Decision Table Software

A Handbook

by Herman McDaniel
U.S. Civil Service Commission

Brandon/Systems Press, Inc.
Princeton New York London

Library of Congress Catalog Card Number 78-101001
Standard Book Number: 87769-024-3

First Printing

Printed in the United States of America

DECISION TABLE SOFTWARE— A HANDBOOK

PREFACE

This book is intended as a guide for the data processing manager or systems analyst who has decided to "look into" decision table processors. Here, in concise, easy to read form, is basic information regarding more than thirty processors. Language, hardware, availability, and cost information are provided, along with a list of major features attributed to each processor. Special reference manuals are indicated for some of the processors. Segments of computer programs that have been generated by some of the processors are shown.

It should be a relatively simple task to select those processors that are acceptable at first glance on the basis of hardware, language, availability, and cost. The features should be checked thoroughly to see what benefits might be realized within an organization. More detailed information is required to make a final selection, and sources for such information are suggested.

No attempt has been made to evaluate and rank the available decision table software. To avoid giving any impression of ranking, the software is arranged in alphabetical sequence by name. The amount written about a particular processor has no relationship to its value or effectiveness. The longer write-ups are directly attributable to the fact that more information is available regarding some processors.

In view of the rapid changes and continuous updating of the features by some vendors, neither the author nor the publisher should be held responsible for inaccuracies in this area. Every effort has been made to be accurate and fair in describing the various processors, but errors undoubtedly have crept in. If the reader detects errors or omissions, he is urged to notify the author (through the publisher) so that future editions of this book will be more accurate.

The information contained in *Decision Table Software* should prove valuable to anyone attempting to select a decision table processor.

Herman McDaniel

Washington, D.C.

ACKNOWLEDGMENTS

A project such as this book is never the result of the efforts of a single individual. The author must obtain information from others who are knowledgable in the field.

In gathering information for this book, I sought out all of the American computer manufacturers, the major European and Asian manufacturers, and the primary American software vendors. The following people deserve recognition for contributing useful information toward this endeavor; in view of the rapid changes of affiliation in the EDP field, only the names of the individuals are listed: James M. Adams, Jr., Park F. Anderson, Jr., Paul Anson, Jill Arbudde, G. W. Armerding, Charles W. Bash, Ronald L. Basler, Florence Bernhardt, Harry Bjork, C. W. Blaxter, R. Boot, Richard T. Bueschel, Tom Caldwell, A. Chiappinelli, Jr., H. G. Conrad, Wesley H. Cowley, Judith Currie, Joseph D'Aulerio, Jr., Carol Davidson, Henri Denoff, Donald J. Devine, Carl Diesen, P. J. Dixon, Bert Engelhardt, Howard Fletcher, James F. Foley, Barry Forester, L. H. Foxx, Jerome W. Geckle, Frank X. Goelz, George G. Hancock, Jr., Fred G. Harold, John S. Hermistone, T. F. Kavanagh, W. P. Keating, Henry Kee, Kamil Khan, B. H. King, Jr., Thomas A. Kraska, David Lawrence, John K. Lenher, H. I. Meyer, James I. Morgan, Svein Nordbotten, James L. O'Brien, George Oerter, A. Michael O'Reilly, Carl Payden, Larry M. Pigg, Sol Pollock, Joseph V. Popolo, Lewis T. Reinwald, Perros Roebas, C. B. Rogers, Jr., H. L. Shoemaker, James W. Snively, Jr., Charles R. Sterbakov, William E. Sullivan, Philip S. Thornton, Everett B. Turner, Robert Van Roijen, Ira Victor, Karl-Heinz Wobig.

INTRODUCTION

A decision table processor is a software program for translating decision tables into executable computer programs. Such translators are now available to translate decision tables to computer programs for almost any hardware configuration and programming language desired.

Of course, the tables to be processed must conform to certain standards for a particular processor—table size, format, words used in the stubs, and so forth. Some processors require that the entire computer program be depicted in decision table form; others allow a mixture of own coding and decision tables. Once the necessary tables have been structured, the processor will generate the appropriate coding to achieve the desired result.

Even before the structuring of the tables, the problem must be broken into usable segments or modules. The tables are usually manually checked after construction—before the information is keypunched for input to the computer.

At the computer, the decision table processor is loaded into the computer, and the decision tables are read. The processor examines each table and produces coding to accomplish all of the *if–then* relationships contained in the table. Once this has been completed, the processor might call attention to any missing situations, point out any redundancies, or pinpoint any contradictions within the table.

Each decision table results in a segment of coding. Hence each segment of the program can be traced directly to the decision table that caused its generation. Changes can be made to an existing program with relative ease by changing one or more decision tables. The effect of such change on the program is easy to observe and evaluate.

Each program will follow the same type of coding approach. The coding will be straightforward and free of cute gimmicks and programming tricks reflecting the personality of a particular programmer. Each programmer should be able to follow any program in the installation and make the required changes.

It is generally conceded that there are three types of decision table processors: (1) The decision table interpreter is basically an object program written in a machine language. The interpreter is a series of subprograms. As a decision table is examined, each situation will cause a transfer of control to the appropriate subprogram. Upon the completion of each subprogram, control is returned to the main program. Such interpreters require a precise language that allows a very limited vocabulary. Changes or modifications are made to the tables themselves; consequently the documentation for a program always reflects the same logic as the running program. Execution time on the computer for a program generated by an interpreter is usually greater than that required for handcoded programs.

(2) The decision table translator translates one language into another. This requires that words used in the decision tables be words which may be easily translated into FORTRAN, COBOL, or some similar compiler language. Again, a precise language is required. An intermediate language is used in the translating, thus requiring longer compilation time. This also allows certain inefficiencies to creep into the translating process, which usually get carried over into the object program. Programs can be modified at the intermediate language level, but if this is done, the documentation will no longer agree with the running program.

(3) Decision table compilers are capable of taking a source language and generating the appropriate instructions for the computer. The language might be an existing compiler level programming language, or it may be one specially designed for subject matter specialists using decision tables. Such compilers usually translate decision tables into efficient object programs. Many compilers employ optimizing techniques as well as numerous checks for contradictions, omissions, and redundancies. When such situations are detected, the compiler will provide the appropriate diagnostic messages for the user.

Since the terms "decision table processor" and "decision table translator" have been used interchangeably during recent years, no attempt has been made in this book to place the various pieces of software into one of the above categories.

The use of a decision table processor should greatly reduce programming effort, time, and costs within an organization. A program should require significantly less computer time for compilations and tests before it is ready to be put into production. Maintenance caused by overlooked situations or inconsistencies in the program should be reduced.

Decision table software has improved dramatically in the last few years. Those knowledgable about decision table processors and their capabilities generally concede that the quality of programs generated from decision tables by a processor is now as good as that manually coded by a better-than-average programmer. In some instances, where a table processor was written to take advantage of certain hardware quirks, the generated coding actually executes faster than comparable handcoded jobs.

Some installations have found that with the newer processors intelligent subject matter specialists can be taught to generate their own programs with relative ease. The amount of training required to teach them how to work with a decision table processor is only a fraction of the effort required to teach them a programming language.

In addition to the cost of a decision table processor, which might range from "free" to more than $40,000, the manager must be prepared to make other investments in order to achieve the benefits a table processor can provide. These investments include proper decision table training for all those who will construct and use decision tables. An hour's tutorial by a fellow programmer or analyst is not sufficient.

Management must plan for decision table implementation. The manager frequently expects too much too soon. People work much more slowly with a new tool; consequently early jobs should be expected to require more time than subsequent ones. Implementation planning should take this into consideration.

Such planning should also include the establishment of standards for the construction and use of decision tables in the organization. Without such standards, chaos usually reigns.

Decision tables offer a bright future for those who are willing to pay the price—proper planning, adequate education, and careful selection of the right processor for the installation's requirements.

CONTENTS

AUTOCODER DECISION TABLE ASSEMBLER

The Autocoder Decision Table Assembler allows the use of decision tables within the assembly language for certain IBM computers. Consequently the programmer can relatively easily give a compact, yet readable, representation of complicated logic and its relationships.

Features Decision tables are permitted within normal assembly language coding.

A separate and distinct Exit area follows the action stub on each table.

No housekeeping commands are generated to separate sequential actions.

Corrections may be made directly to the Autocoder coding after it has been assembled.

A pre-edit feature for the decision table cards is available.

Output coding may be directed onto tape.

Testing of the tables is on a condition-by-condition basis.

Regular Autocoder statements must be used in stub portions of the tables.

Only limited entry tables are permitted.

Tables must be sorted into a condition-by-condition test pattern.

Up to 50 lines are allowed for action and exit statements.

A maximum of 15 rules is permitted in a single table.

Hardware	IBM 7070 with a minimum of 5K memory and six tape units. Can also be used on IBM 1401 (4K minimum) with Hi-Lo-Equal Compare, Sense Switches, and card reader/punch.
Language	IBM Autocoder
Reference	*Autocoder Decision Table Assembler* (1.1.002), The Guide General Program Library.
Availability/Cost	This decision table assembler is available as a part of the Guide General Program Library at no cost.

BASIC DETAP

Information Management Incorporated has a family of three decision table processors. The processors have certain similarities and certain differences; consequently, they are treated in this book as three separate and distinct decision table processors. For information on the other two processors available from IMI, see the section on Compact Detap and Detap.

Features Among the features attributed to Basic Detap are these:

Basic Detap converts limited entry decision tables to Cobol.

Each decision table appears in the Cobol program as a Note.

Sequence of actions common to several rules are grouped into paragraphs and coded only once.

Only limited entry tables can be processed.

Initialization cannot be achieved through a decision table.

Up to 50 conditions may appear in a single table.

A maximum of 50 actions is allowed in a single table.

A table may contain up to 50 rules.

Basic Detap is said to be upward compatible with Compact Detap and Detap.

Hardware The distributor claims that Basic Detap operates on any computer that has a Cobol compiler and a minimum of 32K memory.

Language Cobol

Availability/Cost At the time of this writing, Basic Detap, including on-site training, users' manuals, and a year's maintenance, is available at a cost of $5,500. Contact:

Information Management, Inc.
447 Battery Street
San Francisco, California 94111

or

11 West 42nd Street
New York, New York 10036

Betab (from "*be*sluts*tab*ellen") first made its appearance in Sweden in 1968. While it appears to be based to a great extent on the Detab efforts in this country, it is not a Cobol processor. It processes decision tables to generate Algol output. This should prove interesting to European decision table enthusiasts.

Betab is a single pass system with simultaneous paper tape reading and punching. Consequently, speed is restricted to the speed of a paper tape punch (approximately 150 characters per second).

Features Betag checks input tables for completeness, contradictions, and redundancies; appropriate diagnostic messages are given.

Nested block structure of tables is permitted.

Betab provides an extension to the processing capabilities of Algol 60.

Most Algol restrictions must be imposed on the tables. Tables are restricted to a maximum of 21 rules (the upper limit allowed by the range of positive integers in Algol).

There is no stated limit for the number of conditions and actions.

Only limited entry tables may be processed.

Hardware Betab has been installed on DataSaab D21 and D22. A minimum of 16K 24-bit words is required. Computer must also have a paper tape reader and punch, a lineprinter, and a minimum of four magnetic tape units.

Language The processor is written in a mixture of Algol and of Genius. The Results of the decision table processing are in Algol 60.

Reference A detailed description of this decision table processor is available (in Swedish language) in *Beslutstabeller i Algol 60,* DataSaab 9006, A6372.01.23, February 1969.

Availability/Cost Betab/68 is a part of the DataSaab program library and is available without cost to DataSaab users from:

Saab Aktiebolag
DataSaab
Linkoping, Sweden