Chapter 2, Using the DIF Format in a VisiCalc Context explains why we may wish to do this.)

In Figure 5-1 we have part of a larger sheet visible on the screen. This sheet has a formula at each entry that forecasts our budget for each item for a 10-year period. Suppose we are satisfied with the values for Fiscal Year 0, Fiscal Year +1, and Fiscal Year +2. We want to replace the formulas in these columns with their values, since doing so will speed up recalculation time for the sheet and will provide additional memory.

Notice the three lines

```
>C12 RETURN /S#S temp RETURN E100 RETURN RETURN
/S#L temp RETURN RETURN
/SD temp.dif RETURN Y
```

which appear at the top of the screen. They contain the exact steps which a user should follow to save a section of the worksheet temporarily (TEMP) as a DIF file, reload that DIF file into the same area of the sheet (destroying formulas and formats, and therefore pounding the values into place as desired), and finally to delete the DIF file.

This simple documentation assists the user by providing a series of easy to follow instructions which significantly reduce the decision-making selections that would otherwise need to be made.

DOCUMENTATION WHERE SEVERAL PROGRAMS SHARE THE FILE

When several programs share a file, a format like that of Figure 5-2 can be used to record details of the interchange from the perspective of each application program. This form records information about:

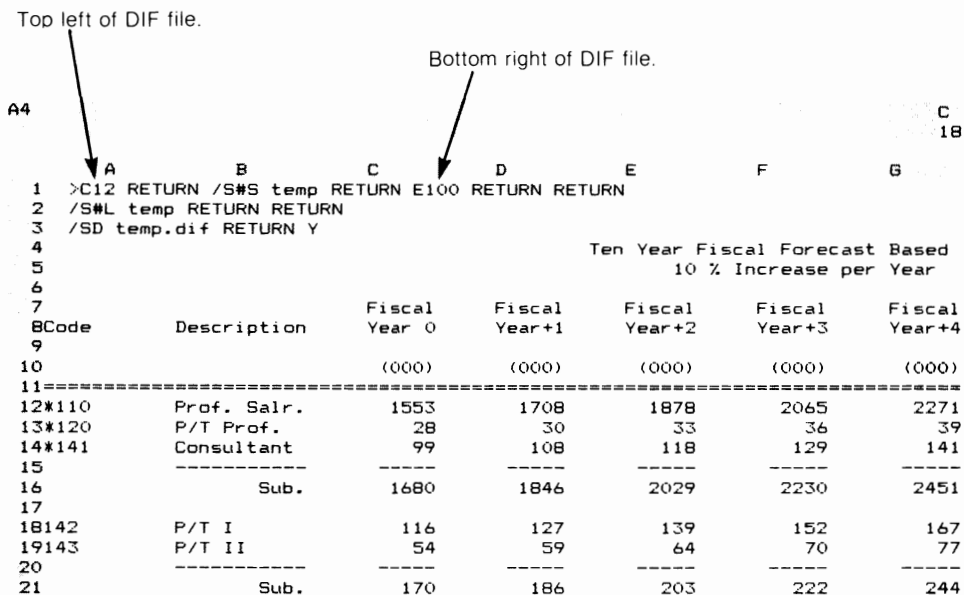


Figure 5-1

- The individual who prepared these procedures.
- The procedures to create the DIF file.
- The procedures to access this file.
- Activity involving the exchange process.

CONCLUSION

The often involved procedures for exchanging data can be simplified by preparing documentation that accurately reflects the steps to be followed to interchange data. If the software allows internal documentation (as demonstrated above with VisiCalc in Figure 5-1), take advantage of this capability to assist the user. Where several applications packages are involved, use external documentation to clarify the exchange process.

Prepared by _____
Phone _____
Date _____

Creating the DIF File.

Program Created From _____
DIF Filename _____
Diskette Id _____

Backup Filename _____
Backup Diskette Id _____

Orientation (Row or Column) _____

Field Names (in order)	Width	Data Type
_____	_____	_____
_____	_____	_____
_____	_____	_____
_____	_____	_____
_____	_____	_____
_____	_____	_____

Steps to create

Other Notes

Attachments (File Format Listings, Actual Data, etc.)

Figure 5-2

Accessing the DIF file.

Program accessing _____

Steps to access

Other Notes

Usage Log

Date	User	Notes, comments, problems

Figure 5-2 continued

Chapter 6

A Tutorial on the DIF Format

In this chapter we'll examine the file structure of DIF files. We'll do so by presenting collections of data, and then representing them as DIF files. Since DIF is a conceptual representation of data, we do not need a computer for this task.

We'll see that DIF files are appropriate for representing some, but *not* all, types of data. We'll also observe that the representation has limitations, (mostly intentional), which are sacrifices made for simplicity and transportability of the data. We'll need to remember that the intention of the DIF format is easy interchange of data between application programs, and not necessarily efficient, elegant representation.

This tutorial is intended as an introduction to the DIF format. Appendix A, the *DIF Technical Specification*, contains full details and is the source definition of the DIF format. This chapter is intended to be a presentation of the major elements of DIF files, and hopefully it will make Appendix A easier to understand. The tutorial is therefore intentionally incomplete.

All DIF files are organized in two sections:

Header Section
Data Section

The Header Section appears first and contains general information about the file, for example filename, size, etc. The Data Section follows, and contains the data itself. We'll begin our tutorial by studying the Data Section.

A PLACE TO START: THE DATA SECTION

Let's start with a single numeric value, the number 12, and display a part of the DIF file associated with that value, as shown in Figure 6-1.

We notice a number of items in the DIF representation. First, we see three fields—the 0, the 12, and the V—one of which is our numeric value. These fields are called the *Data Value*. Second, the three items are split into two lines. Third, the representation is a combination of numeric values (0 and 12), and string character data (the V). We have placed an ellipsis (...) above and below the DIF representation to indicate that the right column in that figure is *not* a full DIF file, but only the part of it associated with the value shown.

Value	DIF Representation
	.
	.
	.
12	0,12
	V
	.
	.
	.

Figure 6-1

Let's look at Data Values for some additional numbers, as shown in Figure 6-2. We can add some additional observations. Numbers in the DIF file are represented in base 10 notation. Negative numbers are included with a - sign (the + sign is also acceptable). Decimal points and scientific notation are acceptable.

Value	DIF Representation
	.
	.
	.
126	0,126
	V
	.
	.
	.
-4.68	0,-4.68
	V
	.
	.
	.
1.427E-21	0,1.427E-21
	V
	.
	.
	.

Figure 6-2

All Data Values representing numeric data have the 0 (zero) and V in the DIF file as shown in Figures 6-1 and 6-2, although we will not explain the reason here. Refer to Appendix A for the full explanation.

The three fields we've seen above which compose the Data Value, are named below.

Type Indicator, Number Value
String Value

In the examples of Figures 6-1 and 6-2 we have examined numeric values and shown their representation in the DIF format. Now let's move from these discrete values to a file of values. Suppose we have production data for the four quarters of the year. The units produced are

15
20
23
19

There are two ways of conceptualizing this data, each of which results in a different DIF file. We can think of the data above either as

one column of length four,
or four rows each of length one.

Additionally, we could present this data as below.

15 20 23 19

All we have done is to write the data horizontally instead of vertically. We can now represent this either as

one row of length four
or four columns each of length one.

The words row and column are not meaningful in these contexts; instead we need to know if we have one group of four items, or four groups each of one item. The distinction in data processing terms is between one record with four fields, or four records each with one field.

In DIF format terminology, records are called *Tuples* and fields are called *Vectors*. Together they give us a *Table* of data. We will use the notation below in the remainder of this chapter.

tuple (record)
vector (field)

The user determines the orientation of the data, because the data alone (the four numbers in our example) do not tell us its organization. However, the DIF file does include information that defines this organization. This is different from many file organizations in which we are able to distinguish records (through some end-of-record mechanism often invisible to the programmer), but are unable to distinguish fields within the records. In those instances, our programs create record overlays that break the string of data of a record into fields. This does not happen with the DIF format. We've seen that each piece of data is isolated with the use of the 3-part Data Value defined earlier in this section.

We will demonstrate two representations of our data. Let's start by assuming that we have one tuple (record) with four vectors (fields). Figure 6-3 displays our four values and their DIF file representation. It is as expected, based on the above Data Value discussion. Let's add several items to this representation, and we will have the complete Data Section of the DIF file, as shown in Figure 6-4.

```

.
.
.
0,15
V
0,20
V
0,23
V
0,19
V
.
.
.

```

Figure 6-3

```

.
.
.
-1,0
BOT
0,15
V
0,20
V
0,23
V
0,19
V
-1,0
EOD

```

Figure 6-4

We have introduced two new elements into the DIF file in Figure 6-4. The first, above our data is

```

-1,0
BOT

```

This 3-part Data Value indicates *Beginning Of Tuple* (record). The second new element, another Data Value is

```

-1,0
EOD

```

and indicates *End Of Data*. The items BOT and EOD are called *Special Data Values*, and are used respectively to signal the beginning of a tuple (record) and the end of data. The Data Value including EOD as the String Value, occurs only once in a DIF file as a required entry at the end of the

file as shown in Figure 6-4. Note that we have removed the ellipsis at the bottom of the data since we are now showing the true end of the DIF file.

Also notice that the Type Indicator is -1 for both, and the Number Value is 0. These are requirements of the DIF format as explained in Appendix A.

The EOD value allows programmers to test for the end of file, and it is useful for systems that do not include capabilities for determining the end of file. Those who have taken computer programming courses will recognize this testing, which often occurs in the form of a unique value as the last piece of data signaling the end of a file of numeric values.

Let's use the same data, but now change the orientation from one tuple (record) with four vectors (fields) to four tuples (records) each with one vector (field). That representation is shown in Figure 6-5.

```
.  
. .  
-1,0  
BOT  
0,15  
V  
-1,0  
BOT  
0,20  
V  
-1,0  
BOT  
0,23  
V  
-1,0  
BOT  
0,19  
V  
-1,0  
EOD
```

Figure 6-5

We now see that the Data Value

```
-1,0
BOT
```

occurs four times, since we have four tuples here.

We are also beginning to sense the overhead of a DIF file. The part of the file that we see here uses 27 separate data items to represent our four quarterly data points, and we have not added the Header Section yet. This is a worst case example since each tuple (record) has only one vector (field). Let's also recognize that this count of data items is an incomplete method of determining overhead since it gives neither an indication of storage used (which is dependent on the length of the parts of the Data Value), nor of the time required to read the file.

One additional comment about overhead is necessary. The file is designed for exchange of data between programs. This means that we may have data in three different formats to complete the exchange: the format of the source application, the DIF format, and the format of the target application. We can see that low overhead is not a high priority design criterion for the middle format—the DIF format—for potentially the data will only be in the DIF format for the exchange alone. Thus the DIF file may be only a temporary representation of the data.

Let's expand our original data to include string data as well as numeric data. We'll add the words FIRST, SECOND, THIRD, and FOURTH to our quarterly data, so that we now have eight pieces of data as shown below, and not four.

```
FIRST    15
SECOND  20
THIRD   23
FOURTH  19
```

First, we'll need to present how the non-numeric string data above is represented in the three-part Data Value. This is done as follows:

```
1,0
"FIRST"
```

Here the Type Indicator is 1, the Number Value 0, and the String Value is the string itself, here in quotation marks.

With this additional knowledge, we can create the Data Section for the DIF file representation of the eight data items above. Again we'll need to impose a desired organization on this data, and think of it as two tuples (records) of four vectors (fields) each, or of four tuples (records) of two vectors each.

Let's look at both representations in Figure 6-6. On the left we've got four tuples (records) each with two vectors (field) and on the right two tuples (records) each with four vectors (fields). Notice that there is no end of tuple (record) marker but that the BOT marker serves as this separator. Also notice the single EOD marker at the end of the Data Section of the file.

.	.
.	.
.	.
-1,0	-1,0
BOT	BOT
1,0	1,0
"FIRST"	"FIRST"
0,15	1,0
V	"SECOND"
-1,0	1,0
BOT	"THIRD"
1,0	1,0
"SECOND"	"FOURTH"
0,20	-1,0
V	BOT
-1,0	0,15
BOT	V
1,0	0,20
"THIRD"	V
0,23	0,23
V	V
-1,0	0,19
BOT	V
1,0	-1,0
"FOURTH"	EOD
0,19	
V	
-1,0	
EOD	

Figure 6-6

If we think of this data on a VisiCalc screen for a moment, as in Figure 6-7, we are now able to connect the VisiCalc prompt

DATA SAVE: R, C OR RETURN

that we see when we are creating a DIF file (after */S#S name*) with the resultant Data Section of the DIF file. If we save the data shown on the VisiCalc screen by columns (press C) we get the representation of data at the left of Figure 6-6, while if we save by rows (by pressing R on RETURN) we get the representation at the right of Figure 6-6. Although this seems to be reversed, that is the way that VisiCalc creates these files.

The four strings above could also be placed as labels in the Header Section. We'll discuss this briefly later; also refer to Appendix A for an example.

FIRST	15
SECOND	20
THIRD	23
FOURTH	19

Figure 6-7

ADDITIONAL CONSIDERATIONS OF THE DATA SECTION OF A DIF FILE

The *DIF Technical Specification* (reprinted as Appendix A) contains additional information about the topics covered above, and also Data Section details regarding

- Specifications for representing NA, ERROR, TRUE, and FALSE.
- General information about generating definitions for additional types of values that may be needed by users.

THE HEADER SECTION

The Header Section is the first section of the DIF file. Its basic building block is the *Header Item*. Multiple Header Items describe the organization of the DIF file, with each describing some aspect of the file. Four Header Items are required, and a number of optional ones also exist. The

specifications detail how users can invent new ones as needed and have them added to the *DIF Technical Specification*.

Let's look at the four Header Items required for the quarterly production data below, assuming that the data is organized into four tuples (records), each composed of two vectors (fields).

```
FIRST    15
SECOND  20
THIRD   23
FOURTH 19
```

Figure 6-8 shows the minimum Header Section for this data with the organization desired. We can make a number of observations: each Header Item is composed of four fields spread over three lines; the four required Header Items are named TABLE, VECTORS, TUPLES, and DATA, with these names, called Tokens, as the first line of the Header Item.

```
TABLE
0,1
"PRODUCTION"
VECTORS
0,2
" "
TUPLES
0,4
" "
DATA
0,0
" "
.
.
.
```

Figure 6-8

Let's look at each of these four items.

The first item is the TABLE Header Item. It must be first in the file. The second line always contains

0,1

and the third line contains a title, between quotes, for the DIF file.

The next two items in our file have the general format

```
VECTORS
```

```
0,count
```

```
“ ”
```

```
TUPLES
```

```
0,count
```

```
“ ”
```

where “count” indicates the number of vectors (fields) and tuples (records) in the file, respectively. Together they give us the size of the file. These are required items.

If we are creating a DIF file, there may be times when we do not know how many tuples (records) there are until after we’ve created the file. For example, when creating a DIF file from a large data base on a mainframe computer we may not have a tuple (record) count until we’ve written the whole Data Section of the file. In such a situation it is still our responsibility to accurately write the tuples (records) Header Item with the correct count. If we were able to count tuples (records) while creating the file, we finish writing the Data Section, close the file, then reopen the file, position to the correct line of the file, and write the correct count.

If we were not able to count the tuples (records) while creating the DIF file, then we must read all of the newly created DIF file, counting the tuples (records) by using the BOT marker as a separator, and then return to the tuple (records) Header Item to write the correct count. The steps necessary to do so will depend on the operating system and programming language used.

The last Header Item in Figure 6-8 is

```
DATA
```

```
0,0
```

```
“ ”
```

This must be the last item of the Header Section. Some programs reading DIF files may only be interested in the Data Section, and they can ignore all Header Items until they reach this one that signals the data is next in the file. This completes the discussion of the required Header Items. Figure 6-9 contains the full DIF file we have been developing, assuming four tuples (records).

TABLE
0,1
"PRODUCTION"
VECTORS
0,2
" "
TUPLES
0,4
" "
DATA
0,0
" "
-1,0
BOT
1,0
"FIRST"
0,15
V
-1,0
BOT
1,0
"SECOND"
0,20
V
-1,0
BOT
1,0
"THIRD"
0,23
V
-1,0
BOT
1,0
"FOURTH"
0,19
V
-1,0
EOD

Figure 6-9

ADDITIONAL CONSIDERATIONS OF THE HEADER SECTION OF A DIF FILE

We have examined four Header Items, but there are currently nine additional Header Items in the *DIF Technical Specification*. That document describes how DIF file users can create additional items and have those items appear in the specifications. The current optional Header Items are LABEL, COMMENT, SIZE, PERIODICITY, MAJORSTART, MINORSTART, TRULENGTH, UNITS, and DISPLAYUNITS.

Let's look at LABEL. Suppose a data-base program has records on company personnel, and that each record has 40 fields. Perhaps we have used VisiCalc to calculate new salaries for each individual and we now have a DIF file with first name, last name, and new salary. The LABEL items allow the program reading this file to know the relative location of these three vectors (fields) within the tuples (records).

If the last name was the third vector (field) of each tuple (record), then our LABEL Header Item would appear as below.

```
LABEL
3,0
"LAST NAME"
```

The specification states that any program requiring optional items not found in a DIF file it is reading should prompt users for the missing items. As we've seen earlier, VisiPlot does this, but DB Master which requires LABEL items, does not, and users must manipulate the DIF files before they can be read by that data-base program.

ADDITIONAL INFORMATION ON THE DIF FORMAT

The *DIF Technical Specification* is included as Appendix A, and is the formal definition of the format. It includes programs in BASIC and Pascal that read and write DIF files, and Appendix B lists references to other programs. Users may wish to refer to those sources for added details of DIF files.

MANIPULATING DIF FILES WITH WORD PROCESSORS

Since DIF files are text files, they can be read by many word processing programs. These programs, as do other editors, allow us to read DIF files, to print them, to modify them, or to create them.

As an example, suppose we are working with VisiCalc and that we have retrieved a lengthy file in the DIF format from a mainframe com-

puter. The file has employee names and a department number for each, perhaps as below.

```

JOE      442
LIL      17
MARIAN  483
NOAH     100
GABE     100
MARTHA  483
TOMMY   483
    
```

Suppose that instead of the seven records shown above we had 300 records. VisiCalc has only 254 rows, and therefore to load a file of 300 records we would need to break it into several parts.

We'll demonstrate how to do this for a DIF file of the seven records we've shown above. What we want to do is shown in Figure 6-10, a VisiCalc screen in which we show five of the records at the left and the last two records at the right.

```

A249                                     C-
                                         29
                                         A      B      C      D      E
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250JOE      442      MARTHA  483
251LIL      17      TOMMY   483
252MARIAN  483
253NOAH     100
254GABE     100
    
```

Figure 6-10

A single DIF file of the seven records cannot be loaded onto the bottom of the VisiCalc spreadsheet as shown in that figure. VisiCalc does not contain that type of editing capability.

However, some word processing programs can be used for such purposes as shown in Figure 6-11. The DIF file from the mainframe is in the center of that figure. We could use a word processor to create two new DIF files from that one as shown at the left and right of that figure. Each can then be loaded separately onto the VisiCalc sheet.

For each we change the tuple count as shown. For the LEFT.DIF file we preserve the first five tuples by deleting the last two tuples with the word processor. We then save the new file as LEFT.DIF.

Reload the mainframe DIF file into the word processor, change the tuple count to 2, delete the first five tuples, and save the abridged file as DIF.RIGHT.

Now each file can be loaded onto the desired location of a VisiCalc worksheet.

Most word processors insert control codes of one kind or another in their documents when they save them to a disk; however, many allow text files to be "printed" to a disk. This is the desired action to take for these files, although even this capability does not ensure that the new files will be free of undesired characters. The example of Figure 6-11 was completed on an IBM Personal Computer using the Line Editor (EDLIN).

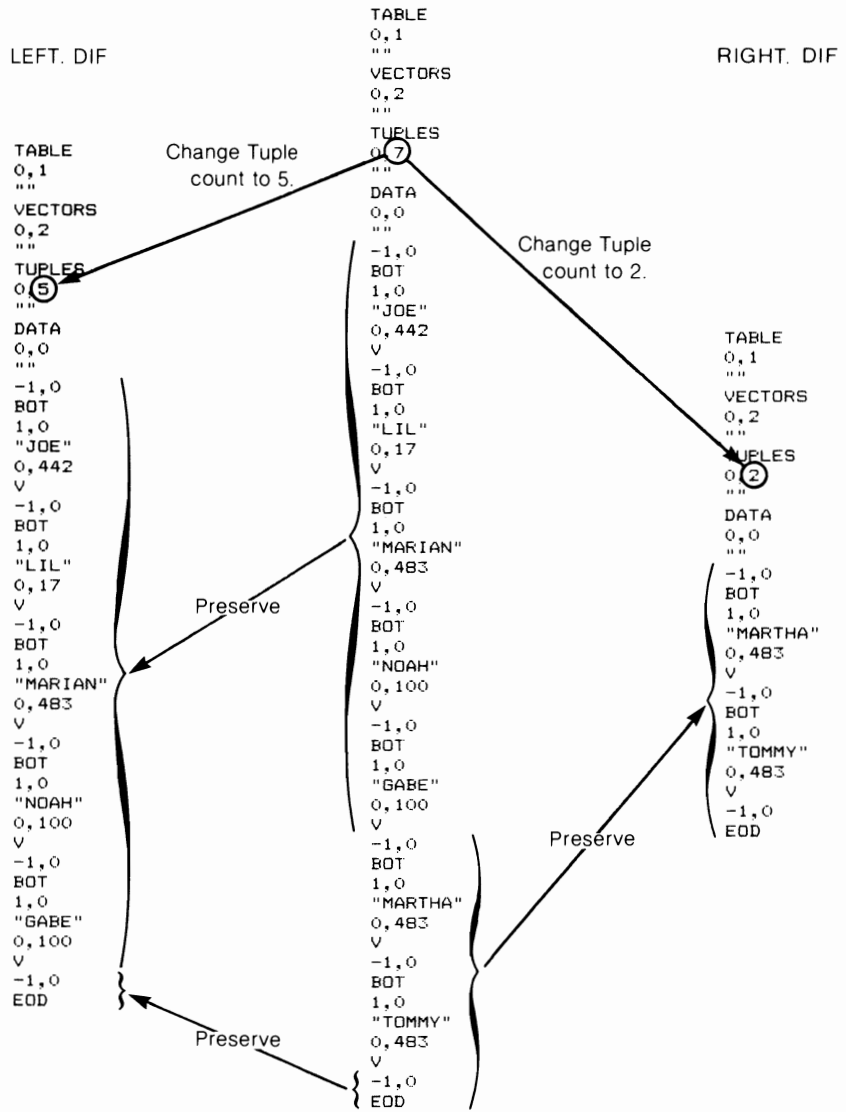


Figure 6-11

Chapter 7

Limitations of a System Using the DIF Format

We will examine in this chapter the limitations of a *system* in which the DIF format is used. It is important to realize that limitations do not mean negative criticisms. Instead, these observations should provide a perspective on the whole process of exchanging data. The DIF format is only a part of that action.

Readers should also refer to the *DIF Technical Specification*, Appendix A, which includes a section entitled "Constraints of the Format." Listed there are constraints imposed on the format by its designer.

Remember that the DIF format does work. It successfully allows us to interchange data in situations for which it was designed. The case studies presented earlier demonstrate the wide variety of applications and the diverse nature of the software that has the DIF format as a common link. It is this format that allowed us to develop case studies using many prominent pieces of software.

We will discuss the limitations of the format itself, and then look at other limitations that are at work when the interchange process occurs.

LIMITATIONS OF THE DIF FORMAT

The DIF format is designed for numeric data. This means that the format is not necessarily appropriate for the interchange of non-numeric data. In an earlier case study we created a DIF file from columns of numbers that were part of a word processing document. This is an appropriate use of the DIF format. However, if we wanted to exchange the full text of the document, the DIF format would not necessarily be usable. Data interchange often involves a combination of both numeric and non-

numeric data, and the DIF format may be awkward or unusable in such instances.

In addition to numeric and textual data in a file, we may also have unusual data that does not fit in any category. For example, in a VisiCalc file we may have a repeating label at any entry that contains

/--

When a DIF file is created from VisiCalc, this Repeating Label becomes

- (a single hyphen)

in the DIF file.

In this case, VisiCalc makes a conversion of the data, but does so without any guidance from the format. The *DIF Technical Specification* does not contain a specification that describes what should occur with this piece of data. Other software with unusual data formats may also need to make a conversion when creating a DIF file and may not find guidance from the current technical specification.

However, the specification does contain a section on defining new value indicators which can be used to meet these unique needs. Developers of new data types are encouraged by the *DIF Technical Specification* to register them with the DIF Clearinghouse.

Data must be rectangular with tuples (records) each containing an equal number of vectors (fields). If our data contains variable length records, we may need to do significant reformatting before we can generate the DIF file. In addition, the DIF file in such a case will contain unneeded overhead.

The format itself carries significant overhead, and the data being transmitted may amount to only a small percentage of the total DIF file. However, it is important to remember that the file is designed to accomplish interchange, meaning that the data may be in the DIF format only during the exchange. The source program and the target program may both use their own data formats, and the DIF file may be a temporary creation used only for the interchange.

DIF does not have formal acceptance as "the" standard for interchange. Although informal acceptance has been strong, as evidenced by its growing use by software producers, there are other proposed standards.

LIMITATIONS OF SOFTWARE USING THE DIF FORMAT

The DIF format works within an environment of software, either of purchased applications packages, or programs prepared by or for the user. This means that limitations outside the DIF format also have an impact on the use of the format. We will look at examples in this section.

The ability to exchange data does not imply that there is no editing of the data required. We've seen this repeatedly in our examples, where we had to condense data, remove unnecessary data, or reformat data. In some instances this reformatting may be significant work.

Different programs that "create DIF files" may create different DIF files from the same data. This applies to the Header and the Data Sections. For example, DIF files created from the same data by VisiCalc and VisiTrend/Plot will be different. It's important to recognize that although we are saying it is the same data, it is not the same data within the individual frameworks of the two programs. For example, if we have 12 monthly production figures, and the names of the months on a VisiCalc worksheet, and if we have the same data within a VisiTrend/Plot context, the data is different for both programs, although to an outside user it is essentially the "same" data.

Because DIF is often an ancillary capacity of the software product, it may not be as well tested, documented, etc., as the main features of the software. This is of course not a limitation of the format, but it is a reality when we try to complete an exchange.

Disk swapping is common in data exchange and we may need to work with many diskettes. The growing use of hard disks on microcomputers may offer a partial solution to this limitation.

Finally, it is important to realize that a data format is only a small part of the process of interchanging data when considered in the context of a larger system. An interchange format does not mean that various software packages which use the format function in a consistent manner. One program may use a menu for user control while another uses a command language and a third functions by questioning and then accepting responses from the user. Typing a key on the keyboard that means "continue to the next step" in one software package may mean "cancel this step" in another. All of these are limitations on the successful interchange of data, and not on the format used for the exchange of that data.

Appendix A

DIF_{T.M.} Technical Specification

DIF Technical Specification
© 1983 Software Arts Products Corp.
Reprinted with permission from (and thanks to)
Software Arts Products Corp.

DIF Clearinghouse

PO. Box 638
Newton Lower Falls, MA 02162

<p>DIF TECHNICAL SPECIFICATION</p>
--

DIFtm is the format for the exchange of data
developed by Software Artstm.

DIF-0283

© Copyright 1983 by Software Arts Products Corp.
All rights reserved.

Software Arts is a trademark of Software Arts Products Corp. and Software Arts, Inc.

DIF is a trademark of Software Arts Products Corp.

TK, TK! and TK!Solver are trademarks of Software Arts, Inc.

VisiCalc is a registered trademark of VisiCorp.

VisiPlot, VisiTrend/VisiPlot are trademarks of VisiCorp.

TREND-SPOTTER is a registered trademark of Friend Information Systems.

Limited License to Copy:

This Technical Specification is intended for the use of the original purchaser only. The original purchaser is hereby licensed to copy it for his own use, provided that this notice, together with the copyright, trademark and warranty notices, are reproduced on each such copy. Copying of this document in any form for purposes of resale, license or distribution is prohibited.

No Warranty:

This document is being published to enhance the usefulness of DIF, a format for data interchange, as used by the VisiCalc (R) and other programs.

NEITHER SOFTWARE ARTS PRODUCTS CORP. NOR THE DIF CLEARINGHOUSE MAKES ANY WARRANTY, EXPRESS OR IMPLIED, WITH RESPECT TO THE QUALITY, ACCURACY OR FREEDOM FROM ERROR OF THE DIF FORMAT OR OTHER CONTENTS OF THIS DOCUMENT, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR OF FITNESS FOR A PARTICULAR PURPOSE, AND SOFTWARE ARTS PRODUCTS CORP. SPECIFICALLY DISCLAIMS ALL LIABILITY FOR DAMAGES RESULTING FROM THE USE OF SUCH FORMAT OR OTHER CONTENTS.

VisiCalc is a registered trademark of VisiCorp.

DIF Technical Specification

Table of Contents

1. Introduction, 1
2. Constraints of the Format, 1
3. Organization of the DIF Data File, 2
4. The Header Section, 2
 - 4.1 The Header Item, 2
 - 4.1.1 The Topic, 3
 - 4.1.2 The Vector Number, 3
 - 4.1.3 The Numeric Value, 3
 - 4.1.4 The String Value, 3
 - 4.2 Header Items, 3
 - 4.2.1 The First Header Item, 4
 - 4.2.2 Vector Count, 4
 - 4.2.3 Tuple Count, 4
 - 4.2.4 The Last Header Item, 5
 - 4.2.5 Vector Label, 5
 - 4.2.6 Vector Comment, 5
 - 4.2.7 Field Size, 6
 - 4.2.8 Periodicity, 6
 - 4.2.9 Major Start, 6
 - 4.2.10 Minor Start, 6
 - 4.2.11 True Length, 7
 - 4.2.12 Units, 7
 - 4.2.13 Display Units, 7
 - 4.3 Defining New Header Items, 7
5. The Data Section, 8
 - 5.1 The Type Indicator Field, 8
 - 5.2 The Number Value Field, 9
 - 5.3 The String Value Field, 9
 - 5.3.1 Special Data Value, 9
 - 5.3.2 Numeric Value and Value Indicator, 9
 - 5.3.3 String Value, 10
6. Definitions, 10
7. Applications Programs, 11
 - 7.1 The CCA/DMS Program, 12
 - 7.2 The TREND/SPOTTER Program, 12
 - 7.3 The VisiCalc Program, 12
 - 7.4 TK!Solver, 12
 - 7.5 The VisiPlot and VisiTrend/VisiPlot Programs, 12
- I. Sample DIF File, 13
- II. Sample BASIC program that writes a DIF file, 15
- III. Sample BASIC program that reads a DIF file, 17
- IV. Sample Pascal program using a DIF file, 19

1. Introduction

This document is the technical specification of DIF, a format for the exchange of data, developed by Software Arts Product Corp. It is a reference document and not a tutorial. It includes a description of the DIF file organization and structure, required items, and optional standard items. It also explains the use of the optional standard items by specific applications. The last section is an example of a DIF data file.

Programs should use defined standard items when possible. The DIF Clearinghouse will update this document to describe new items as they are defined and record their use in specific programs. Programmers developing new software that incorporates new optional items should inform the Clearinghouse fully about them so that they can be standardized for common use by any program supporting the DIF format.

Programmers should remember that the program reading the data can be extremely simple. The program writing the data must handle it in such a way that it can be read by any program supporting DIF.

Within this text, upper case characters are actual values to be entered as shown and lower case characters name the value to be entered to a field. It is assumed that the ASCII character set is being used. See the section on Definitions for a discussion of character sets.

2. Constraints of the Format

The DIF format was designed for ease of use, and, for the sake of simplicity, certain constraints have been imposed on the format. Because DIF is not intended to be a universal representation for all data, one of these constraints is the representation of data in tables with rows of equal length and columns of equal length. A second constraint is that, because many users program in BASIC, the files must be compatible with BASIC programs. Programs written in another language, such as Pascal, can use a set of subroutines to read and write DIF files.

Below is a list of specific constraints on a DIF file.

1. Because some BASICs have only primitive facilities for reading and writing strings, the convention of keeping numbers and strings on separate lines has been adopted.
2. Two items, VECTORS and TUPLES, are required to support systems that require preallocation of space.
3. Because some systems do not allow programs to test for the end of a file, a special data value, EOD, provides graceful termination to a program.

4. To simplify programming, there are only two formats within the file, and all fields are predefined as character strings or numbers.
5. Strings must be enclosed in quotes if they contain characters other than alphanumerics.
6. The character set is restricted to the printable ASCII characters.
7. Although DIF places no explicit restriction on the length of data strings, some systems may impose restrictions.

Since the DIF format is not meant to meet all the needs for data representation, it may be necessary to use multiple DIF files or additional formats for some applications. A word processor, for example, would not use a DIF file to store text but could use DIF files for tables of values within a report.

3. Organization of the DIF Data File

A DIF file is a text file using the standard printable character set of the host machine. The model for the data is a table. Fields are called vectors; records are called tuples. Data is organized into vectors of equal length. Each tuple consists of a row of corresponding values read across each vector. The user determines the specific groupings of vectors and tuples. Often vectors are treated as columns and tuples are treated as rows, but because DIF can transpose columns and rows, the terms vectors and tuples are used instead of the terms columns and rows.

The DIF file consists of two sections, a header section and a data section. The header section contains descriptions of the file and the data section contains the actual values.

4. The Header Section

The header section is composed of header items. There are four standard required header items and several standard optional header items.

4.1 The Header Item

The header items describe the data organization. Each header item consists of four fields arranged on three lines as illustrated below. The first line is a

token¹, the second line consists of two numbers, and the third line contains a string.

```
Topic  
Vector Number, Numeric Value  
"String Value"
```

4.1.1 The Topic

The first line of the header item is the Topic. It identifies the header item, and must be a token.

4.1.2 The Vector Number

The first field on the second line is the Vector Number. If the header item describes a specific vector, the Vector Number specifies the vector being described. If the header item describes the entire file and not one specific vector, the Vector Number is zero (0).

4.1.3 The Numeric Value

The Value is an integer and occupies the second field of the second line, separated by a comma from the Vector Number. If the header item does not use a numeric value, the Value is zero (0).

4.1.4 The String Value

The String Value occupies the third line of the header item. The String Value is always enclosed in quotation marks. If it is not used, the line consists of a null string, a pair of quotations marks with no space between them.

4.2 Header Items

There are four required header items. The other header items described in this document are standard optional header items. The defined standard items should be used by new programs using DIF. If it is absolutely necessary, a new header item may be defined to meet the needs of a particular program. For details, see the section on Defining New Header Items.

¹A token is an upper case string of alphanumeric characters. It is usually short, 32 characters or less. See the Definitions section for more information.

A program may ignore all header items until it finds the header item DATA, described below.

The following four header items are required:

4.2.1 The First Header Item

```
TABLE
0,version
"title"
```

The header item TABLE must be the first entry in the file. It identifies the file as a DIF file. The version number must be 1. The "title" is the title of the table and describes the data.

4.2.2 Vector Count

```
VECTORS
0,count
""
```

The header item VECTORS specifies the number of vectors in the file.

Note: This header item must appear before header items that refer to vector numbers. Otherwise, it can appear anywhere within the header section.

4.2.3 Tuple Count

```
TUPLES
0,count
""
```

The header item TUPLES specifies the length of each vector (the number of tuples). This can be used by a program to preallocate storage space for the data. This item may appear anywhere within the header section.

Note: Programs reading the data assume that the tuple count is correct. Some programs may be able to generate this information only after all data has been generated. These programs must reread the DIF file to count the tuples, and rewrite the TUPLES item with the correct count.

4.2.4 The Last Header Item

```
DATA
0,0
""
```

The header item DATA must be the last header item. It tells the program that all remaining data in the file are data values.

The following header items are optional. The programs that are known to use them are noted with the item. For detailed information on each program's specific use of the item, see the section below on Applications Programs.

4.2.5 Vector Label

```
LABEL
vector#,line#
"label"
```

The header item LABEL provides a label for the specified vector. The line number provides an option for labels that span more than one line, and can be ignored by a system that allows single line labels only. The values 0 and 1 are equivalent line numbers.

Note: Some programs do not use the LABEL field. If the first vector in a tuple contains string values, the first data value in the tuple may be treated as a label.

Used by the VisiPlottm and VisiTrend/VisiPlottm programs.

4.2.6 Vector Comment

```
COMMENT
vector#,line#
"comment"
```

The header item COMMENT is similar to LABEL. It provides an option to systems that allow an expanded description of a vector in addition to a label.

Used by the VisiPlot and VisiTrend/VisiPlot programs.

4.2.7 Field Size

```
SIZE
vector,bytes
""
```

The header item SIZE provides to programs such as data base systems the option to allocate fixed size fields for each value.

Because SIZE is an optional item, programs using SIZE must be able to read files produced by programs unable to generate SIZE information.

Used by the CCA/DMS program.

4.2.8 Periodicity

```
PERIODICITY
vector#,period
""
```

The header item PERIODICITY provides the option of specifying a period in a time series.

Used by the VisiPlot and VisiTrend/VisiPlot programs.

4.2.9 Major Start

```
MAJORSTART
vector#,start
""
```

The header item MAJORSTART specifies the first year of a time series.

Used by the VisiPlot and VisiTrend/VisiPlot programs.

4.2.10 Minor Start

```
MINORSTART
vector#,start
""
```

The header item MINORSTART specifies the first period of a time series.

Used by the VisiPlot and VisiTrend/VisiPlot programs.

4.2.11 True Length

```
TRUELENGTH
vector#,length
""
```

The header item TRUELENGTH specifies the portion of a vector that contains significant values.

Used by the VisiPlot and VisiTrend/VisiPlot programs.

4.2.12 Units

```
UNITS
vector#,0
"name"
```

The header item UNITS specifies the unit of measure for the values in the given vector. Name is the unit, for example meters or ft.

Used by the TK!Solver(tm) program.

4.2.13 Display Units

```
DISPLAYUNITS
vector#,0
"Name"
```

The header item DISPLAYUNITS specifies the unit in which the values in the given vector should be displayed. This unit may be different from the one in the UNITS field. The values in the given vector are always stored in the unit specified in the UNITS field, and the application program is responsible for making the value conversion between the UNITS and DISPLAYUNITS.

For example, a vector might be stored in km, but displayed in the program in miles. The UNITS field would be km, the DISPLAYUNITS field would be miles, and the values in the vector would be in km. Any program using the vector would have to define the conversion between km and miles to display the values in miles.

Used in the TK!Solver program.

4.3 Defining New Header Items

If there is no standard optional header item to fulfill the specific need of a subsystem, a new header item may be defined. Because the DIF format is

intended for common use, new optional header items should be standardized through the DIF Clearinghouse. They will then be added to this document.

To be accepted as standard items, new optional items must be consistent with existing conventions.

An optional item extends the format for a specific application. Any program reading the DIF file should be able to operate without optional items. If a reading program requires the information provided by an optional item, it should prompt the user to supply the missing information and not require the item itself.

5. The Data Section

The data section consists of a series of tuples. The Data Values within the tuples are organized in vector sequence.

Each Data Value represents one element of data in the file. The data may be either the actual data or one of the two Special Data Values that mark the beginning of a tuple (BOT) and the end of data (EOD) in the file.

Each Data Value consists of two lines. The first line consists of two fields containing numeric values, and the second line consists of one field containing a string value. The format is:

```
Type Indicator, Number Value  
String Value
```

5.1 The Type Indicator Field

The Type Indicator is an integer that tells the program what kind of data is represented by this value. There are currently three possible values.

- 1 The data is a Special Data Value, indicating either the beginning of a tuple or the end of data. The Number Value is zero (0) and the String Value is either BOT or EOD. See the description below of Special Data Values.
- 0 The data is numeric. The Number Value field contains the actual value and the String Value field contains a Value Indicator. See the descriptions below of the Number Value and String Value fields.
- 1 The data is a string value. The Number Value is zero (0) and the String Value field contains the actual string value.

5.2 The Number Value Field

When the Type Indicator is 0, the Number Value field contains the actual value. The value must be a decimal (base 10) number. It may be preceded by a sign (+ or -) and it may have a decimal point. It may be preceded or followed by one or more blanks. If the data value contains an exponent of a power of ten, the value is followed by the letter E and the signed or unsigned exponent power of ten.

Note: This is the only place where DIF allows a non-integer value. Some programs accept only integer values.

5.3 The String Value Field

The contents of the String Value field are dependent on the Type Indicator.

5.3.1 Special Data Value

If the Type Indicator is -1, the String Value is one of the two Special Data Values, BOT or EOD, and the Number Value is 0.

Each tuple begins with the Special Data Value BOT (Beginning of Tuple). If a program cannot generate a VECTORS header item before generating all data, it can use the Special Data Value BOT to determine the number of vectors in the file by counting the number of Data Values between BOTs when it rereads the file. A program can also verify its position in a file by using the BOT Special Data Value.

The Special Data Value EOD (End of Data) indicates the end of data in the file. The EOD occurs at the end of the last tuple in the file. If the program is unable to generate a TUPLES header item before generating all data, it can determine the number of tuples by counting the number of BOTs before the EOD when it rereads the file. A program can also use the EOD Special Data Value to detect the end of the file.

5.3.2 Numeric Value and Value Indicator

If the Type Indicator is 0, the data is numeric, and the String Value is one of the Value Indicators described below. The Value Indicator overrides the value.

A subsystem may define Value Indicators for its own needs. New Value Indicators should be registered with the DIF Clearinghouse.

The Value Indicators currently defined are:

V Value - This is the String Value most commonly used with a numeric value. The Number Value contains the actual value.

NA	Not Available - The value is marked as not available. The Number Value is 0.
ERROR	The value represents the result of an invalid calculation. The Number Value is 0.
TRUE	Logical value. The Number Value is 1.
FALSE	Logical value. The Number Value is 0.

The String Value can be ignored in favor of the Number Value, or all values with a Value Indicator other than V can be considered nonexistent. Quotes are not permitted around the Value Indicator.

5.3.3 String Value

If the Type Indicator is 1, the String Value is the actual character string. If the value is a token, the quotation marks are optional. However, if there is a beginning quotation mark, there must be a terminating quotation mark.

6. Definitions

This section defines specific characteristics of DIF.

Character Sets This document assumes use of the ASCII character set. The following characters are permitted:

```
!"#$%&'()*+,-./
0123456789:;<=>?
@ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]^_
`abcdefghijklmno
pqrstuvwxyz{|}~
```

(The first character in this list is a space.)

There are 95 printable characters, including the space. If the host computer has more than 95 characters, the additional characters must be mapped into the 95 ASCII characters to transfer data to another machine.

Some computers permit only 64 characters. When data is transferred to these machines, lower case characters and the characters `{}` are mapped into their corresponding upper case characters. If these transformations affect the integrity of the data, associated documentation should specify the effect.

Transfers between character sets should be transparent to most users. To assure compatibility, strings should not contain nonprinting characters.

EBCDIC EBCDIC is a binary representation of characters and is used primarily for large IBM computers. An awareness of the representation used is not essential, but if files are transferred between machines they must be converted to the standard representation of the host machine.

Because EBCDIC defines more than the 95 standard printable characters, users should avoid the additional characters when preparing data files on an EBCDIC machine.

String Length Some programs place a length limit on strings that they read. This results in the truncation of long string values. Some systems also limit the length of lines in a data file. Programs should support a minimal string length of 64 characters, but longer ones are preferable.

String Delimiters

Some systems delimit strings with apostrophes instead of quotation marks. When files are transferred to or from these systems, appropriate changes must be made.

Tokens A token is a string consisting of upper case alphanumeric characters. It should have a maximum length of 32 characters. Commonly, tokens may or may not be contained within quotation marks; however, a token that is a required string, such as a header item topic, must be represented without quotation marks.

Floating Point Numbers

A floating point number consists of an optional sign and a series of digits followed by an optional decimal point. The number may be followed by the letter E (exponent) and a signed decimal exponent.

Note: Some systems generate the letter D to indicate a double precision floating point number. This is not standard, but it can be read by compatible programs within a single system. When transferring data to other computers, the D must be converted to an E.

7. Applications Programs

This section records the specific use of DIF by applications programs that support it. Programmers who intend to interface with any of these programs should note the specifics listed here. Standardized optional items used by these applications are listed in the general section on Optional Header Items. However, if a program uses a header item that varies significantly from the

conventions, it is mentioned only in this section. The accuracy of this information is not guaranteed.

7.1 The CCA/DMS Program

Published and distributed by VisiCorp.

Uses: SIZE

7.2 The TREND-SPOTTER(R) Program

Published and distributed by Software Resources, Inc.

The TREND-SPOTTER program requires that the DIF file contain either only one tuple or only one vector.

7.3 The VisiCalc(R) Program

Published and distributed by VisiCorp.

Program created and written by Software Arts(tm).

The VisiCalc program does not generate the LABEL items. Some programs interfacing to the VisiCalc program have adopted the convention of examining the first Data Value in a tuple, and, if it is a string value, treating it as a label.

7.4 TK!Solver

Published and distributed by Software Arts, Inc.

Uses: UNITS, DISPLAYUNITS

7.5 the VisiPlot and VisiTrend/VisiPlot Programs

Published and distributed by VisiCorp.

Early versions of the VisiPlot and VisiTrend/VisiPlot programs used the Number Value and String Value incorrectly, storing the Number Value in the String Value field. Programs exchanging data with these versions should check the String Value. If it is not null, the string must be converted and the Number Value computed.

Uses: LABEL, COMMENT, MAJORSTART, MINORSTART, PERIODICITY, TRUELENGTH

I. Sample DIF File

This is an example of a DIF data file. The data in the file is represented by the table below.

PROFIT REPORT

YEAR	SALES	COST	PROFIT
1980	100	90	10
1981	110	101	9
1982	121	110	11

The Data File

```

TABLE          ----->
0,1            >
"PROFIT REPORT" >
VECTORS -----> Header >
0,4           >
""           -----> Item >
TUPLES       >
0,3          >
""          >
LABEL       >
1,0        >
"YEAR"     > Header
LABEL     >
2,0      >
"SALES"  >
LABEL   >
3,0    >
"COST" >
LABEL  >
4,0   >
"PROFIT" >
DATA   >
0,0   >
""    ----->

-1,0 ----->
BOT  >
0,1980 >
V    >
0,100 >
V    >
0,90 >
V    >
0,10 >
V    >
-1,0 >
BOT  >

0,1981 -----> >
V    > > Data
0,110 > >
V    > > Part
0,101 ---> Data > Tuple >
V    ---> Value > >
0,9  > >
V    > >
-1,0 > >
BOT  -----> >
0,1982 > >
V    > >
0,121 > >
V    > >
0,110 > >
V    > >
0,11  > >
V    > >
-1,0  > >
EOD   ----->

```


II. Sample BASIC program that writes a DIF file

This program enters student records into a file by prompting the user for a student's name and test scores and copying the information into a DIF file.

```

100 REM - THIS PROGRAM CREATES A DIF FILE CONTAINING THE
110 REM - NAME AND TEST SCORES OF A GIVEN NUMBER OF STUDENTS.
120 REM - IT PROMPTS FOR A FILE NAME, THE TOTAL NUMBER OF
130 REM - STUDENTS, AND THE NUMBER OF TEST SCORES FOR
140 REM - EACH STUDENT. IT THEN PROMPTS FOR A STUDENT'S
150 REM - NAME AND TEST SCORES, AND WRITES THEM TO THE
160 REM - FILE AS A TUPLE.

1000 PRINT "OUTPUT FILE NAME:"; :REM - GET FILE NAME.
1010 INPUT F$
1020 OPEN "O",1,F$ :REM - OPEN FILE FOR OUTPUT.
1030 PRINT "NUMBER OF STUDENTS:";
1035 :REM - PROMPT FOR NUMBER OF
1040 INPUT NT :REM - TUPLES.
1050 PRINT "NUMBER OF TEST SCORES PER STUDENT:";
1060 INPUT NV :REM - NUMBER OF VECTORS IS
1070 NV = NV + 1 :REM - NUMBER OF SCORES + 1.
1080 GOSUB 3000 :REM - USE SUBROUTINE TO
1090 :REM - OUTPUT DIF HEADER.

2000 FOR I = 1 TO NT :REM - OUTPUT A TUPLE FOR
2010 :REM - EACH STUDENT.
2020 T = -1: V = 0: S$ = "BOT"
2025 :REM - OUTPUT BOT SPECIAL
2030 GOSUB 4000 :REM - DATA VALUE.
2040 PRINT "NAME OF STUDENT #";I;
2050 INPUT S$ :REM - GET NAME OF THIS STUDENT.
2060 T = 1: V = 0 :REM - OUTPUT AS STRING DATA
2070 GOSUB 4000 :REM - VALUE.
2080 FOR J = 1 TO NV-1 :REM - PROCESS EACH SCORE.
2090 PRINT "SCORE #";J;
2100 INPUT V :REM - GET SCORE.
2110 T = 0: S$ = "V" :REM - OUTPUT SCORE AS A DATA
2120 GOSUB 4000 :REM - VALUE.
2130 NEXT J
2140 NEXT I
2150 T = -1: V = 0: S$ = "EOD" :REM - OUTPUT EOD SPECIAL DATA
2160 GOSUB 4000 :REM - VALUE.
2170 CLOSE 1 :REM - CLOSE THE OUTPUT FILE.
2180 STOP :REM - DONE.

```

```
3000                                     :REM - ROUTINE TO OUTPUT HEADER.
3010 PRINT#1,"TABLE":PRINT#1,"0,1":GOSUB 3500
3020 PRINT#1,"TUPLES":PRINT#1,"0,";NT:GOSUB 3500
3030 PRINT#1,"VECTORS":PRINT#1,"0,";NV:GOSUB 3500
3040 PRINT#1,"DATA":PRINT#1,"0,0":GOSUB 3500
3050 RETURN
3500                                     :REM - ROUTINE TO OUTPUT A
3510                                     :REM - NULL STRING ("").
3520 PRINT#1,CHR$(34);CHR$(34) :REM - PRINT 2 QUOTATION MARKS.
3530 RETURN

4000                                     :REM - ROUTINE TO OUTPUT A DATA
4010                                     :REM - VALUE. T IS THE TYPE
4020                                     :REM - INDICATOR, V IS THE
4030                                     :REM - NUMBER VALUE, AND S$
4040                                     :REM - IS THE STRING VALUE.
4050 PRINT#1,T;"",";V
4060 PRINT#1,S$
4070 RETURN
5000 END
```

III. Sample BASIC program that reads a DIF file

This program uses the output DIF file from the previous sample program to calculate an average score and letter grade for each student.

```

100 :REM - THIS PROGRAM READS A DIF FILE CONTAINING THE
110 :REM - TEST SCORES OF A GROUP OF STUDENTS, CALCULATES
120 :REM - AN AVERAGE SCORE FOR EACH STUDENT, MATCHES THE
130 :REM - AVERAGE TO A LETTER GRADE, AND PRINTS THE
140 :REM - STUDENT'S NAME, AVERAGE, AND LETTER GRADE.

500 DIM T(100)           :REM - MAXIMUM OF 100 VECTORS.
510 DIM V(100)           :REM - T IS THE TYPE INDICATOR, V IS
520 DIM V$(100)          :REM - THE NUMBER VALUE, AND V$ IS
530                     :REM - THE STRING VALUE OF EACH DATA
535                     :REM - VALUE.
540 GOSUB 5000           :REM - INITIALIZATION SUBROUTINE.
550 GOSUB 6000           :REM - SUBROUTINE TO READ HEADER.
560 FOR I = 1 TO NT      :REM - FOR EACH TUPLE,
570   GOSUB 7000         :REM - GET ALL VECTOR ELEMENTS IN
575                     :REM - TUPLE.
580   M=0                :REM - M IS THE SUM OF THE SCORES.
590   FOR J = 1 TO NV    :REM - FOR EACH VECTOR VALUE,
600     IF T(J)=1 THEN PRINT V$(J) :REM - PRINT NAME.
610     IF T(J)=0 THEN M = M+V(J)  :REM - ADD SCORES.
620   NEXT J
630   M = M/(NV-1): PRINT M :REM - PRINT STUDENT'S AVERAGE
640   IF M<=50 THEN PRINT "THIS STUDENT'S FINAL GRADE IS F"
650   IF M<=70 AND M>50 THEN PRINT "THIS STUDENT'S FINAL GRADE IS D"
660   IF M<=85 AND M>70 THEN PRINT "THIS STUDENT'S FINAL GRADE IS C"
670   IF M<=94 AND M>85 THEN PRINT "THIS STUDENT'S FINAL GRADE IS B"
680   IF M>94 THEN PRINT "THIS STUDENT'S FINAL GRADE IS A"
690 NEXT I
700 CLOSE 2
710 PRINT "FINISHED CALCULATING GRADES"
720 STOP

5000                     :REM - INITIALIZATION CODE.
5010 PRINT "FILE NAME";
5020 INPUT F$
5030 OPEN "I",2,F$       :REM - OPEN FILE FOR INPUT.
5040 NV = 0              :REM - INITIAL VECTOR COUNT.
5050 NT = 0              :REM - INITIAL TUPLE COUNT.
5060 RETURN

```

```
6000 :REM - READ HEADER. GET NUMBER OF VECTORS AND TUPLES.
6010 INPUT#2,T$ :REM - GET TOPIC.
6020 INPUT#2,S,N :REM - GET VECTOR NUMBER AND VALUE.
6030 INPUT#2,S$ :REM - GET STRING VALUE.
6040 IF T$="VECTORS" THEN 6500:REM - CHECK FOR KNOWN HEADER
6050 IF T$="TUPLES" THEN 6600 :REM - ITEMS.
6060 IF T$="DATA" THEN RETURN
6065 :REM - "DATA" ENDS HEADER.
6070 GOTO 6010 :REM - IGNORE UNKNOWN ITEMS.
6500 NV = N :REM - NUMBER OF VECTORS.
6510 IF NV<=100 THEN 6010 :REM - CHECK FOR 100 OR LESS VECTORS.
6520 PRINT "TOO MANY VECTORS. PROGRAM CAPACITY 100 VECTORS."
6530 CLOSE 2
6540 STOP
6600 NT = N :REM - NUMBER OF TUPLES.
6610 GOTO 6010 :REM - GET NEXT HEADER ITEM.

7000 :REM - SUBROUTINE TO GET ALL VECTOR ELEMENTS IN A TUPLE.
7010 GOSUB 8000 :REM - GET NEXT DATA VALUE.
7020 IF T1<>-1 THEN 9000 :REM - MUST BE BOT, ELSE ERROR
7030 IF S$<>"BOT" THEN 9000
7040 FOR K = 1 TO NV :REM - GET EACH DATA VALUE.
7050 GOSUB 8000
7060 IF T1>1 THEN 9000
7070 T(K) = T1 :REM - SAVE TYPE INDICATOR.
7080 V(K) = V1 :REM - SAVE NUMBER VALUE.
7090 V$(K) = S$ :REM - SAVE STRING VALUE.
7100 NEXT K
7110 RETURN

8000 :REM - SUBROUTINE TO GET NEXT DATA VALUE.
8010 INPUT#2,T1,V1 :REM - GET TYPE INDICATOR, NUMERIC
8020 INPUT#2,S$ :REM - VALUE, AND STRING VALUE.
8030 RETURN

9000 :REM - ERROR ROUTINE
9010 PRINT "ERROR IN FILE FORMAT"
9020 CLOSE 2 :REM - END PROGRAM
9030 STOP
9040 END
```

IV. Sample Pascal program using a DIF file

This program is a Pascal program that reads data from a DIF file into an array and displays the results on the terminal.

```
{ This is a simple program which reads DIF file data into an array and
  displays the results on the terminal. It makes use of a procedure
  called "get_dif_array" which handles only numeric data. It is written
  for Apple Pascal 1.1 and may require modification to run on other systems. }
```

```
program dif_read;

const
  max_vector = 10;           { maximum number of vectors }
  max_tuple  = 10;           { maximum number of tuples }

type
  vector_index = 0..max_vector;
  tuple_index  = 0..max_tuple;
  dif_array    = array[1..max_vector, 1..max_tuple] of real;

var
  in_file      : text;           num_vectors : vector_index;
  fname       : string[15];     num_tuples  : tuple_index;
  matrix      : dif_array;      code, i, j  : integer;
```

```
{ "Get_dif_array" reads a DIF file and returns the file data (currently
only numeric) in an array. Also returns number of vectors and tuples
-- these must be specified in file header -- and an error code. }
```

```
procedure get_dif_array (var dif_file: text; var real_array: dif_array;
                        var nvectors: vector_index; var ntuples: tuple_index;
                        var return_code: integer);
```

```
const
    { currently defined data types }
    special = -1; numeric = 0; char_string = 1; other = 2;
```

```
type
    header_item = record
        topic      : string;
        vector_num : vector_index;
        value      : integer;
        string_value : string
    end;
    data_value = record
        kind      : -1..2; { currently defined data types}
        number_value : real;
        string_value : string
    end;
```

```
var
    hdr_item      : header_item;
    data_val      : data_value;
    tuple, vector : integer;
```

```
{ "Read_integer" reads an integer terminated with a comma. The routine
is required because this Pascal dialect's "read" procedure recognizes
only <space>, eoln and eof as delimiters of integer values. }
```

```
procedure read_integer (var number: integer);
var
    sign, magnitude : integer;
    ch : char;
begin
    sign := 1; magnitude := 0;           { initialize }
    read (difile, ch);                  { get 1st character }
    while ch <> ',' do                    { comma is delimiter }
    begin
        case ch of
            '-' : sign := -1;
            '0','1','2','3','4','5','6','7','8','9'
                : magnitude := magnitude * 10 + ord(ch) - ord('0')
        end; { case }
        read (difile, ch)                 { get next character }
    end;
    number := sign * magnitude           { return result }
end; { read_integer }
```

```

{ "Read_string" deletes leading and trailing blanks and strips the
  quotes from quoted strings. }

procedure read_string (var str: string);
begin
  readln (difile, str);
  while str[1] = ' ' do { leading blanks }
    delete (str, 1, 1);
  if str[1] = '"' { strip quotes }
  then begin
    delete (str, 1, 1);
    delete (str, pos('"', str),
      length(str) - pos('"', str) + 1)
  end
  else if pos(' ', str) > 0 { trailing blanks }
  then delete (str, pos(' ', str),
    length(str) - pos(' ', str) + 1)
end; { read_string }

procedure read_header_item (var item: header_item);
begin
  read_string (item.topic); { get topic }
  read_integer (item.vector_num); { get vector number }
  readln (difile, item.value); { get value }
  read_string (item.string_value) { get string value }
end; { read_header_item }

procedure read_data_value (var value: data_value);
begin
  read_integer (value.kind); { get data type }
  readln (difile, value.number_value); { get number value }
  read_string (value.string_value) { get string value }
end; { read_data_value }

begin { get_dif_array }
  return_code := 0; { assume no problems }
  nvectors := 0; ntuples := 0; { initialize }
  repeat { read header }
    read_header_item (hdr_item);
    if hdr_item.topic = 'VECTORS'
    then nvectors := hdr_item.value { vector count }
    else if hdr_item.topic = 'TUPLES'
    then ntuples := hdr_item.value { tuple count }
  until hdr_item.topic = 'DATA';

```

```

if (nvectors = 0) or (ntuples = 0)           { check counts }
then return_code := 1
else begin
  for tuple := 1 to ntuples do             { read data }
  begin
    read_data_value (data_val);           { BOT }
    for vector := 1 to nvectors do
    begin
      read_data_value (data_val);
      if data_val.kind = numeric
      then real_array[vector, tuple] :=
          data_val.number_value
    end
  end;
  read_data_value (data_val);             { EOD }
  if (data_val.kind <> special) or
  (data_val.string_value <> 'EOD')
  then return_code := 2
end

end; { get_dif_array }

begin { dif_read }

  writeln;                                { get DIF file name }
  write ('DIF file name: ');
  readln (fname);
  reset (in_file, fname);                 { open and point to BOF }

  get_dif_array (in_file, matrix, num_vectors, num_tuples, code);

  close (in_file);                        { close DIF file }
  case code of                             { display results }
  0: begin
    writeln;
    writeln ('"', fname:15, '"', ' contains ', num_vectors:3,
              ' vectors and ', num_tuples:3, ' tuples. ');
    writeln ('The data values follow in tuple order: ');
    writeln;
    for i := 1 to num_tuples do
    begin
      for j := 1 to num_vectors do
      write (matrix[j, i]:10:2);
      writeln
    end;
    writeln
  end;
  1: writeln ('Error. Tuple or vector count not found. ');
  2: writeln ('Error. Data not properly terminated. ');
end { case }

end. { dif_read }

```


Appendix B

References to Published Listings of BASIC and Pascal Programs That Process DIF Files

The items below have been extracted from the Bibliography on the DIF Format in this book. These may be of interest to those writing programs to create and manipulate DIF files.

"DIF Technical Specification." Wellesley, MA: *The DIF Clearinghouse*, 1983. Version DIF-0283, reprinted as Appendix A of this book, contains BASIC and Pascal programs to process DIF files.

Fylstra, Dan and Bill Kling. *VisiCalc User's Guide for the Apple II and II Plus, 16 Sector Version*. San Jose, CA: VisiCorp (Personal Software), 1981. Some editions of this product documentation include an appendix with programs in BASIC to dump DIF files record by record as stored, to print a worksheet from the data of a DIF file, and to create a DIF file.

Hergert, Douglas. *Mastering VisiCalc*. Berkeley, CA: SYBEX, 1983. Includes three chapters on DIF files, including numerous examples of BASIC programs to process DIF files.

Kalish, Candace E. and Malinda F. Mayer. "DIF: A Format for Data Exchange Between Applications Programs." *BYTE* 6, no. 11 (November 1981): 174, 176, 178, 180, 182, 186, 188, 190, 192, 194, 196, 198, 200, 202, 204, 206. Presents DIF in detail including BASIC and Pascal programs to create and read DIF files.

Miller, David. *Apple Files*. Reston, VA: Reston Publishing Company, Inc., 1982. Contains a full chapter explaining DIF files, and includes BASIC programs to process them.

"Programmer's Guide to the Data Interchange Format." Wellesley, MA: *The DIF Clearinghouse*, 1980. Includes BASIC programs to create and list a DIF file. This document was included with some versions of the documentation for VisiCalc.

Wolverton, Van *VisiCalc User's Guide for the IBM Personal Computer*. San Jose, CA: VisiCorp (Personal Software), 1981. Includes an appendix on DIF which contains BASIC programs that print a DIF file as stored, print a worksheet from a DIF file, and create a DIF file.

Appendix C

Products That Use The DIF Format

This Appendix lists commercial software that has the capability of using data files in the DIF format. Below are some general notes on the information.

- These are not intended as software product reviews, nor as endorsements of the application packages.
- The Bibliography on the DIF Format contains references to additional sources of information regarding some of these products.
- These products may be existing or announced (but not yet on the market).
- Product cost and hardware availability have not been listed since these are subject to change.
- A product that is “compatible with VisiCalc” does not necessarily use the DIF format since VisiCalc can save files in several formats. All products listed are believed to use DIF files.
- This information has been compiled from advertisements, press releases, product announcements, magazine/newsletter articles, telephone calls or letters to software companies, product documentation, and the *DIF Program List* from the DIF Clearinghouse (see the Bibliography on the DIF Format for a full reference).
- Trademarks are listed in the front matter of this book.
- The column headed Description/Notes contains brief comments about the product and is not intended to be a full description of product capabilities.

- The addresses listed are believed to be the major source of information about the product. In some cases, they are not the creators of the program.
- Some products may be general purpose file handlers (for example, communications programs) that handle DIF files in the same way they handle any file. They are included if their product material specifically lists DIF files.
- Because the products listed use DIF files does not mean that they use them properly within the DIF specification.

ALPHABETICAL LISTING OF PRODUCTS

Product	Vendor	Description/Notes
1-2-3	Lotus Development Corp. 55 Wheeler Street Cambridge, MA 02138 (617) 492-7171	Incorporates spreadsheet, data base, graphics, and text editing into a single product.
<i>Accounting Plus II</i>		Refer to <i>Data Plus</i> in this listing.
<i>AEN Grading System and Update</i>		Refer to <i>AEN Grading System Utility</i> in this listing.
<i>AEN Grading System Utility</i>	Apple Educators' Newsletter 9525 Lucerne Street Ventura, CA 93004	Utility program that allows for file transfer in support of the <i>AEN (Apple Educators' Newsletter) Grading System</i> for classroom teachers.
<i>Apple Business Graphics</i>	Apple Computer Inc. 20525 Mariani Avenue Cupertino, CA 95014 (800) 538-9696 (800) 662-9238 (CA)	Graphics.
<i>Bridge</i>	Sun Microsystems, Inc. P.O. Box 1388 Ft. Lauderdale, FL 33302 (305) 368-8221	Creates DIF files from PFS data files.

<i>CALCPACK</i>	Jini Micro-Systems, Inc. Box 274E Kingsbridge Station Riverdale, NY 10463 (212) 796-6200	Interface program for the <i>JINSAM</i> data manager software.
<i>CCA/DMS DIF Interface</i>	F/S Associates 664 18th Street Manhattan Beach, CA 90266	DIF interface for CCA/DMS.
<i>CHARTDIF and PLOTDIF</i>	The Consultant 414 Can-Data Mt. Prospect, IL 60056	Charts DIF data.
<i>CHARTMAN</i>	Graphic Software, Inc. P.O. Box 367 MS-2 Kenmore Station Boston, MA 02215 (617) 491-2434	Graphics.
<i>Chart-Master</i>	Decision Resources 44 White Birch Road Weston, CT 06883 (203) 222-1974	Graphics.
<i>CompuServe</i>	CompuServe 5000 Arlington Centre Blvd. P.O. Box 20212 Columbus, OH 43220	Listed here because it is used in a case study. Also refer to Compustat Retrieval/DIF and QTRAN in this listing.
<i>Compustat Retrieval System/DIF</i>	CompuServe 5000 Arlington Centre Blvd. P.O. Box 20212 Columbus, OH 43220	Extracts COMPUSTAT II or Value Line data and writes it to a DIF file.
<i>Data Plus</i>	Software Dimensions 6371 Auburn Blvd. Citrus Heights, CA 95610 (916) 722-8000	Converts data from <i>Accounting Plus II</i> into the DIF format.

<i>Data Sorter</i>	InterCalc P.O. Box 254 Scarsdale, NY 10583	Sorts VisiCalc print-to-disk files, creating DIF files from them.
<i>Data-Trans</i>	Abt Microcomputer Software Abt Associates, Inc. 55 Wheeler Street Cambridge, MA 02138 (617) 492-7100	Communications software that allows file transfer and conversion, including DIF, between microcomputers.
<i>DB Master</i>		Refer to <i>DB Master Utility Pak #1</i> in this listing.
<i>DB Master Utility Pak #1</i>	Stoneware Micro-computer Products 50 Belvedere Street San Rafael, CA 94901 (415) 454-6500	Includes the ability to exchange data between <i>DB Master</i> and DIF files.
<i>DIFmaster</i>	Starside Engineering P.O. Box 18306 Rochester, NY 14618 (716) 461-1027	Three-dimensional business graphics.
<i>DIFPLOT</i>	Strobe Inc. 897-5A Independence Avenue Mountain View, CA 94043 (415) 969-5130	Reads and graphically displays DIF files. For use in conjunction with the <i>Strobe 100 Graphics Plotter and Software</i> .
<i>The Draftsman</i>	Starware 1701 K Street, N.W. Washington, DC 20006	Graphics.
<i>The Executive Secretary</i>	Sof/Sys 4306 Upton Avenue, S. Minneapolis, MN 55410	Word processing package that allows for integration of DIF files.
<i>FAST GRAPHS</i>	Innovative Software 9300 West 110th St., Suite 380 Overland Park, KS 66210 (913) 383-1089	"Graphic report" program.

<i>FileMaster 2.0</i>	NF SYSTEMS, Ltd. P.O. Box 76363 Atlanta, GA 30358 (404) 252-3302	File management system.
<i>Formstemp for Business</i>		Refer to <i>SpreadTemps</i> in this listing.
<i>GIRAPH</i>	Data Display 171 West 4th Street New York, NY 10014 (212) 924-8167	Graphics.
<i>Graph</i>		Refer to <i>PFS:Graph</i> in this listing.
<i>Graph 'n' Calc</i>	Desktop Computer Software Santa Cruz, CA	Business graphics and calculation.
<i>The Graphics Generator</i>	Robert J. Brady Co. Bowie, MD 20715 (301) 262-6300	Business and technical graphics.
<i>GRAPHMAGIC</i>	International Software Marketing 120 East Washington St. Suite 421, University Building Syracuse, NY 13202 (315) 474-3400	Creates visual diagrams from mathematical data.
<i>GraphPower</i>	Ferox Microsystems, Inc. 1701 N. Ft. Myer Drive, Suite 611 Arlington, VA 22209	Graphics.
<i>HealthTemp</i>		Refer to <i>SpreadTemps</i> in this listing.
<i>INI Client Write-Up System</i>	INI Inc. 4013 Chestnut Street Philadelphia, PA 19104	General ledger and report generator.
<i>Investor's Interface</i>	MarketWare P.O. Box 34647 Richmond, VA 23234 (804) 276-8577	Captures Dow Jones News/Retrieval stock data and converts it to DIF format.

<i>JINSAM</i>		Refer to <i>CALCPACK</i> in this listing.
<i>List Handler</i>	Silicon Valley Systems 1625 El Camino Real Belmont, CA 94002 (415) 593-4344	Data base.
<i>LoadCalc</i>	Cypher 121 Second Street San Francisco, CA 94105 (415) 974-5297	Utility program that converts text files into DIF files.
<i>MAGICALC</i>	Artsci, Inc. 5547 Satsuma Avenue North Hollywood, CA 91601 (213) 985-2922	Spreadsheet
<i>MAINLINE</i>	Gregg Corporation 100 Fifth Avenue Waltham, MA 02254 (617) 890-7227	Mainframe access program that passes data from mainframe data bases to microcomputers using the DIF file format.
<i>MDBS.QRS (Query System/Report Writer)</i>	Micro Data Base Systems, Inc. P.O. Box 248 Lafayette, IN 47902	Data base manager.
<i>MergeCalc</i>	Cypher 121 Second Street San Francisco, CA 94105 (415) 974-5297	Allows for consolidation, merging, or manipulation of multiple VisiCalc models including data in DIF formats.
<i>MICROMAGIC</i>	I.P. Sharp Associates 1200 First Federal Plaza Rochester, NY 14614 (716) 546-7270	Creates DIF files from I.P. Sharp's on-line databases.
<i>Micrograph by 2Y's</i>	2Y's Associates Ltd. Box 6733, Station "J" Ottawa, Ontario, Canada K2A 3Z4	Graphics and trend analysis.
<i>Pascal DOS 3.3 Transfer Utility</i>	Orange Software 293 Manley Heights Orange, CT 06477	File transfer including DIF.

<i>PEAR PLUS</i>	Remote Computing Corp. 1044 Northern Blvd. Roslyn, NY 11576 (800) 645-3120 (212) 895-3810	Creates DIF files from PEAR (Portfolio Evaluation and Reporting System); includes data analysis.
<i>The Personal Investor</i>	PBL Corporation P.O. Box 559 Wayzata, MN 55391 (612) 471-7644	Investment analysis.
<i>PFS:Graph</i>	Software Publishing Corporation 1901 Landings Drive Mountain View, CA 94043 (415) 962-8910	Graphics.
<i>QTRAN</i>	CompuServe 5000 Arlington Centre Blvd. P.O. Box 20212 Columbus, OH 43220	Converts DIF files into a format readable by CompuServe planning languages.
<i>SMART</i>	Software Resources, Inc. 186 Alewife Brook Parkway Cambridge, MA 02138	Investment tools.
<i>Smartware</i>	Cypher 121 Second Street San Francisco, CA 94105 (415) 974-5297	The name used by this company for a series of products from a variety of software producers that work together by using the DIF file format.
<i>speedSTAT</i>	SoftCorp International 229 Huber Village Blvd. P.O. Box 29765 Columbus, OH 43229 (800) 543-1350 (513) 891-5044 Ohio	A family of statistical software packages.

<i>SpreadTemps</i>	SpreadSoft P.O. Box 192 Clinton, MD 20735 (301) 856-1180	Application templates including <i>FormsTemp for Business</i> (business forms) and <i>HealthTemp</i> (health, nutrition, etc.) that use DIF files.
<i>Statistics with DAISY</i>	Rainbow Computing, Inc. 19517 Business Center Dr. Northridge, CA 91324 (800) 423-5441 (213) 349-0300	Statistical package.
<i>Strobe 100 Graphics Plotter and Software</i>		Refer to <i>DIFPLOT</i> in this listing.
<i>SuperCalc</i>		Refer to <i>SuperData Interchange</i> in this listing.
<i>SuperData Interchange</i>	Sorcim 2310 Lundy Avenue San Jose, CA 95131 (408) 942-1727	File transfer between <i>SuperCalc</i> and other products, including the DIF format.
<i>Tax Helper</i>	Decision Support Software 1438 Ironwood Drive McLean, VA 22101 (800) 368-2022 (703) 241-8316	Federal tax model including all forms. Uses DIF files. Also includes the ability to use files from data of <i>The Accountant Finance Database System</i> .
<i>TDM/TPG (The Data Machine/The Program Generator)</i>	Pascal Systems, Inc. 830 Menlo Avenue, Suite 109 Menlo Park, CA 94025 (415) 321-0761	Data base management software with DIF interface capabilities.
<i>TextMaster</i>	N.F. SYSTEMS, Inc. P.O. Box 76363 Atlanta, GA 30358 (404) 252-3302	Text processor for form letters, reports, etc.
<i>T.I.M. III</i>	Innovative Software, Inc. 9300 W. 110th St. Suite 380 Overland Park, KS 66210 (913) 383-1089	Data management.

<i>TK!Solver</i>	Software Arts, Inc. 27 Mica Lane Wellesley, MA 02181 (617) 237-4000	Interactive tool for solving engineering and business problems without programming.
<i>Transfer ///</i>	Mind Systems Corporation P.O. Box 506 Northampton, MA 01061	File transfer between Apple II and Apple ///.
<i>Trend-Spotter</i>	Software Resources, Inc. 186 Alewife Brook Parkway, Suite 310 Cambridge, MA 02138	Produces trend graphics from data including data in the DIF format.
<i>ULTRA PLOT/D.I.F./ DataGraph</i>	Avant-Garde Creations P.O. Box 30160 Eugene, OR 97403 (503) 345-3043	Package composed of <i>ULTRA PLOT</i> and <i>ULTRA PLOT/D.I.F./Interface</i> that allows plotting of data in DIF files.
<i>ULTRA PLOT/D.I.F./ Interface</i>		Refer to <i>ULTRA PLOT/D.I.F./DataGraph</i> in this listing.
<i>Utility Pak #1</i>		Refer to <i>DB Master Utility Pak #1</i> in this listing.
<i>VC-Loader</i>	Micro Decision Systems P.O. Box 1392 Pittsburgh, PA 15219 (412) 276-2387	Converts text files to DIF format. Includes ability to add LABELS to DIF files generated.
<i>VC-Manager</i>	Micro Decision Systems P.O. Box 1392 Pittsburgh, PA 15219 (412) 276-2387	Consolidates <i>VisiCalc</i> models including DIF format. Includes feature to add required Header Section items to DIF files for use in <i>DB Master</i> . Graph plotter.

<i>VERSAPLOT</i>	Spectra Soft 350 North Lantana, Suite 775 P.O. Box 3000 Camarillo, CA 93011 (805) 987-6602	Graph plotter.
<i>VisiBridge/RPT</i>	SOLUTIONS, INC. P.O. Box 989 Montpelier, VT 05602	Report formatter for VisiCalc data in the DIF format.
<i>VisiCalc</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Spreadsheet.
<i>VisiCalc Advanced Version</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Spreadsheet.
<i>VisiCalc Business Forecasting Model</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Interrelated business forecasting templates for use with VisiCalc.
<i>VisiCalc Utilities</i>	Robert H. Flast & Co. 6 Peter Cooper Road New York, NY 10010	Includes a DIF sort of VisiCalc data.
<i>VisiFile</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Data base.
<i>VisiPlot</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Graphics.
<i>VisiSchedule</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Project planner.

<i>VisiTerm</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Communication software for file transfer.
<i>VisiTrend/Plot</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Combines graphics with trend analysis in one software package.
<i>VisiWord</i>	VisiCorp 2895 Zanker Road San Jose, CA 95134 (408) 946-9000	Word processing. Listed here because it is used in the case study; however, it does not read or write DIF files, other than as text files.
<i>VIZ.A.CON</i>	Abacus Associates Suite 240 6565 West Loop South Bellaire, TX 77401	VisiCalc consolidator.
<i>WordStar Connection</i>	Sofstar 13935 U.S. 1 June Square Juno Beach, FL 33408 (305) 627-5511	Converts a DIF file to <i>WordStar</i> word processing format.

Bibliography on the DIF Format

- "AEN Grading System Utility." *Apple Educators' Newsletter* 3, no. 16: 3. Describes an interface between this package and VisiCalc through the use of DIF files.
- Ahl, David H. "Charts and Graphs from an Apple Computer." *Creative Computing* 8, no. 11 (November 1982): 55-56, 58-59. This article is a product review of PFS:Graph from Software Publishing Corp. It discusses the ability to use DIF files as input.
- Ahl, David H. "Computer Tool Kit for Solving Business Problems." *Small Business Computers* 6, no. 7 (January/February 1983): 30-33. A review of TK!Solver, including mention of its ability to read and write DIF files.
- Ahl, David H. "Evaluation of VisiTrend and VisiPlot from Personal Software." *Creative Computing* 7, no. 12 (December 1981): 98, 100, 102, 104; and a related letter to the editor, "GRM of an Idea" from Reed Jenney 8, no. 3 (March, 1982): 8. Discussion of these products and their interface with DIF.
- Ahl, David H. "TK!Solver from Software Arts." *Creative Computing* 8, no. 11 (November 1982): 33, 35, 38, 43-44, 46. A product review of TK!Solver, including its use of DIF files.
- "All-in-One Package Provides Useful Thinking, Visual Tool." *Business Computer Systems* 2, no. 1 (January 1983): 141. Description of 1-2-3 including brief mention of its DIF capability.
- "Apple-Based Visi-Schedule Joins VisiCorp. Line." *Computer Business News* 5, no. 7 (February 15, 1982): 16. Discusses the interface of this product with others.
- "At 3.3, VisiCalc Spawns a Family." *Softalk* 1, no. 10 (June 1981): 31-32. Describes related "Visi" products and their interrelationship through DIF.

- Bayle, Elisabeth. "Picture This - And Do It Yourself." *Personal Computing* 6, no. 8 (August 1982): 50-54, 58, 64, 150. Discusses the use of graphics and slide-show packages by managers, including several that can accept the DIF file format.
- Bayer, Barry D. "Visulating: Eliminating Drugery." *Desktop Computing* 2, no. 10 (October 1982): 14-17, 19. Contains an example of DIF files used within a VisiCalc context in creating real estate construction contractors' and mechanics' lien affidavits. And "Visulating: Another Look at IRAs," 2, no. 11 (November 1982): 12-17, which includes an example using DIF in IRA evaluation. And "Visulating: Real Estate Analysis," 3, no. 2 (February 1983): 26-31. Includes the use of DIF files in computing golf handicaps.
- Beil, Donald H. *The VisiCalc Book, Apple Edition*. Reston, VA: Reston Publishing Company, 1982. Includes brief mention of DIF. Also available in editions for the ATARI and IBM Personal Computer.
- "Business Software Update: Financial Planning, Text Processing and Accounting." *Personal Computing* 6, no. 5 (May 1982): 190, 192. Describes the GRAPH product from Software Publishing Corp., and its ability to use VisiCalc files.
- Campbell, Donald. "Spreadsheet Plus PERT, A Valuable Package." *Small Business Computers* 6, no. 3 (May/June 1982): 49-51. Describes the Desktop/Plan-III for the Apple /// including its ability to use VisiCalc files.
- Casella, Philip. "Graph, a Graphics Program for the Apple from SPC." *InfoWorld* 4, no. 30 (August 2, 1982): 54-55. A "software report card" on PFS:Graph, from Software Publishing Corporation, including the capability of using VisiCalc files, and some of the "necessary evils" of exchanging data.
- Coffey, Michael. "...Apple Cart..." *Creative Computing* 9, no. 2 (February 1983): 260, 262. Includes a book review of *The Power of VisiCalc* including discussion of a DIF example in that book.
- Coffey, Mike. "Form Fiddling." *Creative Computing* 8, no. 9 (September 1982): 50, 53. A product review of PFS (Personal Filing system), including mention of PSF:Graph, a companion product, and its use of DIF.
- "Combining Spreadsheet, Graphing and Information Management in One Program." *Personal Computing* 7, no. 1 (January 1983): 208, 211. Discussion of 1-2-3 including its DIF capabilities.
- "Computer-Based Business Files on the Move." *Small Business Computers* 6, no. 8 (March/April 1983): 12-16, 18-24. A lengthy discussion of

file transfer from microcomputers to large computer systems including discussion of approximately 25 communications and file transfer programs.

- "Computer-Based Business Planning." *Small Business Computers* 6, no. 3 (May/June 1982): 38-47. Presents the capabilities of spreadsheets, including the ability to transfer data from spreadsheet files to files used by other programs.
- D'Andrea, Lucy. "Check Book Register/General Ledger." Part 1, *Spreadsheet* 7 (November 1981): 7-8. Part 2, 8 (January 1982): 6. Describes a small checkbook to general ledger application using DIF.
- "Data*Trans." *Softalk* 3, no. 5 (January 1983): 138. Product review of the Data*Trans communication package, with discussion of its ability to transfer and create DIF files.
- "Decision Support System." *Interface Age* 8, no. 2 (February 1983): 136. Product discussion of Graph 'n' Calc including mention of its DIF capability.
- Deliman, Tracy and Chris Doerr. *PFS:Graph*. Software Publishing Company, 1901 Landings Drive, Mt. View, CA 94043. Product manual, including instructions on reading DIF files into PFS:Graph.
- Desautels, Edouard J. *VisiCalc for the IBM Personal Computer*. Dubuque, Iowa: Wm. C. Brown Company Publishers, 1982. Includes discussion on using DIF files. Also available in editions for TRS-80 computers.
- "DIF Program List." Wellesley, MA: *DIF Clearinghouse*. A list of commercially available programs that support DIF. (P.O. Box 638, Newton Lower Falls, MA 02162).
- "DIF Technical Specification." Wellesley, MA: *DIF Clearinghouse*. Technical specifications for DIF. (P.O. Box 638, Newton Lower Falls, MA 02162).
- "DIF Translator - VisiCalc to DB Master." *Stoneware Age* Summer 1982: 8. Announces a program, the DIF Translator, available from Stoneware (50 Belvedere St., San Rafael, CA 94901) to be used in conjunction with DB Master and Utility Pak #1 to transfer data in a DIF format from VisiCalc to DB Master.
- Ditlea, Steve. "Second-Generation Word-Processing Programs." *Popular Computer* 1, no. 8 (June 1982): 38, 42, 44, 46, 48. Discussion of word processing packages, including the Executive Secretary and Letter Perfect and their ability to access VisiCalc files.
- Edlin, Jim. "Easywriter to Get Improvements." *PC* 1, no. 2 (April-May 1982): 114. Brief description of using the Easywriter word processing program on the IBM Personal Computer to work with VisiCalc files.

- "Engineering Applications." *Interface Age* 8, no. 2 (February 1983): 124-125. Product discussion of TK!Solver including its DIF capability.
- Ewing, Richard and John Unger Zussman. *VisiFile User's Guide*. San Jose, CA: VisiCorp, Personal Software Inc., 1981. This product manual, for VisiFile, contains details on creating and using DIF files to interchange data with other programs such as VisiCalc.
- Ewing, Richard. *VisiTrend + VisiPlot*. San Jose, CA: VisiCorp, Personal Software Inc., 1981. Product manual, including details of the DIF interchange.
- "Exchanging Data Using the DIF Format." *SATN, The Journal for VisiCalc Users* 1, no. 1 (September/October 1981): 4. Brief summary of DIF, programs using DIF, and the DIF Clearinghouse. (See the *SATN* entry in this bibliography).
- Ferris, David. "Rules for the PC." *Software News* 2, no. 12 (December 1982): 54. The DIF format is included as one of the concepts users of the IBM PC should know to get their PCs working.
- Ferris, David. "Snags Mar Integrated 'Revolution'." *Software News* 3, no. 2 (February 1983): 61. In a general discussion of product integration, the processes of using DIF files for interchange are described as 'clumsy' in comparison to file transfer within an integrated package.
- Fluegelman, Andrew. "Calc Wars." *PC* 1, no. 4. (August 1982): 71-78, 80. Includes brief criticism of the VisiCalc documentation for the IBM Personal Computer for isolating the discussion of the DIF format in a technical appendix rather than placing a simple explanation of its usefulness in a main section of the documentation.
- Fylstra, Dan, and Bill Kling. *VisiCalc ATARI 800 32k*. San Jose, CA: VisiCorp (Personal Software), 1979. The VisiCalc product manual for the ATARI 800 microcomputer.
- Fylstra, Dan, and Bill Kling. *VisiCalc User's Guide for the Apple II and II Plus, 16 Sector Version*. San Jose, CA: VisiCorp (Personal Software), 1981. The VisiCalc product manual for the Apple II microcomputer.
- Gabriele, Rosemarie. "VisiTrend/VisiPlot." *Popular Computing* 2, no. 5 (March 1983): 166, 168, 171-172. Product review of this software including transfer of data from VisiCalc in the DIF format.
- Gitchell, Michael G. "The Accountant, a Financial Data Base for Apple II." *InfoWorld* 5, no. 1 & 2 (January 3 and January 10, 1983): 40-41, 43. A software review of this product including DBCALC, its VisiCalc conversion program.
- Good, Phillip. "Beyond VisiCalc." *Popular Computing* 1, no. 3 (January

- 1982): 38, 40-42. Descriptions of various business software products including several which use VisiCalc files.
- Good, Phillip. "VisiCalc, An Electronic Worksheet." *Popular Computing* 1, no. 2 (December 1981): 34, 36, 38. General product description with references to data interchange with other VisiCorp products.
- "Gregg Pkg. Automates Apple II Data Access." *MIS WEEK* 3, no. 45 (November 10, 1982): 19. Announces "Mainline," from Gregg Corporation to be available in the first quarter of 1983. The software "requests and passes information from remote mainframe databases to microcomputers via telephone" incorporating the DIF file format.
- Hancock, Ken R. "Terminal Communications for the Apple." *Creative Computing* 8, no. 5 (May 1982): 27-29. Software profile of VisiTerm including its relationship to VisiCalc and DIF.
- Heck, Mike. "The Executive Series." *Interface Age* 7, no. 12 (December 1982): 138, 140, 143. Includes discussion of The Executive Secretary and its ability to use data in the DIF format from DB Master Utility Pak #1.
- Heintz, Carl. "Guide to Database System Software." *Interface Age* 8, no. 2 (February 1983): 52-53, 55-58, 60. Includes brief mention of the DIF capabilities of VisiFile.
- Heintz, Carl. "Major Additions to VisiCalc Revealed." *Interface Age* 6, no. 8 (August 1981): 70-72, 148-149. Description of VisiCalc and related products from VisiCorp, Personal Software, including product interfacing with DIF.
- Heite, Ned. "Jinsam 8.0: Data-Base-Management for PET/CBM." *InfoWorld* 4, no. 25 (June 28, 1982): 63-65. Includes discussion of the interchange of files with VisiCalc through DIF.
- Hergert, Douglas. *Mastering VisiCalc*. Berkeley, CA: SYBEX, 1983. Includes three chapters on DIF files, including numerous examples of BASIC programs to process DIF files.
- Hixson, Amanda. "VisiTrend and VisiPlot, Trend Analysis and Plotting." *InfoWorld* 4, no. 17 (May 3, 1982): 34-36. Description of this product including interfacing with VisiCalc.
- Hughes, Patricia J. and Kaz Ochi. *The Power of VisiCalc-VisiFile*. Portland, OR: Management Information Source, 1982. Includes one chapter on the VisiCalc to VisiPlot data transfer with DIF.
- "Integrated Package as Easy as 1-2-3." *Desktop Computing* 3, no. 2 (February 1983): 79. Brief description of 1-2-3 including its use of DIF files.
- InterCalc. (See *SpreadSheet*).

- Jadrnicek, Rik. "VisiTran, A Utility Program for Users of VisiCalc." *InfoWorld* 4, no. 30 (August 2, 1982): 48-49. A software report card for VisiTran, rating it as "good", and describing it as "an interesting alternative to DIF file transfer (where) you often have to journey through a variety of cumbersome commands when saving a DIF file and then again when loading it into another program."
- Kalish, Candace E. and Malinda F. Mayer. "DIF: A Format for Data Exchange Between Applications Programs." *BYTE* 6, no. 11 (November 1981): 174, 176, 178, 180, 182, 186, 188, 190, 192, 194, 196, 198, 200, 202, 204, 206. Presents DIF in detail including BASIC and Pascal programs to create and read DIF files.
- Kapor, Mitchell. "VisiCalc and VisiPlot." *SATN, The Journal for VisiCalc Users* 2, no. 1 (September/October 1982): 1-5. An explanation of the use of the DIF format to exchange data between VisiCalc and VisiPlot and VisiTrend/VisiPlot.
- Korngold, Bob. "Building a Check Register Into a General Ledger." *Small Business Computers* 6, no. 8 (March/April 1983): 50-53. Includes the discussion of DIF files for moving a balance forward.
- Levy, Nick. "VC Magic for Your Christmas." *Windfall* December 1982: 32-34. Includes an example of datagramming (using VisiCalc in an execute mode) that includes saving the command file as a DIF file.
- "List Handler." *Softalk* 3, no. 6 (February 1983): 133. Review of the List Handler including mention of its use of DIF files.
- LOADCALC. Micro Decision Systems. Distributed by Cypher, 121 Second Street, San Francisco, CA 94105. Product manual for this text-to-DIF file utility.
- Magid, Lawrence J. and David Bunnell. "VisiCalc Creators Look Forward to Future Glory." *PC* 1, no. 4 (August 1982): 30-31. Includes a description of TK!Solver and its support of DIF.
- MAINLINE. Gregg Corporation, Waltham, MA 1982. Product documentation.
- Mason, Archie. "DIF Makes the Difference." *SpreadSheet* no. 9 (March-April 1982): 4-5. Detailed article describing the combining of quarterly quota/sales reports into regional and other categories.
- McGorry, David. "Alpha Pkg. Links Apple II, PC Files." *MIS Week* 3, no. 47 (November 24, 1982): 1, 25. Discusses the "Apple-IBM Connection," a product from Alpha Software that transfers files between the Apple II and the IBM Personal Computer including DIF files.
- McLamb, Ken. "Persuasion Made Easy with Powerful Presentations."

- Personal Computing* 6, no. 2 (February 1982): 24-27, 39, 84. Using graphics packages, including VisiTrend/Plot and its relationship to VisiCalc.
- "'Micromagic' Lets PCs Tap I.P. Sharp Files." *MIS Week* 4, no. 6 (February 9, 1983): 19. Describes Micromagic that allows users to access I.P. Sharp databases and create DIF files from them.
- Miller, David. "Aen Grading System Utility." *Apple Educators' Newsletter* 3, no. 16 (undated): 3. Describes the AEN Grading System for educators, and announces its support of DIF.
- Miller, David. *Apple Files*. Reston, VA: Reston Publishing Company, Inc., 1982. Contains a full chapter explaining DIF files, and includes BASIC programs to process them.
- Morgan, Chris. "Can We Agree on Standards?" *BYTE* 6, no. 11 (November 1981): 6-7. Informs readers of DIF, and Software Arts' efforts to encourage it as a standard.
- "Multifunction Package." *Interface Age* 8, no. 2 (February 1983): 124. Product discussion of 1-2-3 including mention of its DIF capabilities.
- Needle, David. "VisiCalc Creators Unveil 'No-Program' Problem Solver." *InfoWorld* 4, no. 23 (June 14, 1982): 6, 9. Description of TK!Solver, from Software Arts, and its ability to interface with VisiCalc.
- "New Business Software Packages." *COMPUTE!* 3, no. 6 (June, 1981): 147-148. Describes VisiCorp, Personal Software products, and their ability to pass information between programs.
- O'Connor, Rory J. "Micros Can File-Swap With DIF Data Format." *Computer Business News* 5, no. 7 (February 15, 1982): 1, 5; and a related letter to the editor from Bill Gurley, 5, no. 15 (April 12, 1982): 8. Discusses the establishment of DIF and the DIF Clearinghouse, and Software Arts' encouragement of its use.
- Olivieri, Peter. "Mind Your Business." *Softalk* 2, no. 10 (June 1982): 85-86, 88. Discusses a consultant's recommendations for software including VisiCalc, and the difficulty in selecting a product mix for clients including a DIF interface. Also 3, no. 1 (September 1982): 183-184, 186. Refers to *SATN* and its coverage of DIF. Also 3, no. 5 (January 1983): 115-116, 118, 120. Discusses Apple Business Graphics and its use of DIF files. Also 3, no. 6 (February 1983): 180-184. Includes a short explanation of DIF files.
- Paul, Philip F. "Business Decision Making." *SpreadSheet* 11 (July - August 1982): 6-9. Discussion of Decision Support Systems including brief discussion of graphics packages using DIF.

- Paul, Philip F. "Expansion of VisiCalc - An Easy Way." *SpreadSheet* 8 (January 1982): 3-4. Includes brief mention of using DIF to obtain more memory.
- Paul, Philip F. "Forecasting Templates from VisiCorp." *SpreadSheet* 12 (September-October, 1982): 2-3. A review of these templates, including reference to their use of DIF to transfer data between models.
- Perry, Robert. "Financial Modeling Software: Tools for the Overworked Manager." *Personal Computing* 5, no. 6 (June 1981): 22-28, 59-70, 108. This lengthy article compares approximately 15 financial modeling software programs, and briefly mentions DIF.
- Posner, John, Steven E. Miller, Ezra Gottheil, Jeff Hill. 1-2-3 Lotus Development Corporation, Cambridge, MA 1982. Product documentation.
- "Problem Solver." *DATAMATION* 28, no. 9 (August 1982): 150. Product description of TK!Solver, including mention of its support of DIF.
- "Programmer's Guide to the Data Interchange Format." Cambridge, MA: *The DIF Clearinghouse*. A guide to the language for data interchange developed by Software Arts, Inc. (P.O. Box 638, Newton Lower Falls, MA 02162). No longer published.
- Radding, Alan. "Rapid Reading of Research Data." *Personal Computing* 6, no. 1 (January 1982): 42-44, 47, 48, 148. Personal computer use in research including VisiCalc use with VisiPlot.
- Ramsdell, Robert E. "The Flexibility of VisiPlot." *BYTE* 7, no. 2 (February 1982): 32, 34, 36. Review of this product including data transfer from VisiCalc using DIF; and a letter to the editor, "Powerful ELF," 7, no. 6 (June 1982): 36, 40 from Eric Weiss.
- Ramsdell, Robert E. "The Power of VisiCalc: Product Review." *BYTE* 5, no. 11 (November 1980): 190-192. Early review of VisiCalc with strengths and program constraints and discussion of DIF.
- Risken, John. *The Executive Secretary User's Manual*. Product manual for this word processor.
- Sandler, Corey. "A Graphic Rendition." *PC* 1, no 8 (December 1982): 304-307. A software review of Graphmagic including its use of data in the DIF format.
- SATN (Software Arts Technical Notes). *The Journal for VisiCalc Users*, Software Arts, Inc., 27 Mica Lane, Wellesley, MA 02181. Bimonthly journal for VisiCalc users published by the creators of VisiCalc.
- "Saving Thirty Hours with DIF." *SpreadSheet* no. 9 (March-April 1982): 4. A brief summary of experience using DIF to combine quarterly quota/sales reports and forecasting.

- Schilling Jr., Robert. "Silicon Office." *Popular Computing* 1, no. 12 (October 1982): 78, 82-84, 86. Software review of the Silicon Office, which contains a data base manager, electronic calculator, word processor, and communications package. It takes the "first major step in simplifying (the) process" of data interchange since DIF which "still assumes you have a fair knowledge of programming and file structure."
- Shelton, Joe. "The Customization of VisiCalc: for the Apple ///." *Softalk* 2, no. 12 (August 1982): 186-188. Compares the template consolidation steps using DIF available with VisiCalc and VisiCalc Advanced Version.
- Shelton, Joe. "Ventures with VisiCalc." *Softalk* 2, no. 8 (April 1982): 143, 145-146, 148. Brief mention of the /S# command of VisiCalc. Also 2, no. 11 (July 1982): 115-116, 118. Reviews of graphics packages for the Apple II and Apple /// including those that use DIF. Also 3, no. 1 (September 1982): 203-204, 206. Centers on discussion of DIF within VisiCalc Advanced Version and VisiCalc for consolidation.
- Simpson, Tom and Shaffer & Shaffer. *VisiCalc Programming: No Experience Necessary*. Boston, MA: Little, Brown & Co. Refers to DIF capability.
- Sobel, Joseph J. with Barry D. Bayer. "VisiCalc Tips and Techniques." *Apple Orchard* 3, no. 3 (July-August 1982): 48-54. Suggests using DIF to save files of "formats" (data frameworks which are used often) that can then be loaded on a template as needed. "Part 2: Printing, Saving, and Enhancements." *Apple Orchard* 3, no. 4 (October 1982): 74-76, 78, 80. Includes details of creating DIF files, Overlays with several suggested uses of the files with VisiCalc. Also "Part 4 Overlays, DIF and other Goodies" 4, no. 1 (February 1983): 36-38, 40-43. Includes discussion of DIF files in several contexts; however, the article incorrectly presents the tuple layout of DIF files created by VisiCalc.
- SpreadSheet*. VisiGroup 1, no. 1 (November 1980, and subsequent issues). A valuable newsletter from VisiGroup (P.O. Box 1010, Scarsdale, NY 10583), National VisiCalc Users' Group. An index of articles from issues 1-7, 1981 appears in the January 1982 issue. After March-April, 1982, Issue 9, the name of the group was changed to *InterCalc*.
- "Standard Data Interchange Format." *Small Business Computers* 6, no. 3 (May/June 1982): 15. Brief discussion of DIF and the DIF Clearinghouse.
- Stein, Donna. "Your Business Image." *Business Computer Systems* 1, no. 3 (November 1982): 62-68. Discussion of business graphics, including

reference to twenty packages, many using DIF.

- Stein, Donna. "Software Renaissance a Boon for Business." *Business Computer Systems* 2, no. 1 (January 1983): 78, 80, 82. Discusses the "push" for standardization including a forecast for additional products that use DIF.
- Stewart, George. "TK!Solver." *Popular Computing* 1, no. 12 (October 1982): 53-56. Software review of TK!Solver including its DIF capabilities.
- Stinson, Craig. "Exec Context: The Software Synthesis." *Softalk for the IBM Personal Computer* 1, no. 7 (December 1982): 10-12, 14. Discusses the executives of Context, the company producing the MBA, a single product integrating a spreadsheet, data base, text processor, and graphing module. The article includes the comment that they "...thought DIF sounded inadequate as a tool for integrating software functions."
- Stinson, Craig. "Using VisiCalc and Magic Window." *SATN* 2, no. 3 (January/February 1983): 1-4. A discussion of interchanging data between these two products using the VisiCalc print-to-disk capability (not DIF files).
- Stinson, Craig. "Ventures with VisiCalc." *Softalk* 1, no. 11 (July 1981): 14-17. Describes the Apple 3.3 version of VisiCalc, including DIF capabilities, and other VisiCorp, Personal Software, products. Also 2, no. 1 (September 1981): 91-92. Presents capabilities of the Context Connector, from Context Management Systems, including consolidation of sheets, and compares this capability to VisiCalc's DIF. Also 2, no. 2 (October 1981): 50, 53. Describes a problem in the storage of DIF files with the first release of the Apple 3.3 version of VisiCalc. Also 2, no. 3 (November 1981): 65-66. Includes brief discussion of the problem in storing DIF files with the first release of the Apple 3.3 version of VisiCalc.
- Stinson, Craig. "The View from COMDEX." *Softalk for the IBM Personal Computer* 1, no. 8 (January 1983): 92-94. Includes discussion of 1-2-3, VisiON, and MBA and discusses DIF within this context.
- "Super Data Interchange." *SuperNews* no. 1, Second Quarter 1982: 1. SORCIM newsletter which briefly mentions Super Data Interchange SDI, and a program which will also support conversion using DIF.
- "TK!Solver (in four parts)." Software Arts, Inc., 27 Mica Lane, Wellesley, MA 02181. Product documentation.
- Trost, Stanley R. *Doing Business with VisiCalc*. Berkeley, CA: Sybex Inc., 1982. Includes several examples of the use of DIF files within VisiCalc and in VisiCalc-to-plotting programs.

- Turing, Al (compiler). *DB Master Utility Pak #1*. Stoneware Microcomputer Products, 50 Belvedere Street, San Rafael, CA 94901. Documentation for this utility that includes a DIF file translator.
- ULTRA PLOT D.I.F Interface* and *ULTRA PLOT*. Avant-Garde Creations, P.O. Box 30160, Eugene, OR 97403. Documentation for the products of the same names.
- UPTREND*. Sunnyvale, CA: Personal Software Inc. 18 pages. A publication for users of the firm's software systems, including the use of DIF. No longer published.
- Vanderburgh, Richard C. *VisiCalc VisiSected*. Published by the author: 9459 Taylorsville Road, Huber Heights, OH 45424. A detailed dissection of VisiCalc code, including discussion of DIF, with a lengthy documented disassembly of it.
- "VC-Loader." *Softalk* 2, no 11 (July 1982): 142, 145. Report on this utility that transfers data between programs, creating DIF files from text input files.
- "VC-LOADER." *SpreadSheet* 10 (May-June 1982): 9. Brief description of this product that creates DIF files from text file input.
- "VC-Manager." *Softalk* 2, no. 9 (May 1982): 129. Impressions of this product from Micro Decision Systems, describing it as filling the need to consolidate files and send data from one file to another, and mentioning DIF use for this function.
- "VisiCalc 3.3 for the Apple" *SPREADSHEET* 1, no. 5 (July 1981): 7-8. Describes changes to VisiCalc introduced with this version of VisiCalc, including comment on the value of DIF.
- "The VisiCalc Calendar." *SATN: The Journal for VisiCalc Users* 1, no. 3 (January/February 1982): 1-4. Provides a calendar-producing template, saving the data generated in a DIF file.
- "VisiCalc Forecasting Package Provides Reports, Analyses." *Computer Business News* 5, no. 20 (May 17, 1982): 18. Announcement of VisiCorp's Business Forecasting Model spreadsheets which use VisiCalc and DIF to transfer data from one sheet to another.
- Williams, Robert E. and Brian L. King. *The Power of VisiCalc Volume II*. Portland, OR: Management Information Source, 1982, 97 pages.
- Winkler, Connie. "'1-2-3' Puts DSS-like Facilities on IBM's PC." *Software News* 2, no. 11 (November 1, 1982): 25. Discussion of the "1-2-3" product from Lotus Development Corporation, including its use of DIF.
- Wise, Deborah. "Accountant Home-Finance Package Abjures Jargon." *InfoWorld* 4, no. 21 (May 31, 1982): 13. Describes the Accountant

Finance Data Base System, from Decision Support Software, a double-entry accounting package which interfaces with VisiCalc.

Wolverton, Van. *VisiCalc User's Guide for the IBM Personal Computer*. San Jose, CA: VisiCorp (Personal Software), 1981. Includes an appendix on DIF which contains BASIC programs that print a DIF file as stored, print a worksheet from a DIF file, and create a DIF file.

Index

0 (zero), 156
1-2-3, 121-128
 data base, 121
 graphics, 121
 integrated software, 121
 reasons for using DIF files with 1-2-3, 121-122
 spreadsheet, 121
 Translate option, 123

= (see PFS:Graph)
! (see TK!Solver)
+ 156 (see also VisiCalc)
(see VisiCalc)
" 163-165 (see also VisiCalc)
/ (see VisiCalc)
.DIF, 7, 17, 19, 94, 119, 123, 125, 144-145
.ML, 118
.PRN, 101, 103, 107
.VC, 7, 19, 115, 118
.WKS, 123, 124

A

All-in-one software (see Integrated software)
Apple computers, 10, 12, 13, 19, 37, 51, 72, 91, 112, 145
Arrow key usage in VisiCalc (see VisiCalc)
Artist prepared graph, 51, 53

B

B*, 117
Backup files, 145-146
Base-10, 156
BASIC, 4, 142, 166
Beginning Of Tuple (*see* BOT)
BOT, 158, 164
BREAK key, 13
Budget data, 2, 28, 43
Built-in functions in VisiCalc (*see* VisiCalc)

C

C use in column indication (*see* Column)
C use in filenames, 19, 145
Cancel keys, 13
Clean a file before DIF creation, 141
Column (C) save/load of DIF files, 13, 25, 88, 162
 example (*illus.*), 14
 order, 15, 157
Combining VisiCalc sheets, 46
COMMENT, 166
Common data format, 2
Common elements, sharing in VisiCalc, 34
Comparing VC and DIF files, 19-24
Compatibility with VisiCalc, 1
CompuServe, 112-120
Content of DIF files, 19-24
Copying rectangular areas of VisiCalc worksheets, 30
Count, 164
Creating DIF files from VisiCalc (*see* VisiCalc)
Creator of DIF format, R. Frankston, 1
CTRL-C, 13
Cursor
 pointing, 13
 position when creating and loading a DIF file with VisiCalc (*see* VisiCalc)
Cut-and-paste with DIF files in VisiCalc, 30

D

DATA, 163
Data base, 121 (*see also* DB Master)

Data entry, avoiding, 2
Data interchange case studies (*see individual product names*)
Data processing system, 4-5
Data Section, 154, 155-162, 164, 172
Data Value, 155, 156, 157, 158, 160
DB Master, 91-100, 166
 DIF Translator - VisiCalc to DB Master, 99
 TRANSLATE FILE, 93
 Utility Pak #1, 91, 93, 94, 96
dBASE II, 124
DEC, 4, 117
Decimal point, 156
DIF Clearinghouse, 3
DIF Translate - VisiCalc to DB Master, 99
Difference files (.DIF), 4
Digital Equipment Corporation, 4
Disk access time, 37, 38
Diskette, 11, 17, 94, 99, 148
 swapping, 99, 172
Diskette space reduction in VisiCalc, 37
Display Information Facility (DIF), 4
DISPLAYUNITS, 166
Document Interchange Facility, 4
Documenting data exchange, 5, 70, 83, 85, 88, 115, 140-141, 143-144,
 147, 149-153
 examples (*illus.*), 84, 86, 87, 151, 152-153
Dow Jones News/Retrieval, 122

E

EDLIN, 168
End of data, (*see EOD*)
ENTER key (*see RETURN key*)
EOD, 158
EPSON MX-100 printer, 51, 72
@ERROR (*see VisiCalc*)
Error messages, 140, 141, 147
Executive Secretary, The, 91-100
 commands, 96-97

F

@FALSE (*see VisiCalc*)
FI, 21, 34, 141

File integrity, reason for exchange, 2
File type, 7
Filename, 12, 17, 19, 118, 145-146
 (see also .DIF)
 suffix usage, 7, 17, 19
 symbol used, 7
 VisiTrend/Plot (*see* VisiTrend/Plot)
First entry becomes filename, 60
Formats from VisiCalc not saved in DIF files (*see* VisiCalc)
Formulas from VisiCalc not saved in DIF files (*see* VisiCalc)
Frankston, Robert M., 1

G

GFI, 34
Graphics software
 use increases, 51
 (see also 1-2-3)
 (see also PFS:Graph)
 (see also VisiTrend/Plot)
Grappler interface card, 51, 72
Gregg Corporation, 112
Guidelines for exchange, 5, 139-148

H

Hardware, 4
 data exchange between machines, 4
Hayes Micromodem II, 112
Header Item, 162-164, 166
Header Section, 154, 160, 162-166, 172
HP-83/85 computers, 10

I

IBM, 4, 117
IBM Personal Computer, 12, 13, 19, 121, 124, 129, 145, 168
IBM System/38, 4
Ignoring data in interchange, 51-52
Importance of DIF files, 2
Increased use of DIF files, reasons, 1
Inflation rate, 2

@INT (*see* VisiCalc)
Integer format in VisiCalc (*see* VisiCalc)
Integrated software, 4, 121

J

Justification of labels (*see* VisiCalc)

K

Kilobytes, 35

L

LABEL, 166
Label data, 162
Label entry becomes filename in VisiTrend/Plot (*see* VisiTrend/Plot)
Limitations in data exchange, 90, 99, 170-172
Line Editor, 168
Listing formulas and formats in VisiCalc (*see* VisiCalc)
LoadCalc, 101-111, 122
 similarities to VisiCalc, 103
 steps to create a DIF file, 107-109
Loading DIF files into VisiCalc (*see* VisiCalc)
@LOOKUP (*see* VisiCalc)
LOTUS GRAPH, 124
LPT1, 19

M

Mainframe computers, 1, 114, 120, 143, 149
 accessing data from, 4
MAINLINE, 112-120, 122
 B*, 117
 commands, 117
 data retrieval, 112, 114-118
 login procedures, 112-113
 stock market data, 114
Major start date (*see* VisiTrend/Plot)
MAJORSTART, 166 (*see also* VisiTrend/Plot)
Memory usage of DIF files, 35-36
MICROCOMM, 122

MICROMAGIC, 122
Minor start date (*see* VisiTrend/Plot)
MINORSTART, 166 (*see also* VisiTrend/Plot)
Multiple DIF files, 83

N

N, No Change in VisiCalc (*see* VisiCalc)
@NA (*see* VisiCalc)
Naming DIF files, 7, 17, 19
 (*see also* Filename)
Need for standard (*see* Standard format)
Negative numbers, 156
Non-numeric data, 60, 76, 80, 88, 170-171
Number Value, 156, 159

O

Optional items in DIF format, 3, 58
Overhead in DIF files, 160, 171

P

Pascal, 4, 142, 166
PERIODICITY, 166 (*see also* VisiTrend/Plot)
Periodicity (*see* VisiTrend/Plot)
Personal computers, 1
PFS:Graph, 72-90, 144
 =, 80
 data reduction, 90
 dates, 72, 80
 error messages, 141
 filenames, 74
 identifiers, 72
 invalid data, 7
 multiple DIF files, 83
 non-numeric data, 88
 numeric data, 72
 plots (illus.), 75, 82
 relative data, 88
VisiTrend/Plot, compared to, 76
X-Y coordinates, 72, 76

@PI (*see* VisiCalc)
Pointing the cursor (*see* Cursor)
Pound sign (*see* VisiCalc, #)
Pounding values in place in VisiCalc, 34, 37, 151-152
Print command of VisiCalc (*see* VisiCalc)
Print file, 101, 103, 111
Printing VisiCalc sheets in parts, 30, 31-33
Procedures for exchange, 143
Process of exchange, 147-148
Products that use DIF files, 1, 3, 3-4, 202-212
Programming DIF file usage, 3-4

R

R (*see* Row)
R, Relative in VisiCalc (*see* VisiCalc)
Reasons for DIF file use, 2
Recalculation time reduction in VisiCalc, 37-39
Reconfiguring VisiCalc sheets, 24-29
Rectangular data area, 69, 171
Repeating Label command of VisiCalc (*see* VisiCalc)
Replicate command (*see* VisiCalc)
Required items in format, 3
RETURN key, 13
 same as R, 15
Rolling data from period to period in VisiCalc, 43
Row (R) save/load of DIF file, 13, 25, 88, 162
 example (illus.), 14
 order, 15, 157
 R use in filename, 19, 145

S

Salary increment example, 129-137
Scientific notation, 156
Scrolling diskette directory in VisiCalc (*see* VisiCalc)
Security of data, use in VisiCalc, 35
Separate data area (*see* VisiCalc)
SIZE, 166
Software Arts, Inc., 1, 3
Software using DIF files (*see* Products that use DIF files)
SP, 19

Special Data Values, 158-159
speedSTAT, 122
Standard format, 1, 2, 171
Stock market data retrieval, 114
 (*see also* VALUE)
Stoneware, 99
Storage command of VisiCalc (*see* VisiCalc)
Storing multiple "what if..." computations, 41-42
String data, 160
String Value, 156, 158
SUPR'TERMINAL, 91
Suffix usage (*see* Filename)
Symbols used in this book, 6, 7
System, use of DIF files within, 4-5, 170, 172

T

Table, 163, 163-164
TABLE, 157
Technical specifications of DIF files, 3, 10, 49
Text files, 103, 122
The Executive Secretary (*see* Executive Secretary)
TK!Solver, 129-137
 ! (Action command), 135, 136
 LIST SHEET, 131-132, 134
 RULE SHEET, 130, 131, 134, 135
 VARIABLE SHEET, 130, 131, 135
@TRUE (*see* VisiCalc)
TRUELENGTH, 166
Tuple (record), 157, 158, 159-162, 163-165, 166, 168
TUPLES, 163
Two-dimensional plotting (*see* VisiTrend/Plot)
Type Indicator, 156, 159

U

UNITS, 166
Uses for DIF files, 8-9
Utility Pak #1 (*see* DB Master)
Utility program, 101, 103

V

V, 156
VALUE, 112, 113-114
VAX, 4
Vector (field), 157, 158, 159-162, 163-165, 166
VECTORS, 163
VisiCalc
+, 62
#, 12, 35
", 24
/, 12, 13
arrow key usage, 12
Built-in functions
 @ERROR, 24, 30, 162
 @FALSE, 24, 162
 @INT, 34-35, 142
 @LOOKUP, 109
 example (illus.), 110
 @NA, 24, 162
 @PI, 24
 @TRUE, 24, 162
C, columnwise recalculation, 35
CompuServe (*see* CompuServe)
copying formulas, 62
creating DIF files, 10-17
creators, 1
cursor pointing, 13
cursor positioning, 11, 17, 28, 43
data entry area, 62, 66, 77, 115, 142
 example (*illus.*), 67, 78, 79
DIF file uses within a VisiCalc context, 24-48
DIF usage within, 10-48
documentation (*see* Documenting data exchange)
filenames (*see* Filename)
formats not saved in DIF files, 19-24
formulas not saved in DIF files, 19-24, 28, 30, 34, 35
justification of labels, 21, 24
listing formulas and formats, 19
loading DIF files, 12, 17, 28
MAINLINE (*see* MAINLINE)

- VisiCalc (*Contd.*)
 - Move command, 28
 - N, No Change, 30
 - Print command, 11, 30, 111
 - printing, 30, 31-33
 - prompt lines, 12-13
 - R, relative, 30
 - R, rowwise recalculation, 35
 - Repeating Label command, 21, 24, 171
 - Replicate command, 24, 30
 - scrolling diskette directory, 12
 - Storage command, 12-17
 - use to clean data file, 141-142
 - utility-like usage, 68, 147
 - versions, 10
 - versions not supporting DIF files, 10
 - VisiFile interchange, 140
 - VisiTrend/Plot interchange, 51-69
 - VisiWord exchange, 101-111
 - "what if...", 66
- VisiFile, 140
- VisiTrend/Plot, 51-69, 144, 146, 166, 172
 - charts (*illus.*), 57, 59, 61
 - filename, 60
 - increment, 52, 69
 - internal data format, 58
 - label entry becomes filename, 60
 - LOAD option, 56
 - major start date, 54, 64, 69, 70, 71, 76, 168
 - minor start date, 54, 64, 69, 70, 71, 76, 168
 - periodicity, 56, 70, 76, 168
 - PFS:Graph comparison, 76
 - SAVE option, 64
 - steps for exchange to VisiCalc, 58, 66
 - two-dimensional plotting, 52, 54, 69
- VisiWord, 101-111
 - DISK option, 103
 - print file, 101
 - PRINT menu, 103

W

Word processing

- usage to process DIF files, 111, 147, 166-169, 170
 - (*see also* Executive Secretary)
 - (*see also* VisiWord)

Y

- Year-to-date computations in VisiCalc, 46, 48

THE DIF FILE

Reader Survey: We'd like to have your comments on THE DIF FILE and suggestions on other titles you'd like to see in our Data Integration Series.

Please send one free of charge Reston Computer Catalog.

Yes, I'd like to order additional books by Don Beil:

**The VisiCalc Book/Apple Edition* \$14.95

**The VisiCalc Book/IBM Edition* 16.00

**SuperCalc! The Book* 16.95

Please add appropriate sales tax _____

TOTAL \$_____

Name _____

Address _____

City _____ State _____ Zip _____

Check or money order enclosed Bill me

**PLACE
STAMP
HERE**



**Nikki Hardin
Reston Publishing Company, Inc.
11480 Sunset Hills Road
Reston, Virginia 22090**

Software Integration Series
Donald H. Beil, Series Editor
The DIF File

The use of the DIF file has spread dramatically in recent months, but little has been written to help the computer user benefit from this "phenomenon." DIF, a format for exchanging data, is not a product that can be bought in a computer store. Instead, the ideas it contains, which were developed by Software Arts, Inc., are for anyone's use.

This book takes a look at the wide variety of potentially unassociated software currently available and offers extended case studies that demonstrate data exchange between some of the most popular of those software products. The information contained in this book is of the utmost importance to anyone who uses one of the products that support DIF and who wants to interchange data with another products that support DIF.

The DIF File includes 12 case studies that demonstrate data interchange among some of the prominent software on the market today. It shows how data can be exchanged between popular software including: VisiCalc®, VisiTrend/Plot™, PFS:Graph™, DB Master™, The Executive Secretary™, VisiWord™, LoadCalc™, MAINLINE™, CompuServe™, 1-2-3™, and TK!Solver™.

The Table of Contents includes:

- General Information About the DIF Format
- Using the DIF Format in a VisiCalc Context
- Case Studies: Notes on the DIF Format Case Studies
 - Data Interchange Between VisiCalc and VisiTrend/Plot
 - Data Interchange Between VisiCalc and PFS:Graph
 - Data Interchange Between DB Master and The Executive Secretary
 - Data Interchange From VisiWord to VisiCalc Using LoadCalc
 - Data Interchange From CompuServe to VisiCalc Using MAINLINE
 - Data Interchange From DIF Files to 1-2-3
 - Data Interchange From DIF Files to TK!Solver
- Guidelines for Using DIF Files
- Documenting Data Interchange
- A Tutorial on the DIF Format
- Limitations of a System Using the DIF Format

RESTON PUBLISHING COMPANY, INC.
A Prentice-Hall Company
Reston, VA 22090

0-8359-1305-8

Cover Design by Diane Vogel