

# RIPPLE-TRAC For Source Code and CICS



©

Written by  
Daniel B. Boucher Sr,  
z/OS Solutions Architect,  
Logic Online, Inc.

A Separation of Concerns (SoC) Tool

[www.logiconlineinc.com](http://www.logiconlineinc.com)

This document contains proprietary information, protected by copyright. No part of this document may be reproduced or transmitted for any purpose other than the reader's personal use without the permission of Logic Online Inc.

## **WARRANTY**

The information contained in this document is subject to change without notice. Logic Online makes no warranty of any kind with respect to this information. Logic Online shall not be liable for any direct, indirect, incidental, consequential, or other damage alleged in connection with the use of this information.

## **TRADEMARKS**

All trademarks and registered trademarks used in this guide are property of their respective owners.

RIPPLE-TRAC is a registered trademark in the state of New Hampshire

Logic Online Inc.  
1500A Lafayette Rd #170  
Portsmouth, NH 03801  
e-mail: [info@legacyimpactanalysis.com](mailto:info@legacyimpactanalysis.com)  
[www.logiconlineinc.com](http://www.logiconlineinc.com)



[www.logiconlineinc.com](http://www.logiconlineinc.com) is a resource for:



<http://www-03.ibm.com/systems/z/destinationz/>

# Overview of RIPPLE-TRAC for Source and CICS

[Multi-dimensional separation of concerns](#) is an approach to separation of concerns, supporting construction, evolution and integration of software. Its goals are to enable:

- Encapsulation of all kinds of concerns in a software system, simultaneously.
- Overlapping and interacting concerns.
- On-demand modularization.

*Separation of concerns* is a concept that is at the core of software engineering. It refers to the ability to identify, encapsulate, and manipulate those parts of software that are relevant to a particular concern (concept, goal, purpose, etc.). Concerns are the primary motivation for organizing and decomposing software into manageable and comprehensible parts. Many kinds of concerns may be relevant to different developers in different roles, or at different stages of the software lifecycle. Appropriate separation of concerns has been hypothesized to reduce software complexity and improve comprehensibility; promote traceability; facilitate reuse, non-invasive adaptation, customization, and evolution; and simplify component integration.

The term *multi-dimensional separation of concerns* (MDSOC) refers to flexible and incremental separation, modularization, and integration of software artifacts based on any number of concerns. It overcomes limitations of existing mechanisms by permitting clean separation of multiple, potentially overlapping and interacting concerns simultaneously. MDSOC promotes reuse, improves comprehension, reduces the impact of change, eases maintenance and evolution, improves traceability, and opens the door to system refactoring and reengineering.

MDSOC summary:

Involves decomposition of software according to one or more dimensions of concern. A concern is any piece of concern or focus in a program. [http://en.wikipedia.org/wiki/Separation\\_of\\_concerns](http://en.wikipedia.org/wiki/Separation_of_concerns)

The separation allows:

- To allow people to work on individual pieces of the system in isolation;
- To facilitate reusability;
- To ensure the maintainability of a system;
- To add new features easily;
- To enable everyone to better understand the system;
- To allow support for multi-dimensional separation of concerns.

Remember, a dimension of concern is simply an approach to decomposing, organizing, and structuring software according to concerns of a particular kind. **RIPPLE-TRAC** falls into the realm of multi-dimensional separation of concerns.

***The technology engaged by RIPPLE-TRAC presents the information from the point of view of the concern – the architect only sees information that is related to the concern and has control on what concerns should be brought to the forefront or obscured into the background. The uniqueness of RIPPLE-TRAC is the targeted approach, which we believe has potential to support longer term refactoring of application logic into reusable components and services (SOA). RIPPLE-TRAC can also be used as a support tool for migrating from one platform to another.***

The RIPPLE-TRAC result set helps track and manage numerous cross silo, intricate, legacy component mappings. These mapping support situations where components across applications differ in structure. Manual mappings run the risk of missing critical data in the new/composite system. If missing components are discovered during implementation, the project would be delayed or cancelled.

RIPPLE-TRAC is a batch-driven environment that accepts application libraries as input and provides information on execution flow, physical system component relationships, and dependencies. RIPPLE-TRAC includes the ability to rapidly parse and cross-reference system components across multiple applications into an open repository, produce metrics and reports, and allow analysts to examine and extract information on an ad hoc basis. Results are depicted in a cross-reference list as well as a spreadsheet or any relational model chosen by the analyst.

RIPPLE-TRACs batch analyzer is more conducive to planning activities than an interactive static analysis tool. Planning teams can use metrics and summary-level information to assess the complexity of an application. The more complex a system, the more time it takes to analyze, enhance, improve, or transform. If, for example, a group of programs is highly complex, poorly structured, and utilizes a number of constructs that are hard to decipher, it would increase the time and skills needed to extract business logic from those programs.

## **What RIPPLE-TRAC is not**

RIPPLE-TRAC is not a quick and dirty data migration tool that converts database call structures to embedded SQL. SQL is the standard access mode used to read and update data within a relational database . Such an approach sacrifices comprehensiveness, quality and integrity within the resulting relational database for the sake of time. The result is a database that is poorly designed, limited in its accessibility and flexibility, and detrimental to the business units relying on this data.[1]

# Overview of RIPPLE-TRAC for Source and CICS

We will use a small CICS order entry application as an example of the RIPPLE-TRAC process.

The task is to convert all VSAM access to DB2 access.

## THE PROGRAM FILES AND MAPSETS:

Custinq1.cbl	The customer inquiry
Inqset1.bms	The BMS mapset
Inqset1.cpy	The symbolic map
Custmnt1.cbl	The customer maintenance program trans-id: MNT1
Mntset1.bms	The BMS mapset
Mntset1.cpy	The symbolic map
Custmnt3.cbl	The customer maintenance program with a temporary storage queue, Trans-id: MNT3
Mntset1.bms	The BMS mapset
Mntset1.cpy	The symbolic map
Intedit.cbl	The subprogram that edits integer data
Numedit.cbl	The subprogram that edits numeric decimal data
Syserr.cbl	The generalized error handling program
Dfxxp00a.cbl	The program that produces CICS abends Trans-id: DFXX
Invmenu.cbl	The menu program Trans-id: MENU
Menset1.bms	The BMS mapset
Menset1.cpy	The symbolic map
Custmnt2.cbl	The enhanced customer maintenance program Trans-id: MNT2
Mntset2.bms	The BMS mapset
Mntset2.cpy	The symbolic map
Ordrent.cbl	The order entry program Trans-id: ORD1
Getinv.cbl	The subprogram that retrieves the next invoice number
Ordset1.bms	The BMS mapset
Ordset1.cpy	The programmer-generated symbolic map
Invsum1.cbl	The invoice summary program Trans-id: SUM1
Sumset1.bms	The BMS mapset
Sumset1.cpy	The symbolic map
Custinq2.cbl	The enhanced customer inquiry program Trans-id: INQ2
Inqset2.bms	The BMS mapset
Inqset2.cpy	The symbolic map
Custinq3.cbl	The enhanced customer inquiry program using

Inqset3.bms	alternate indexes, Trans-id: INQ3
Inqset3.cpy	The BMS mapset The programmer-generated symbolic map
Db2inq1.cbl	The DB2 version of the customer inquiry program Trans-id: DIN1
Db2set1.bms	The BMS mapset
Db2set1.cpy	The programmer-generated symbolic map
Cmntp.cbl	The presentation logic portion of the customer maintenance program, Trans-id: CMNT
Cmntb.cbl	The business logic portion of the customer maintenance program
Cmntset.bms	The BMS mapset
Cmntset.cpy	The symbolic map

## **THE COPY MEMBERS:**

Attr.cpy	A copy member for attribute settings
Errparm.cpy	A copy member for error handling values
Custmas.cpy	The record description for the customer master file
Invctl.cpy	The record description for the invoice control file
Invoice.cpy	The record description for the invoice file
Product.cpy	The record description for the product file

## **THE FILES:**

Custmas	Customer master file
Invctl	Invoice control file
Invoice	Invoice file
Product	Product file

## PREREQUISITE

CICS (Customer Information Control System) is an online transaction processing ([OLTP](#)) program from IBM that, together with the [COBOL](#) programming language, has formed over the past several decades the most common set of tools for building customer transaction applications in the world of large [enterprise mainframe](#) computing. A great number of the [legacy applications](#) still in use are COBOL/CICS applications. Using the application programming interface (API) provided by CICS, a programmer can write programs that communicate with online users and read from or write to customer and other records (orders, inventory figures, customer data, and so forth) in a database (usually referred to as "data sets") using CICS facilities rather than IBM's access methods directly.

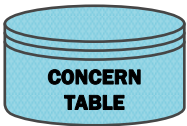
# RIPPLE-TRAC Run Time Behavior For CICS

The intent is to find all references to the COMMAND CODES and map them to their respective business rules as well as any application data names.

Internal Concerns...Concerns that are internal (static) to RIPPLE-TRAC's run time engine:

INCLUDE, INCLUDE(P)	FOR PROCEDURE DIVISION
COPY, COPY(P)	FOR PROCEDURE DIVISION

## EXTERNAL CONCERN TABLE



An EXTERNAL CONCERN TABLE is an element, an element expression, or an aggregate to be passed to RIPPLE-TRAC's run time engine. (INCREMENTS OF up to 500+ in each run)

You should follow the correct CICS syntax when creating this table:

READ

```
>>-READ--FILE(name)--+INTO(data-area)-+----->  
      '-SET(ptr-ref)----'
```

see highlighted code in red in the table below.

**NOTE: ONLY CODE THE COMMAND CODE, NOTHING ELSE**

**IF THE COMMAND CODE HAS A DATA PARM '(???????)', BE SURE TO PUT A PLUS SIGN IN COLUMN 61 SO RIPPLE-TRAC CAN EXTRACT THE DATA OTHERWISE PUT NOTHING IN COLUMN 61.**

```
*****NOMENCLATURE SECTION*****  
****  
**** NOTE: BE SURE NO DUPLICATES EXIST IN COL 1-40 ****  
****  
**** THE NEXT 20 BYTES IS FREE FORM WHATEVER YOU WANT ****  
**** TYPE CODE ALLOWS YOU TO CATAGORISE EACH ARGUMENT ****  
**** EXAMPLES BELOW: ****  
**** TYPE CODE: # = PRECISION ISSUE WITH A CONVERSION ****  
****              T = DB2 TABLE ****  
****              + = GET DATA PORTION OF ANY COMMAND ****  
****              EXAMPLE: ****  
**** EXEC CICS ****  
**** READ FILE('CUSTMAS') ****
```



```

INVCTL-RECORD-KEY F
INVCTL-NEXT-INVOICE-NUMBER F
INV-INVOICE-NUMBER F
INV-INVOICE-DATE F
INV-CUSTOMER-NUMBER F
INV-PO-NUMBER F
INV-PRODUCT-CODE F
INV-QUANTITY F
INV-UNIT-PRICE F
INV-AMOUNT #
INV-INVOICE-TOTAL F
INV-LINE-ITEM F
PRM-PRODUCT-CODE F
PRM-PRODUCT-DESCRIPTION F
PRM-UNIT-PRICE F
PRM-QUANTITY-ON-HAND F
CUSTNO1 F
*****END LANGUAGE SECTION*****
*****DB2 SECTION*****
MMADBV.CUST DB2 TABLE T
*****END DB2 SECTION*****

```

CICS COMMANDS SECTION above refers to the CICS command code that you wish to track.

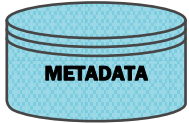
LANGUAGE SECTION above refers to the application data names that you wish to track.

DB2 SECTION above refers to DB2 elements that you wish to track.

If you think a precision issue might occur , then indicate so by coding a '#' in column 61 adjacent to the application field as indicated above for INV-AMOUNT.

The RIPPLE-TRAC repository holds all of output from the **RIPPLE-TRAC** process as a flat file comma delimited.

# METADATA



This optional Metadata table allows you to store the descriptions of all components in a central repository. You need this METADATA to describe the components you are working with.

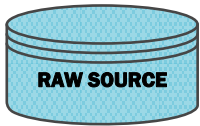
The metadata repository (**up to 500**) is derived from manual input or other feeds.

The metadata table has 4 fields of which two are used for CICS processing.

COMPONENT NAME            28 BYTES  
COMPONENT DESCRIPTION 32 BYTES

```
+-----+ +-----28 bytes-----+ +-----32 bytes-----+ --  
LINK PROGRAM                    INVOKE PROGRAM  
READ DATASET                    READ THE FILE  
READNEXT DATASET                READ NEXT RECORD  
SEND MAP                         SEND THE MAP  
INV-INVOICE-NUMBER               CUSTOMER INVOICE NUMBER
```

# RAW SOURCE



This repository holds all of output from the **RIPPLE-TRAC** process as a flat file and dropped into an excel spreadsheet as depicted below.

## By module:

INDEX	MODULE	TYPE	COMMENTS	BACKREF	COMPONENT	COMPONENT DATA
45	CSTMNTB	*			COPY	CUSTMAS
~0003	CSTMNTB	F		&CUSTMAS	..	CM-CUSTOMER-NUMBER
~0004	CSTMNTB	F		&CUSTMAS	..	CM-FIRST-NAME
~0005	CSTMNTB	F		&CUSTMAS	..	CM-LAST-NAME
~0006	CSTMNTB	F		&CUSTMAS	..	CM-ADDRESS
~0007	CSTMNTB	F		&CUSTMAS	..	CM-CITY
~0008	CSTMNTB	F		&CUSTMAS	..	CM-STATE
~0009	CSTMNTB	F		&CUSTMAS	..	CM-ZIP-CODE
109	CSTMNTB	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
135	CSTMNTB	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
137	CSTMNTB	F				CM-CUSTOMER-NUMBER
175	CSTMNTB	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
185	CSTMNTB	+	SYNONYM FOR DATASET		REWRITE FILE	('CUSTMAS')
185	CSTMNTB	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
222	CSTMNTB	+	SYNONYM FOR DATASET		DELETE FILE	('CUSTMAS')

This information shows the various components via the external Concern table.

Note that the red items in COMPONENT and COMPONENT DATA columns are the items coded in the external Concern table. The remainder of these columns is the data portion of the CICS COMMAND CODE.

- INDEX = The relative pointer in the source code for this line.  
~ indicates the relative pointer in the copy book for this line.
- MODULE = The modules.
- TYPE = The type of component 'F' for field, '+' for extract data portion.
- COMMENT = The assignment for this component, or just a comment.
- BACKREF = The copybook/include that the component is within.
- COMPONENT = The component label name
- COMPONENT DATA = The value of the component label.

The following code snippet relates to the index column above:

```

File Edit Edit_Settings Menu Utilities Compilers Test Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT      SYS2.LOGIC01.SRCLIB.CICS (CSTMNTB) - 01.00          Columns 00001 00072
Command ===>                                           Scroll ===> CSR
000106      1100-READ-CUSTOMER-RECORD.
000107      *
000108      EXEC CICS
000109      READ FILE ('CUSTMAS')
000110      INTO (CUSTOMER-MASTER-RECORD)
000111      RIDFLD (CA-CUSTOMER-NUMBER)
000112      RESP (RESPONSE-CODE)
000113      END-EXEC.

```

```

File Edit Edit_Settings Menu Utilities Compilers Test Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT      SYS2.LOGIC01.SRCLIB.CICS (CSTMNTB) - 01.00          Columns 00001 00072
Command ===>                                           Scroll ===> CSR
000124      SET PROCESS-ERROR TO TRUE
000125      MOVE 'Another user has added a record with that custo
000126      -           'mer number.' TO CA-RETURN-MESSAGE
000127      WHEN OTHER
000128      SET PROCESS-SEVERE-ERROR TO TRUE
000129      PERFORM 9000-SET-ERROR-INFO
000130      END-EVALUATE.
000131      *
000132      2100-WRITE-CUSTOMER-RECORD.
000133      *
000134      EXEC CICS
000135      WRITE FILE ('CUSTMAS')
000136      FROM (CUSTOMER-MASTER-RECORD)
000137      RIDFLD (CM-CUSTOMER-NUMBER)
000138      RESP (RESPONSE-CODE)
000139      END-EXEC.

```

```

File Edit Edit_Settings Menu Utilities Compilers Test Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT      SYS2.LOGIC01.SRCLIB.CICS (CSTMNTB) - 01.00          Columns 00001 00072
Command ===>                                           Scroll ===> CSR
000171      *
000172      3100-READ-CUSTOMER-FOR-UPDATE.
000173      *
000174      EXEC CICS
000175      READ FILE ('CUSTMAS')
000176      INTO (CUSTOMER-MASTER-RECORD)
000177      RIDFLD (CA-CUSTOMER-NUMBER)
000178      UPDATE
000179      RESP (RESPONSE-CODE)
000180      END-EXEC.
000181      *
000182      3200-REWRITE-CUSTOMER-RECORD.
000183      *
000184      EXEC CICS
000185      REWRITE FILE ('CUSTMAS')
000186      FROM (CUSTOMER-MASTER-RECORD)
000187      RESP (RESPONSE-CODE)
000188      END-EXEC.

```

## Another example by module:

INDEX	MODULE	TYPE	COMMENTS	BACKREF	COMPONENT	COMPONENT DATA
21	CUSTINQ1					SEND-DATAONLY-ALARM
29	CUSTINQ1	F				CM-CUSTOMER-NUMBER
30	CUSTINQ1	F				CM-FIRST-NAME
31	CUSTINQ1	F				CM-LAST-NAME
32	CUSTINQ1	F				CM-ADDRESS
33	CUSTINQ1	F				CM-CITY
34	CUSTINQ1	F				CM-STATE
35	CUSTINQ1	F				CM-ZIP-CODE
37	CUSTINQ1	*			COPY	INQSET1
~	CUSTINQ1			&INQSET1	..NO COMPONENTS	
39	CUSTINQ1	*			COPY	DFHAID
68	CUSTINQ1				CICS STATIC LINK	INVMENU
77	CUSTINQ1					SEND-DATAONLY-ALARM
83	CUSTINQ1				TRANSID	INQ1
98	CUSTINQ1					SEND-DATAONLY-ALARM
105	CUSTINQ1	+			RECEIVE MAP	('INQMAP1')
106	CUSTINQ1				MAPSET	INQSET1
121	CUSTINQ1	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
129	CUSTINQ1	F				CM-LAST-NAME
130	CUSTINQ1	F				CM-FIRST-NAME
131	CUSTINQ1	F				CM-ADDRESS
132	CUSTINQ1	F				CM-CITY
133	CUSTINQ1	F				CM-STATE
134	CUSTINQ1	F				CM-ZIP-CODE
155	CUSTINQ1	+			SEND MAP	('INQMAP1')
156	CUSTINQ1				MAPSET	INQSET1
162	CUSTINQ1	+			SEND MAP	('INQMAP1')
163	CUSTINQ1				MAPSET	INQSET1
167	CUSTINQ1					SEND-DATAONLY-ALARM
169	CUSTINQ1	+			SEND MAP	('INQMAP1')
170	CUSTINQ1				MAPSET	INQSET1

This information shows the various components via the external Concern table.

Note that the red items in COMPONENT and COMPONENT DATA columns are the items coded in the external Concern table. The remainder of these columns is the data portion of the CICS COMMAND CODE.

INDEX	=	The relative pointer in the source code for this line. ~ indicates the relative pointer in the copy book for this line. In this case, no components were found.
MODULE	=	The modules.
TYPE	=	The type of component 'F' for field, '*' for copy book or include. '+' for extract data portion.
COMMENT	=	The assignment for this component, or just a comment.
BACKREF	=	The copybook/include that the component is within.
COMPONENT	=	The component label name
COMPONENT DATA	=	The value of the component label.

The following code snippet relates to the index column above:

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT      SYS2.LOGIC01.SRCLIB.CICS (CUSTINQ1) - 01.00      Columns 00001 00072
Command ===>                                          Scroll ===> CSR
000021      88  SEND-DATAONLY-ALARM                      VALUE '3'.
000022      *
000023      01  COMMUNICATION-AREA                          PIC X.
000024      *
000025      01  RESPONSE-CODE                              PIC S9(8)  COMP.
000026      *
000027      01  CUSTOMER-MASTER-RECORD.
000028      *
000029      05  CM-CUSTOMER-NUMBER                        PIC X(6).
000030      05  CM-FIRST-NAME                            PIC X(20).
000031      05  CM-LAST-NAME                             PIC X(30).
000032      05  CM-ADDRESS                               PIC X(30).
000033      05  CM-CITY                                  PIC X(20).
000034      05  CM-STATE                                 PIC X(2).
000035      05  CM-ZIP-CODE                              PIC X(10).
000036      *
000037      COPY INQSET1.
```

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
EDIT      SYS2.LOGIC01.SRCLIB.CICS (CUSTINQ1) - 01.00      Columns 00001 00072
Command ===>                                          Scroll ===> CSR
000102      1100-RECEIVE-CUSTOMER-MAP.
000103      *
000104      EXEC CICS
000105          RECEIVE MAP ('INQMAP1')
000106          MAPSET ('INQSET1')
000107          INTO (INQMAP1I)
000108      END-EXEC.
000109      *
```

Now that you get the general idea, the following examples will depict the decomposition of this ORDER ENTRY application.

## Show all MODULES that use VSAM files with READ PREV:

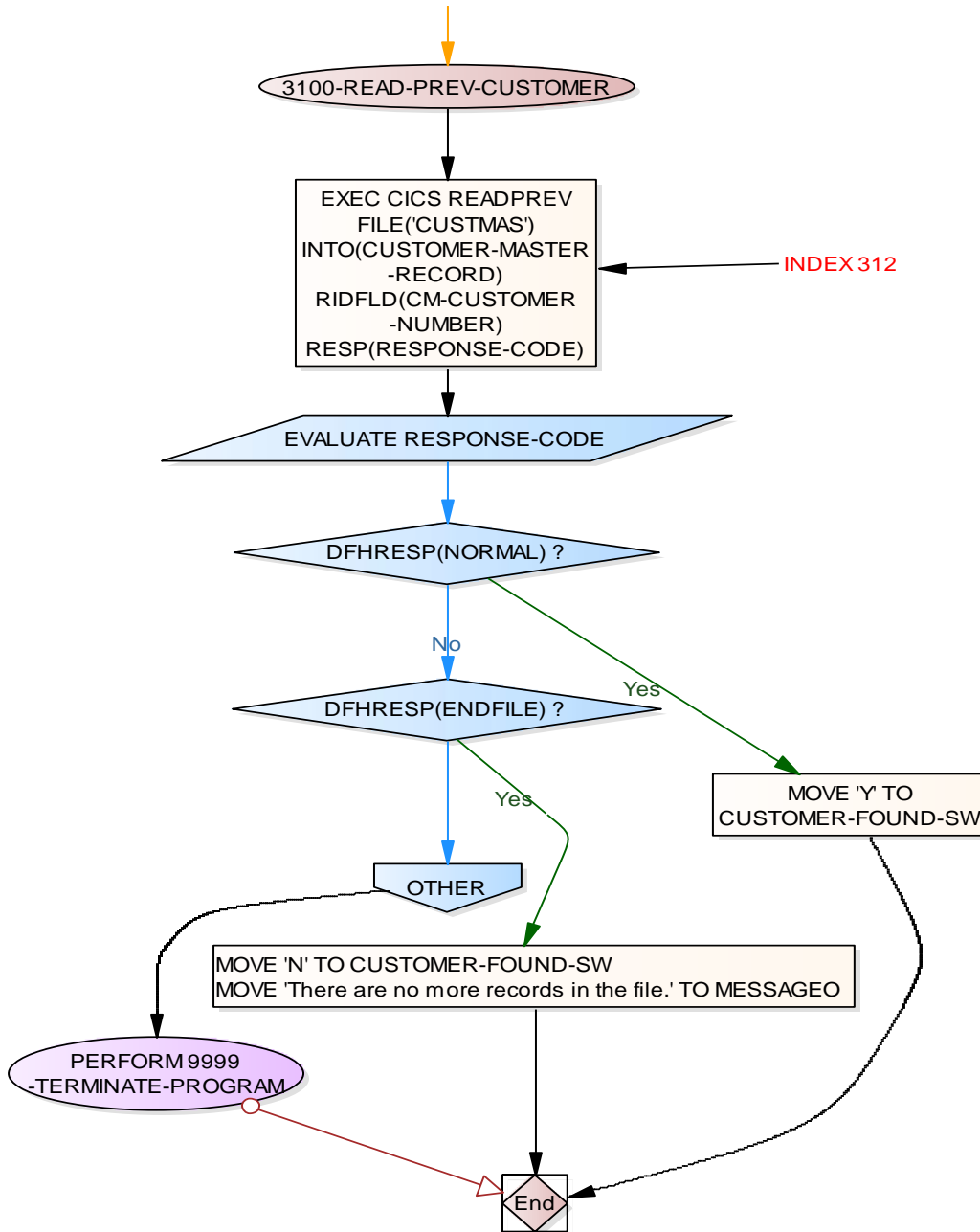
INDEX	MODULE	TYPE	COMMENTS	BACKREF	COMPONENT	COMPONENT DATA
312	CUSTINQ2	+	SYNONYM FOR DATASET		READPREV FILE	('CUSTMAS')
395	CUSTINQ3	+	SYNONYM FOR DATASET		READPREV FILE	('CUSTMAS')

This information shows the various components via the external Concern table.

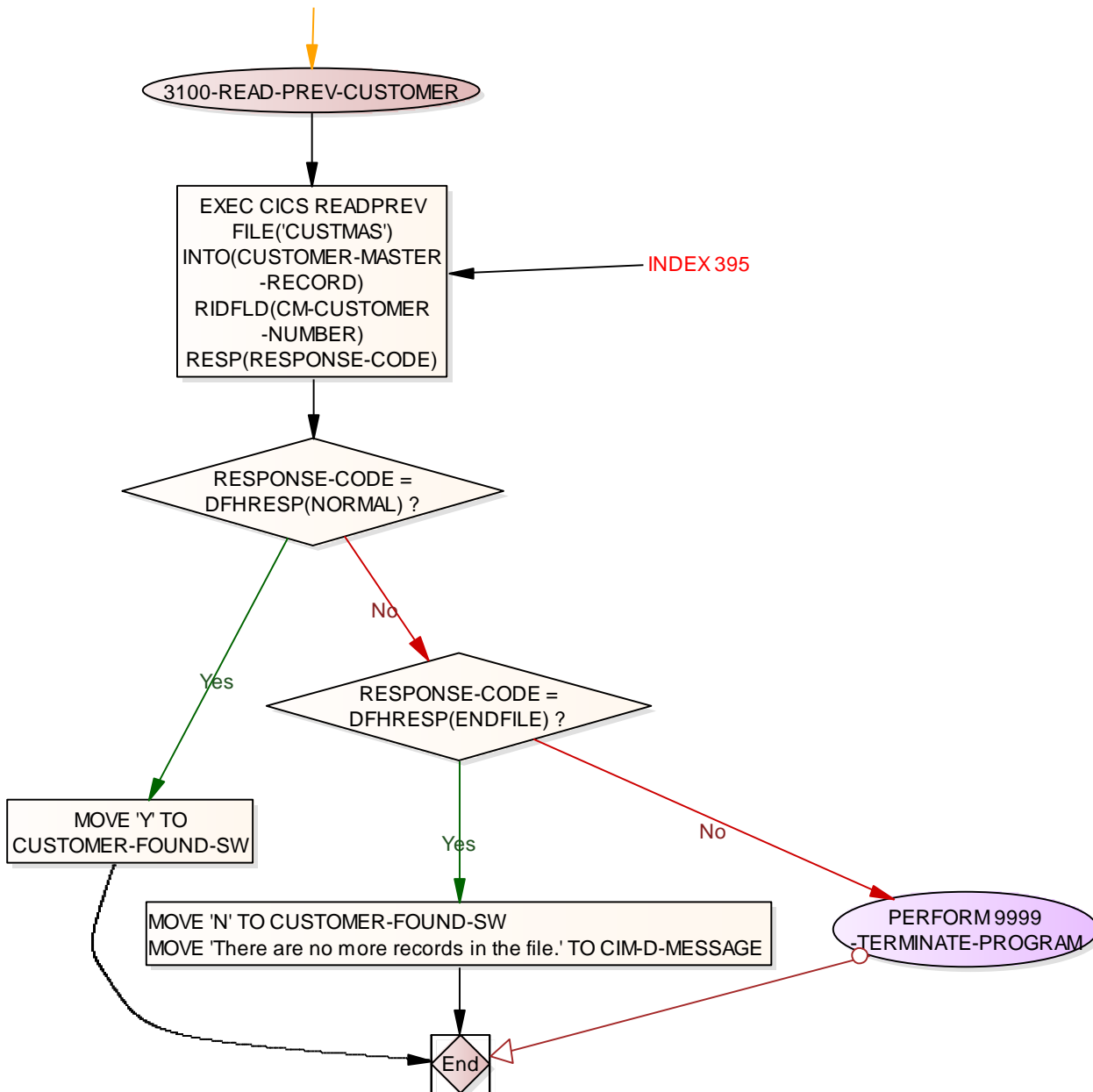
Note that the red items in COMPONENT column are the items coded in the external Concern table.

INDEX	=	The relative pointer in the source code for this line.
MODULE	=	The modules.
TYPE	=	'+' for extract data portion.
COMMENT	=	The assignment for this component, or just a comment.
BACKREF	=	The copybook/include that the component is within. (none)
COMPONENT	=	The component label name
COMPONENT DATA	=	The value of the component label.

Two modules were found, so let's get the business rules surrounding the READPREV COMMAND CODE for module CUSTINQ2 INDEX 312 in above spreadsheet.



Get the business rules surrounding the READPREV  
COMMAND CODE for module CUSTINQ3 INDEX 395 in above spreadsheet.



As you can see, the rules are the same between these two modules.

## Show all MODULES that use file 'CUSTMAS'

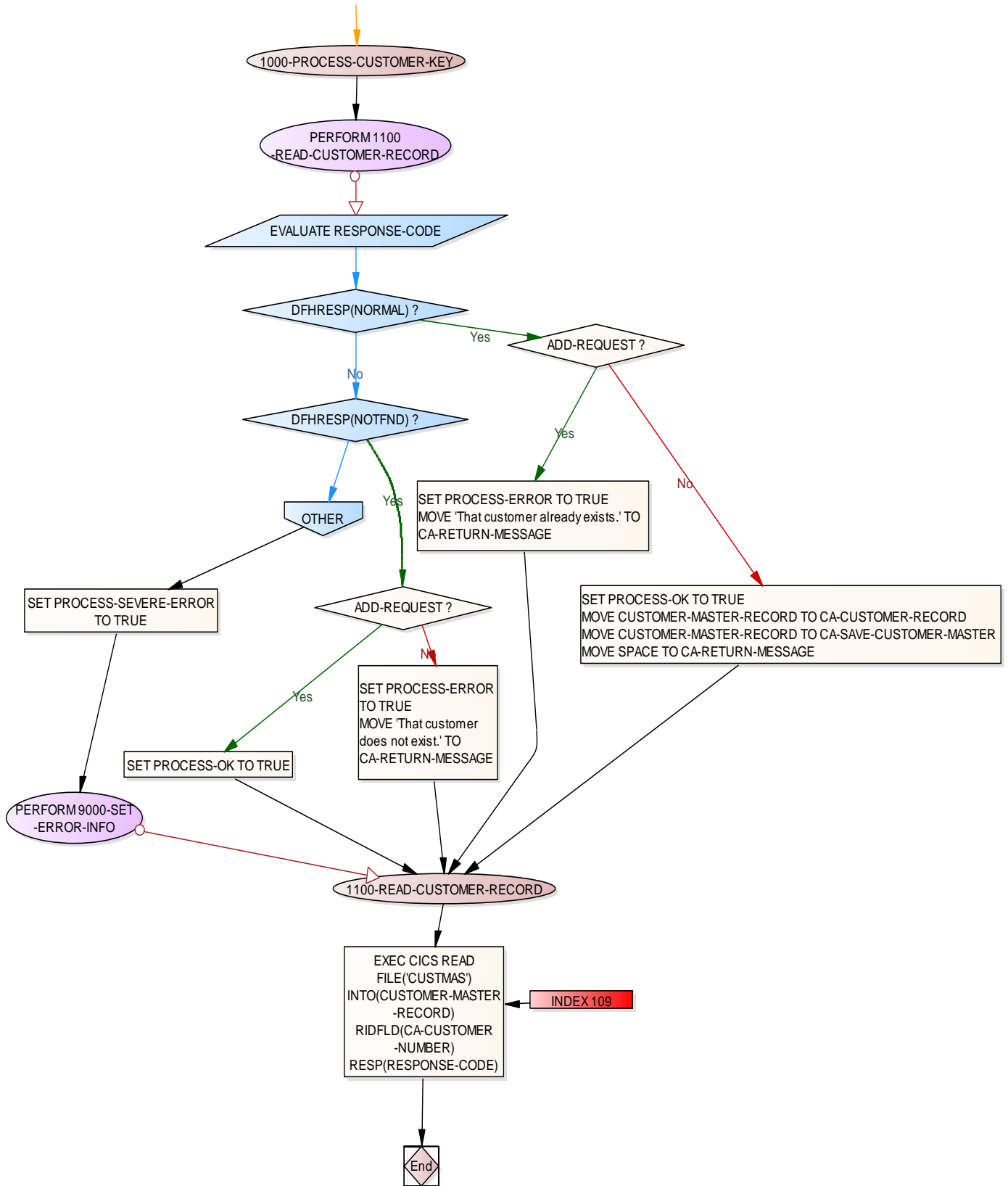
INDEX	MODULE	TYPE	COMMENTS	BACKREF	COMPONENT	COMPONENT DATA
109	CSTMNTB	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
135	CSTMNTB	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
175	CSTMNTB	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
185	CSTMNTB	+	SYNONYM FOR DATASET		REWRITE FILE	('CUSTMAS')
185	CSTMNTB	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
222	CSTMNTB	+	SYNONYM FOR DATASET		DELETE FILE	('CUSTMAS')
121	CUSTINQ1	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
146	CUSTINQ2	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
242	CUSTINQ2	+	SYNONYM FOR DATASET		STARTBR FILE	('CUSTMAS')
263	CUSTINQ2	+	SYNONYM FOR DATASET		READNEXT FILE	('CUSTMAS')
283	CUSTINQ2	+	SYNONYM FOR DATASET		ENDBR FILE	('CUSTMAS')
312	CUSTINQ2	+	SYNONYM FOR DATASET		READPREV FILE	('CUSTMAS')
165	CUSTINQ3	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
324	CUSTINQ3	+	SYNONYM FOR DATASET		STARTBR FILE	('CUSTMAS')
345	CUSTINQ3	+	SYNONYM FOR DATASET		READNEXT FILE	('CUSTMAS')
366	CUSTINQ3	+	SYNONYM FOR DATASET		ENDBR FILE	('CUSTMAS')
395	CUSTINQ3	+	SYNONYM FOR DATASET		READPREV FILE	('CUSTMAS')
255	CUSTMNT1	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
358	CUSTMNT1	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
397	CUSTMNT1	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
417	CUSTMNT1	+	SYNONYM FOR DATASET		REWRITE FILE	('CUSTMAS')
417	CUSTMNT1	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
453	CUSTMNT1	+	SYNONYM FOR DATASET		DELETE FILE	('CUSTMAS')
269	CUSTMNT2	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
446	CUSTMNT2	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
498	CUSTMNT2	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
520	CUSTMNT2	+	SYNONYM FOR DATASET		REWRITE FILE	('CUSTMAS')
520	CUSTMNT2	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
557	CUSTMNT2	+	SYNONYM FOR DATASET		DELETE FILE	('CUSTMAS')
275	CUSTMNT3	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
387	CUSTMNT3	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
431	CUSTMNT3	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')
451	CUSTMNT3	+	SYNONYM FOR DATASET		REWRITE FILE	('CUSTMAS')
451	CUSTMNT3	+	SYNONYM FOR DATASET		WRITE FILE	('CUSTMAS')
492	CUSTMNT3	+	SYNONYM FOR DATASET		DELETE FILE	('CUSTMAS')
299	ORDRENT	+	SYNONYM FOR DATASET		READ FILE	('CUSTMAS')

This information shows 36 components via the external Concern table.

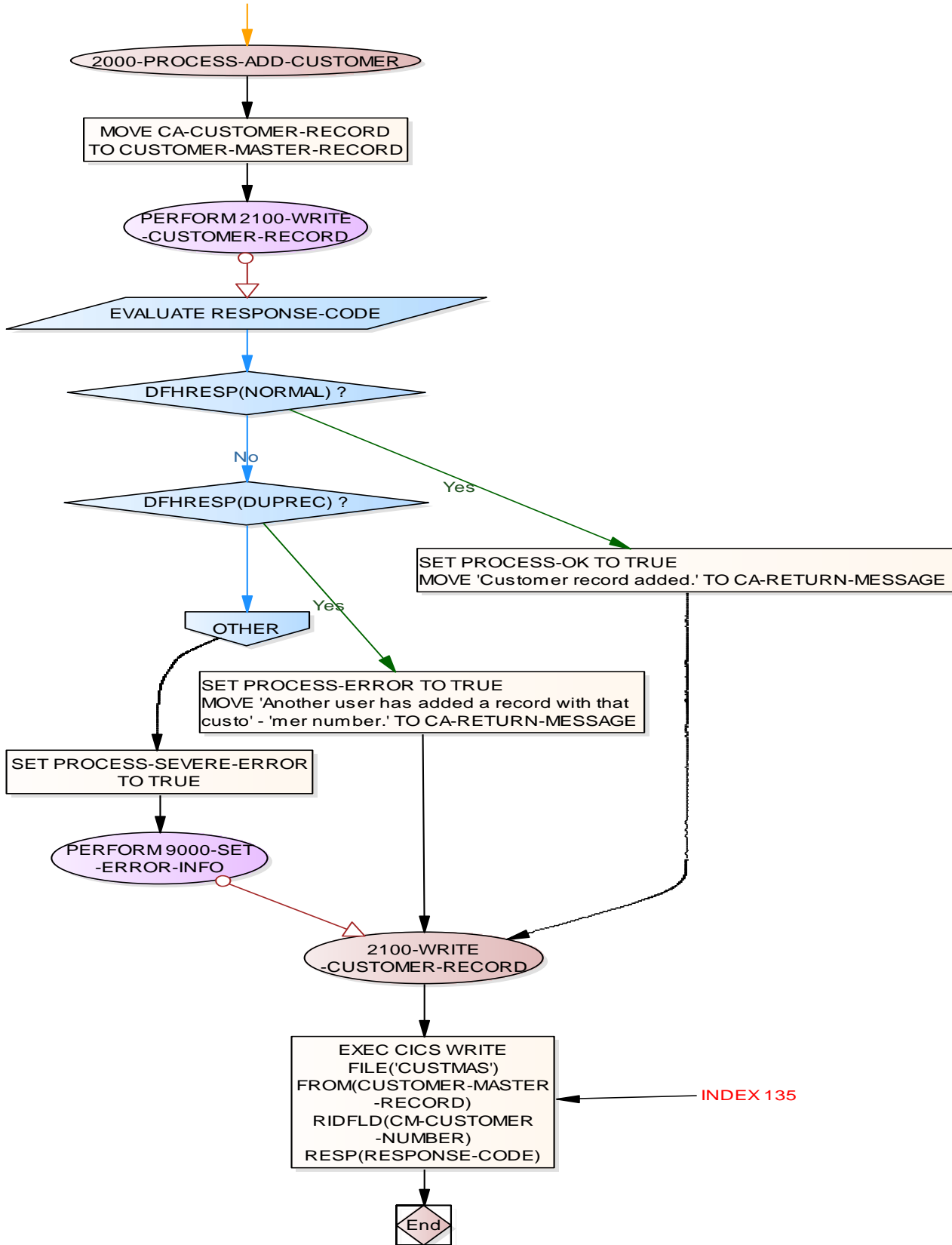
Note that the red items in COMPONENT columns are the items coded in the external Concern table.

INDEX = The relative pointer in the source code for this line.  
MODULE = The modules.  
TYPE = '+' for extract data portion.  
COMMENT = The assignment for this component, or just a comment.  
BACKREF = The copybook/include that the component is within. (none)  
COMPONENT = The component label name  
COMPONENT DATA = The value of the component label.

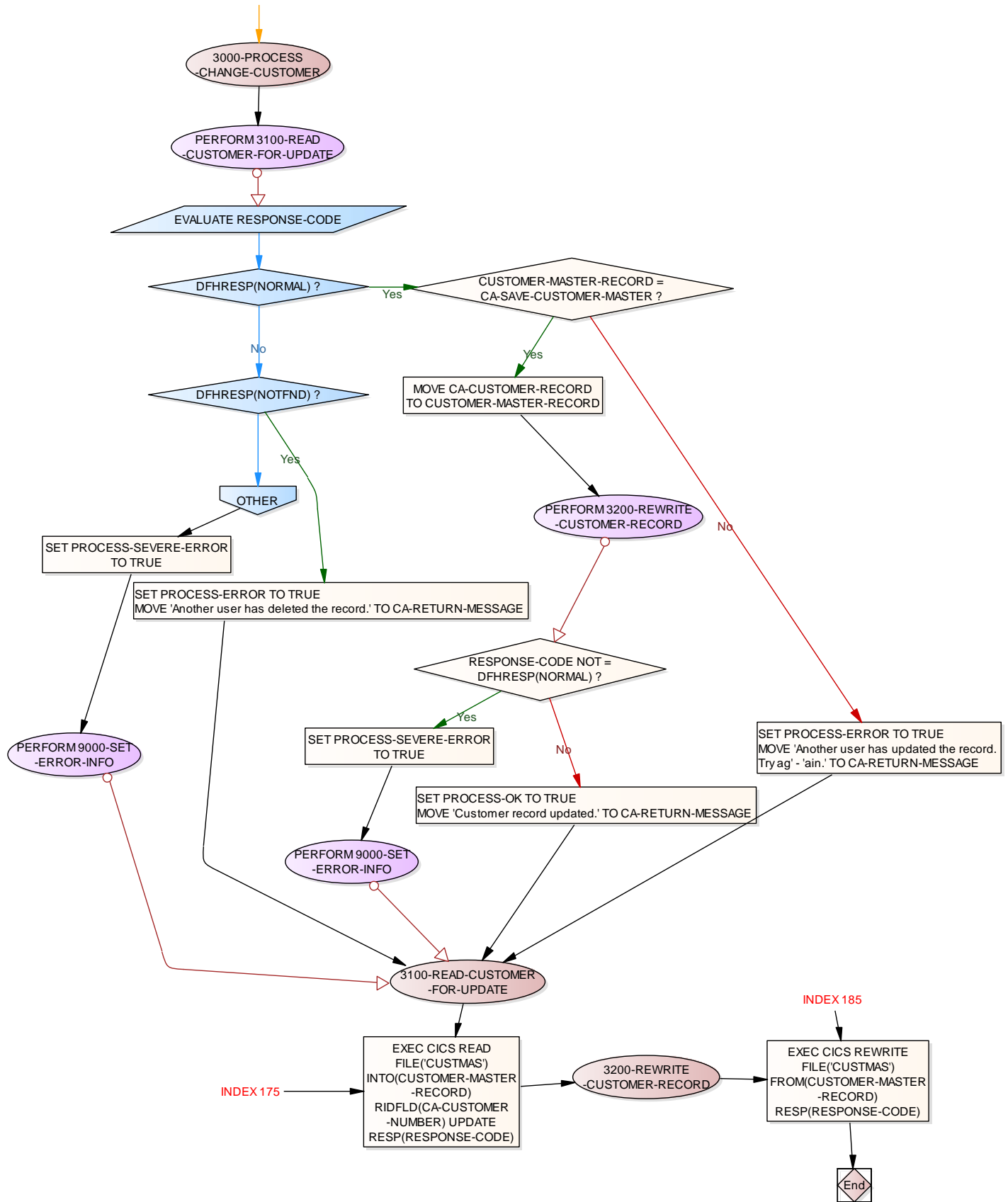
Let's get the business rules surrounding the READ FILE COMMAND CODE for 'CUSTMAS' for module 'CSTMNTB' index 109 in above spreadsheet.



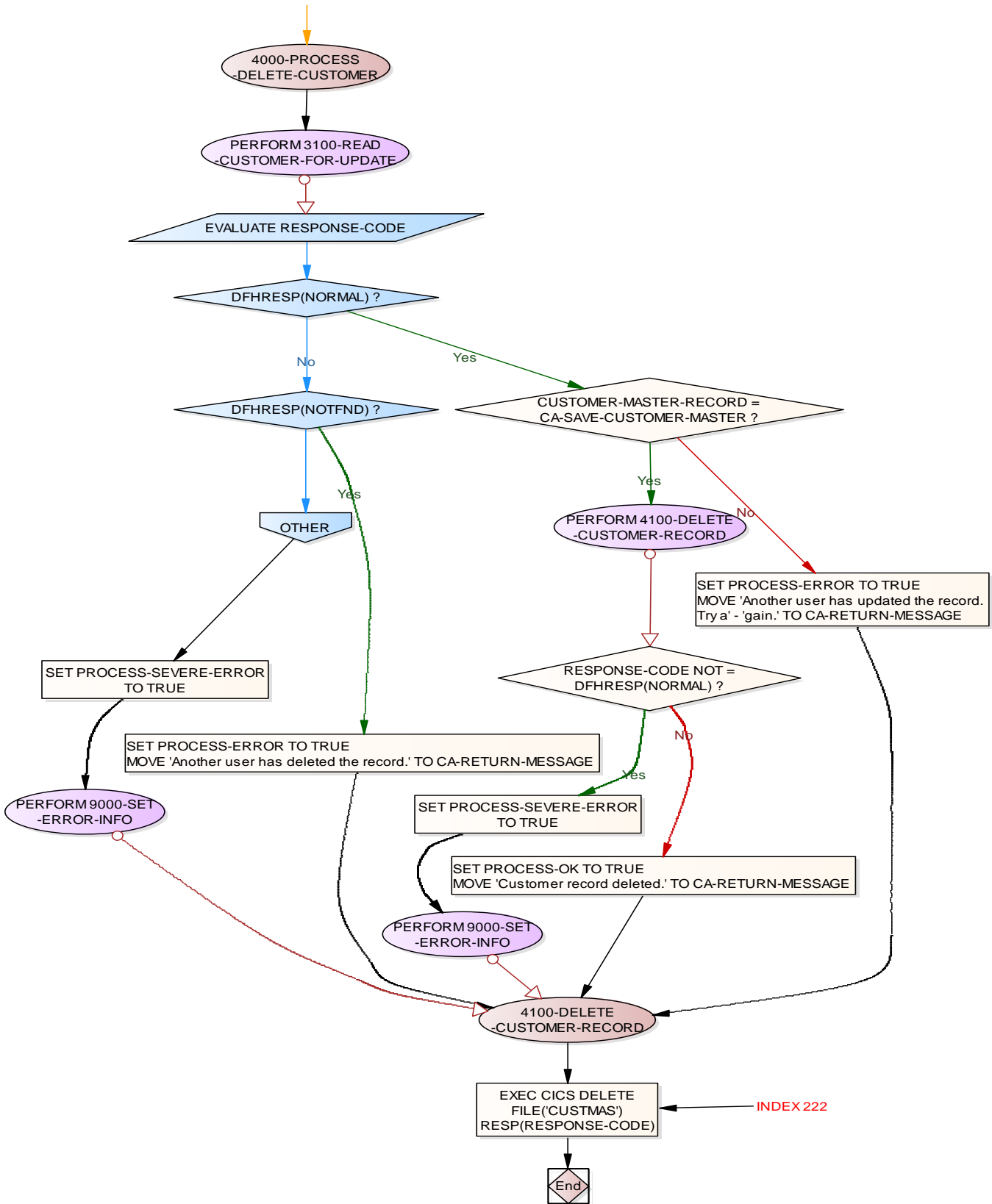
Let's get the business rules surrounding the WRITE FILE COMMAND CODE for 'CUSTMAS' for module 'CSTMNTB' index 135 in above spreadsheet.



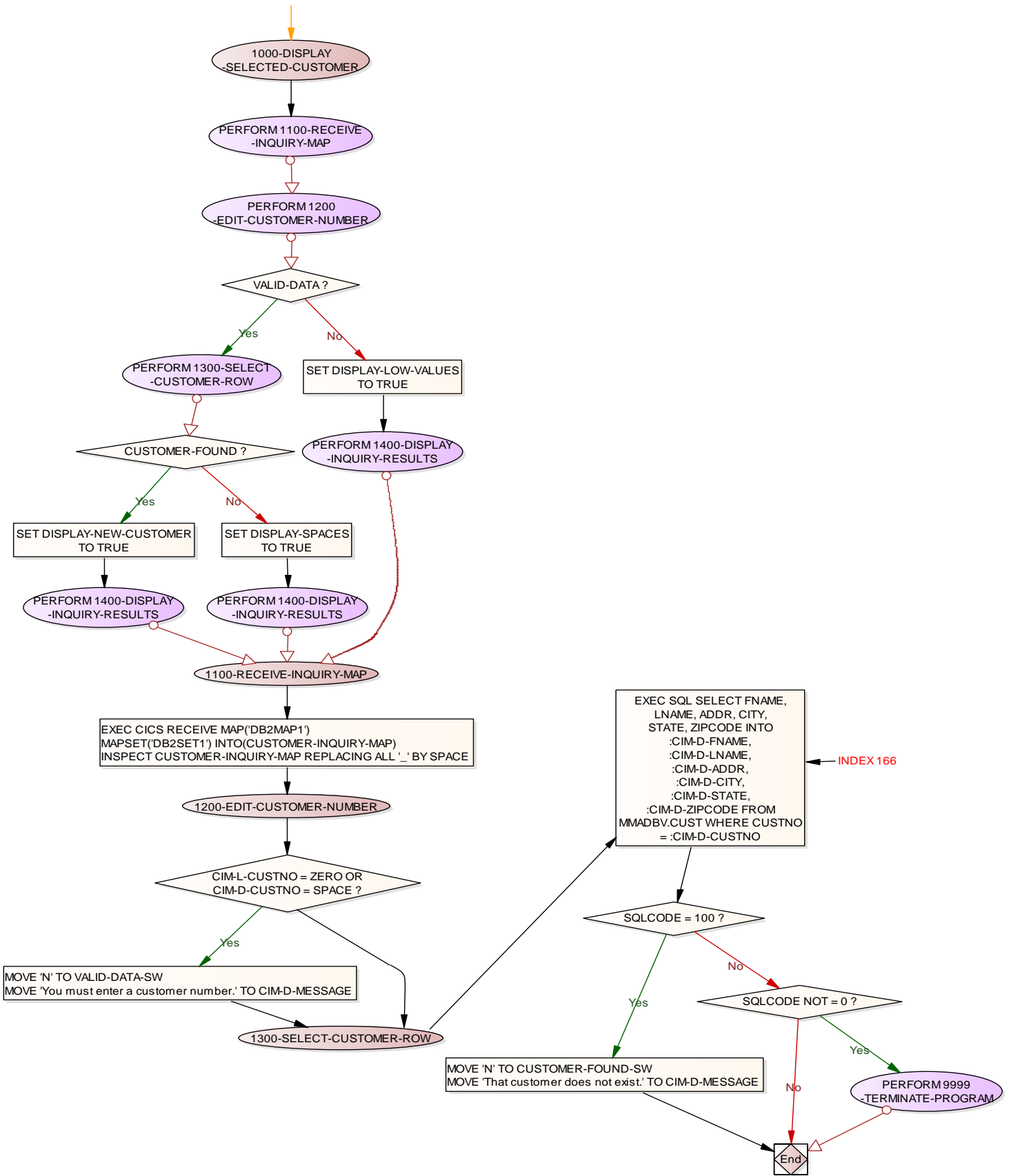
Let's get the business rules surrounding the READ/REWRITE FILE COMMAND CODE for 'CUSTMAS' for module 'CSTMNTB' index 175 AND 185 in above spreadsheet.



Let's get the business rules surrounding the DELETE FILE COMMAND CODE for 'CUSTMAS' for module 'CSTMNTB' index 222 in above spreadsheet.







## Show all MODULES that use CM-LAST-NAME:

INDEX	MODULE	TYPE	COMMENTS	BACKREF	COMPONENT	COMPONENT DATA
~0005	CSTMNTB	F		&CUSTMAS	..	CM-LAST-NAME
31	CUSTINQ1	F				CM-LAST-NAME
129	CUSTINQ1	F				CM-LAST-NAME
~0005	CUSTINQ2	F		&CUSTMAS	..	CM-LAST-NAME
165	CUSTINQ2	F				CM-LAST-NAME
~0005	CUSTINQ3	F		&CUSTMAS	..	CM-LAST-NAME
185	CUSTINQ3	F				CM-LAST-NAME
~0005	CUSTMNT1	F		&CUSTMAS	..	CM-LAST-NAME
161	CUSTMNT1	F				CM-LAST-NAME
351	CUSTMNT1	F				CM-LAST-NAME
410	CUSTMNT1	F				CM-LAST-NAME
~0005	CUSTMNT2	F		&CUSTMAS	..	CM-LAST-NAME
167	CUSTMNT2	F				CM-LAST-NAME
438	CUSTMNT2	F				CM-LAST-NAME
512	CUSTMNT2	F				CM-LAST-NAME
~0005	CUSTMNT3	F		&CUSTMAS	..	CM-LAST-NAME
182	CUSTMNT3	F				CM-LAST-NAME
380	CUSTMNT3	F				CM-LAST-NAME
444	CUSTMNT3	F				CM-LAST-NAME
~0005	ORDRENT	F		&CUSTMAS	..	CM-LAST-NAME
271	ORDRENT	F				CM-LAST-NAME

### 8 modules use CM-LAST-NAME

INDEX	=	The relative pointer in the source code for this line.
MODULE	=	The module.
TYPE	=	'F' Field
COMMENT	=	The assignment for this component, or just a comment.
BACKREF	=	The copybook/include that the component is within.
COMPONENT	=	The component label name
COMPONENT DATA	=	The value of the component label.

In this case the COMPONENT and COMPONENT DATA are one in the same.

## Show all BMS MAPS by Module:

INDEX	MODULE	TYPE	COMMENTS	BACKREF	COMPONENT	COMPONENT DATA
187	CSTMNTP	+			RECEIVE MAP	('CMNTMP1')
358	CSTMNTP	+			RECEIVE MAP	('CMNTMP2')
105	CUSTINQ1	+			RECEIVE MAP	('INQMAP1')
127	CUSTINQ2	+			RECEIVE MAP	('INQMAP2')
145	CUSTINQ3	+			RECEIVE MAP	('INQMAP3')
180	CUSTMNT1	+			RECEIVE MAP	('MNTMAP1')
343	CUSTMNT1	+			RECEIVE MAP	('MNTMAP2')
186	CUSTMNT2	+			RECEIVE MAP	('MNTMAP1')
370	CUSTMNT2	+			RECEIVE MAP	('MNTMAP2')
201	CUSTMNT3	+			RECEIVE MAP	('MNTMAP1')
372	CUSTMNT3	+			RECEIVE MAP	('MNTMAP2')
140	DB2INQ1	+			RECEIVE MAP	('DB2MAP1')
106	INVMENU	+			RECEIVE MAP	('MENMAP1')
225	ORDRENT	+			RECEIVE MAP	('ORDMAP1')

10 modules use BMS maps

INDEX = The relative pointer in the source code for this line.  
 MODULE = The module.  
 TYPE = '+' for extract data portion.  
 COMMENT = The assignment for this component, or just a comment.  
 BACKREF = The copybook/include that the component is within. (none)  
 COMPONENT = The component label name  
 COMPONENT DATA = The value of the component label.

## IN SUMMARY

Software development has seen individual practitioner activities, such as functional and performance testing, become automated. And today's build engineers use timesaving build scripts and tooling. *But more needs to be done to automate the process and steps between roles to improve organizational efficiencies, save money and speed time to market (e.g., RIPPLE-TRAC).*

The **RIPPLE-TRAC** solution can help your team test software more thoroughly and more quickly. Manually analyzing your software applications can result in late releases or inconsistent results. Ad hoc processes for managing your analysis efforts can't ensure the kind of quality you need. By automating your more labor-intensive analysis tasks with **RIPPLE-TRAC**, you can avoid introducing errors into your analysis. The potential payoffs include higher quality and faster time to market—for less money.

Rework is much more costly than building the right solution from the start. To avoid wasteful spending, **RIPPLE-TRAC** is an effective process for defining, capturing, prioritizing and managing modifications to your requirements. **RIPPLE-TRAC** will ensure that IT focuses on the required modifications, so you can deliver what the marketplace is demanding—faster.

## REFERENCES

1. LEGACY SYSTEMS Transformation Strategies by William M. Ulrich  
ISBN 0-13-044927-X

A must read for anyone involved in this realm.