238

CICS update

September 2005

In this issue

A xephon PUBLICATION

Published by

Xephon Inc
PO Box 550547
Dallas, Texas 75355
USA

Phone: 214-340-5690
Fax: 214-341-7081

Editor

Trevor Eddolls
E-mail: trevore@xephon.com

Publisher

Colin Smith
E-mail: info@xephon.com

Subscriptions and back-issues

A year's subscription to *CICS Update*, comprising twelve monthly issues, costs $270.00 in the USA and Canada; £175.00 in the UK; £181.00 in Europe; £187.00 in Australasia and Japan; and £185.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the December 2000 issue, are available separately to subscribers for $24.00 (£16.00) each including postage.

CICS Update on-line

Code from *CICS Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/cics; you will need to supply a word from the printed issue.

Contributions

When Xephon is given copyright, articles published in *CICS Update* are paid for at the rate of $160 (£100 outside North America) per 1000 words and $80 (£50) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of $32 (£20) per 100 lines. To find out more about contributing an article, without any obligation, please download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

CICS access to external LDAP server

INTRODUCTION

I was asked to provide a quick solution so that CICS application programs could access an external LDAP server. I found a solution using a mixture of JCICS classes and a traditional JNDI package.

This article illustrates the use of JCICS when searching an external LDAP server with a single filter.

SAMPLE PROGRAM FUNCTIONS

LOOKUP is a small Java program that receives a COMMAREA containing a TS queue name (the first eight bytes) and an LDAP query (ie uid=myid).

It searches the LDAP server using a standard Java package, with no client authentication, building a search context with the criteria specified, and writes the attributes/values pairs returned to the TS queue specified in the COMMAREA.

JCICS is used to:

- Create a new instance of the TSQ CLASS with the name from the COMMAREA.
- Write to the output TDQUEUE using a new instance of the TDQ CLASS.
- Access the COMMAREA with the CommAreaHolder Object received from arguments in the main routine.

ENVIRONMENT SET UP AND RUNNING

Perform the following steps to set up the environment and execute the program:

1 Make sure the environment variables CICS_HOME,

JAVA_HOME, and CLASSPATH are correctly set.

2 Compile LOOKUP.java:

```
javac LOOKUP.java
```

3 Define the CICS program:

```
DEF PROG("program name") G("group name") JVM(YES) JVMCLASS(LOOKUP)
```

4 You can now link the program with:

```
CICS LINK PROG("Program name") C(&COM) LENGTH(100).
```

With PF5 create the &COM variable with:

```
"My TS name"(UID="id to search").
```

5 Check CSSL for messages:

```
 CECI 6:59:28 JNDI0001I Lookup Starting ..

CECI 6:59:28 JNDI0005I ts:"my ts name" search for (UID="id to search")

 CECI 6:59:28 JNDI0006I Searching......

 CECI 6:59:35 JNDI0007I End of request
```

6 Use CEBR to browse the output TSQ:

```
 Dn:cn=name,ou=XXX,ou=XXX,o=XXX,c=us

 attr:preferredRfc822Recipient

 val :name@name.com

 attr:preferredX400Originator

 val :/G=/S=/P=/A=/C=/

 ..
```

For each attribute (attr), the available values (val) are printed.

REFERENCES

- *Java Applications in CICS*, SC34-6000-00. See the chapter relating to building sample JCICS programs in order to set variables.

- *CICS System Definition Guide*, SC34-5988-00.

LOOKUP

```
//  Function :
//  ==========
//  anonymous search external LDAP server with a SUBTREE scope .
//  Input :
//  =======
//  Receive a 100 length commarea :
//    (return TSQ name) length 8 + FILTER LDAP max 92 => 100
//  Customization
//  =============
//  Find Xxx and update LDAP server url
//
//  Output :
//  ========
//  Output messages in CSSL.
//  Critical error messages in System.err
//
//  Results:
//  ========
//  Pairs of Attribute/Values returned in TS
//
import java.net.*;
import java.io.*;
import javax.naming.*;
import javax.naming.directory.*;
import java.util.Hashtable;
import java.util.Enumeration;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.lang.*;
import com.sun.jndi.ldap.LdapClient;
import com.ibm.cics.server.*;
// =======================================
// LOOKUP CLASS                          =
// =======================================
public class LOOKUP
{
public static String TS_Name   = "        ";
public static String MY_FILTER = "(uid=XXXXX)  ";
public static String MSG       = "             ";
public static String TRAN      = "             ";
public static    TDQ tdq;
public static    TSQ tsq;
    public static void main(CommAreaHolder COM)
    {
    tdq=new TDQ();
```

```
    tdq.setName("CSSL");
    tsq=new TSQ();
        Task t = Task.getTask();
        if ( t == null )
           {System.err.println("lookup Can't get Task");
            return;}
        else
         {
            TRAN = t.getTransactionName();
            displayMsg("JNDI0001I lookup Starting ..");
            };
// ===============================================
// Commarea checks
// ===============================================
    if (COM.value.length == 0) {
    displayMsg("JNDI0002E no commarea availaible  ");
    return; }
    if (COM.value.length < 100) {
    displayMsg("JNDI0003E commarea length is < 100");
    return; }
    byte[] filterbytes;
    byte[] mycom=COM.value;
    ByteArrayInputStream mycomis = new ByteArrayInputStream(mycom);
    DataInputStream mycomdata = new DataInputStream(mycomis);
    filterbytes = new byte[92];
      try {
           int len = (byte)mycomdata.read(filterbytes); }
               catch (java.io.IOException e)
          {displayMsg("JNDI0004E Access to commarea has failed : " + e);
                return;                                              }
// ===============================================
// Get filter and TSQ name
// ===============================================
    MY_FILTER = new String(filterbytes);
    TS_Name   =  MY_FILTER.substring(0,8);
    MY_FILTER = MY_FILTER.substring(8,90);
    displayMsg("JNDI0005I ts:"+TS_Name+" search:"+MY_FILTER);
    tsq.setName(TS_Name);
    MSG= "JNDI0006I Searching......"+MY_FILTER;
    displayMsg(MSG);
// ===============================================
// LDAP server access
// ===============================================
      Hashtable env = new Hashtable(11);
      env.put(Context.INITIAL_CONTEXT_FACTORY,
          "com.sun.jndi.ldap.LdapCtxFactory");
      env.put(Context.PROVIDER_URL, "ldap://xxx.xxx.org:389");
     try {
          DirContext ctx = new InitialDirContext(env);
     SearchControls constraints = new SearchControls();
```

```
     constraints.setSearchScope(SearchControls.SUBTREE_SCOPE);
     NamingEnumeration results = ctx.search("",MY_FILTER, constraints);
         if (!results.hasMore())
             {
    displayMsg("JNDI0010E no data returned");
     displayMsg("JNDI0007I End of request");
               return;
             };
// ================================================
// Process Attributes/Values pairs
// ================================================
     while (results != null && results.hasMore())
      {
         SearchResult sr = (SearchResult) results.next();
         String dn = sr.getName();
        writeMsg("Dn:" + dn);
         Attributes attrs = sr.getAttributes();
      for (NamingEnumeration ne = attrs.getAll(); ne.hasMoreElements();)
              {
                  Attribute attr = (Attribute) ne.next();
                  String attrID = attr.getID();
                  writeMsg("attr:"+attrID);
               for (Enumeration vals = attr.getAll();
vals.hasMoreElements();  )
                 {
                    String values;
                   values =  (String) vals.nextElement();
                    writeMsg("val :"+values);
                 }
              }
      }
        displayMsg("JNDI0007I end of request");
      } catch (NamingException e) {
          displayMsg("JNDI0008E Invalid LDAP request : " + e);
     displayMsg("JNDI0007I end of request"); }

    }
//}
// ================================================
// Display Output Msgs on CSSL
// ================================================

private static boolean displayMsg(String msg)
{
int h,m,s;
String cicsmsg;
String MSGUSR;
Calendar cal = new GregorianCalendar(); // cal is set to current time
h = cal.get(Calendar.HOUR);
m = cal.get(Calendar.MINUTE);
```

```
s = cal.get(Calendar.SECOND);
cicsmsg = TRAN + " " + h +":"+m+":"+s+" " + msg;
if (cicsmsg.length() > 128 )
{MSGUSR  = cicsmsg.substring(0,128);}
else
{MSGUSR  = cicsmsg;};
    try {
    tdq.writeData(MSGUSR.getBytes());}
                catch (Throwable x) {
               System.out.println("Problem WRITE TDQ    =
"+x.toString());
                return false;                                    }
return true;
}
// ====================================================
// Write in TS
// ====================================================
private static boolean writeMsg(String msg)
{
int h,m,s;
String cicsmsg;
Calendar cal2= new GregorianCalendar(); // cal is set to current time
h = cal2.get(Calendar.HOUR);
m = cal2.get(Calendar.MINUTE);
s = cal2.get(Calendar.SECOND);
cicsmsg = msg;
    try {
    tsq.writeItem(cicsmsg.getBytes());}
                catch (Throwable x) {
               System.out.println("Write TS problem   = "+x.toString());
                return false;                                    }
return true;
}
}
```

David Harou
Systems Programmer (France)

If you have ever overcome any difficulties with CICS, or made an interesting discovery, you could receive a cash payment simply by telling us about it.

More information about contributing an article, plus an explanation of our terms and conditions, can be found at www.xephon.com/nfc. Articles can be sent to the editor, Trevor Eddolls, at trevore@xephon.com.

Close and re-open a CICS VSAM dataset

DESCRIPTION

This program will close or open a file in CICS. It takes the dataset name to use from a CICS terminal or a console device.

The application is useful, for example, when you need to deallocate and then reallocate a CICS dataset for batch processing in a long-running CICS session.

The log information looks like the following:

```
23.30.40 JOB26726  +CSOPCP-01 FCLOSE CLOSED    ACTION=C
23.30.40 JOB26726  +CSOPCP-99 DSN: DnameVS
23.31.03 JOB26726  +CSOPCP-07 FOPEN  OPENED    ACTION=O
23.31.03 JOB26726  +CSOPCP-99 DSN: DnameVS
```

CODE

```
*ASM XOPTS(CICS,SP)
*********************************************************************
*  AUTHOR:          PRIVATE                                         *
*  PROGRAM NAME :   CSOPCP                                          *
*  FUNCTION:        THIS IS A PROGRAM TO CLOSE OR OPEN A FILE IN CICS  *
*                   BASED ON THE DSNAME COMING IN FROM A CICS TERMINAL *
*                   OR FROM A CONSOLE DEVICE                        *
*                                                                   *
*  INPUT:           FROM CICS TERMINAL: 'TRAN DATASETNAME'          *
*                   FROM CONSOLE: '/F APPLID,TRAN X DATASETNAME'    *
*                                 WHERE X MUST BE C(LOSE) OR O(PEN)    *
*                                                                   *
*  OUTPUT:          WTO: '+OPCPFCLOSE TEXT'  ON CICS- AND SYSTEMLOG    *
*                                                                   *
*  SUPPORTED CICS VERSIONS:                                         *
*                                                                   *
*  CTS 2.3 (AND LOWER)                                              *
*                                                                   *
*********************************************************************
         EJECT
*
*   EXPAND THE DFHEISTG FOR THE REQUIRED USER FIELDS
*
         SPACE
DFHEISTG DSECT
```

```
WORKSTGX DS    ØF
OPENSTAT DS    XL4
RESPCLOS DS    XL4
RESPINQ  DS    XL4
LEN      DS    XL2
WORKSTG  DS    ØF
APPLID   DS    CL8
IDSNAME  DS    CL44
DSNAME   DS    CL44
FILENAME DS    CL8
SYSID    DS    CL4
INPUT    DS    CL8Ø
ACTIONCH DS    CL1
OTEXT    DS    CL35
         ORG   *-35
OFINMSG  DS    CL72
         EJECT
*************************************************************************
*     MAIN PROGRAM                                                      *
*************************************************************************
         SPACE
CSOPCP   CSECT
CSOPCP   AMODE 31
CSOPCP   RMODE ANY
         B     START
         SPACE
PROGNAME DC    CL8'CSOPCP  '       SET
         SPACE
BEGIN    DS    ØH
         SPACE
*********************************************************************
*   PROCESS TERMINAL INPUT                                          *
*********************************************************************
         SPACE
PROCESS  DS    ØH
         SPACE
* ---------------------------------------------------------------*
         EXEC CICS RECEIVE SET(R5) LENGTH(LEN)
* ---------------------------------------------------------------*
*         INPUT AREA SHOULD BE                                    *
*         XACT BLANK ACTION-CHAR BLANK DSNAME (MAX. 44 CHARS)     *
*         Ø-3  4     5           6     7 ------------> 5Ø         *
* ---------------------------------------------------------------*
         SPACE
         MVC   ACTIONCH,5(R5)      SAVE ACTION CHARACTER
         LH    R6,LEN              LOAD REAL RECEIVE LENGTH
         CH    R6,=X'ØØØ7'         COMPARE IF RECEIVED LENGTH > 7
         BL    INPUTERR            NO SEND ERROR WTO
         LH    R7,PREFIX           LOAD PREFIX LENGTH
         SR    R6,R7               GET CORRECT DATA LENGTH OF DSN
```

```
         STH   R6,LEN              AND STORE IT BACK
         LA    R5,5(R5)            POINT TO INPUT DATA - PART1
         CLI   Ø(R5),C'C'          CLOSE REQUESTED?
         BE    PROCE1ØØ            YES: GO TO PROCE1ØØ
         CLI   Ø(R5),C'O'          OPEN REQUESTED?
         BNE   INVACTIO            NO: ERROR. GO TO PROCE1ØØ
         SPACE
PROCE1ØØ DS    ØH
         LA    R5,2(R5)            POINT TO INPUT DATA - PART2
         LA    R7,DSNAME           POINT TO OUTPUT FIELD
         BCTR  R6,Ø                SUBTRACT ONE FROM LENGTH
         EX    R6,TEXTMOVE         MOVE DSNAME TO DSNAME FIELD
         SPACE
* ------------------------------------------------------------*
         EXEC CICS INQUIRE FILE START
         SPACE
INQLOOP  DS    ØH
         EXEC CICS INQUIRE FILE(FILENAME) DSNAME(IDSNAME) NEXT        *
                   RESP(RESPINQ) OPENSTATUS(OPENSTAT)
         LA    R7,RESPINQ           POINT TO RESP
         CLC   Ø(4,R7),DFHRESP(END) IS  END CONDITION
         BE    NOTFOUND             NOT FOUNDMESSAGE
         SPACE
* ------------------------------------------------------------*
*        CHECK THE DSNAME
* ------------------------------------------------------------*
         SPACE
         LH    R6,LEN              LOAD REAL RECEIVE DSNAME LENGTH
         LA    R5,DSNAME           POINT TO RECEIVED DSNAME
         LA    R7,IDSNAME          POINT TO INQ DSNAME FIELD
         BCTR  R6,Ø                SUBTRACT ONE FROM LENGTH
         EX    R6,COMPARE          COMPARE DSNAMES
         BNE   INQLOOP             GET AND CHECK NEXT
         SPACE
* ------------------------------------------------------------*
*        DSNAME FOUND CLOSE OR OPEN DATASET
* ------------------------------------------------------------*
         SPACE
         EXEC CICS INQUIRE FILE END
         LA    R7,OPENSTAT         POINT TO OPENSTATUS
         CLI   ACTIONCH,C'C'       CLOSE REQUESTED ?
         BNE   REOPEN              NO: GO TO REOPEN
         CLC   Ø(4,R7),DFHVALUE(OPEN) IS WANTED FILE OPEN ?
         BNE   NOTOPEN                NO, NO CLOSE REQUIRED
         EXEC CICS SET FILE(FILENAME) CLOSED RESP(RESPCLOS)
         LA    R7,RESPCLOS            POINT TO RESP
         CLC   Ø(4,R7),DFHRESP(NORMAL) FILE CLOSED CORRECTLY ?
         BNE   CLOSERR                NO, ERROR
         SPACE
* ------------------------------------------------------------*
```

```
*        WTO FILE CLOSED
* ---------------------------------------------------------------*
         SPACE
         MVC   OTEXT,OKTEXTC
         B     RETURN              AND LEAVE
         SPACE
* ---------------------------------------------------------------*
*        DSNAME FOUND OPEN DATASET
* ---------------------------------------------------------------*
         SPACE
REOPEN   DS     ØH
         CLC   Ø(4,R7),DFHVALUE(CLOSED) IS WANTED FILE OPEN ?
         BNE   NTCLOSED                NO, NO CLOSE REQUIRED
         EXEC CICS SET FILE(FILENAME)   OPEN RESP(RESPCLOS)
         LA    R7,RESPCLOS             POINT TO RESP
         CLC   Ø(4,R7),DFHRESP(NORMAL)  FILE CLOSED CORRECTLY ?
         BNE   OPENERR                 NO, ERROR
         SPACE
* ---------------------------------------------------------------*
*        WTO FILE CLOSED
* ---------------------------------------------------------------*
         SPACE
         MVC   OTEXT,OKTEXTO
         B     RETURN              AND LEAVE
         SPACE
*****************************************************************
INVACTIO DS    ØH
         MVC   OTEXT,INVACHAR
         B     RETURN              AND LEAVE
         SPACE
*****************************************************************
INPUTERR DS    ØH
         MVC   OTEXT,INPERR
         B     RETURN              AND LEAVE
         SPACE
*****************************************************************
CLOSERR  DS    ØH
         MVC   OTEXT,CLOERR
         B     RETURN              AND LEAVE
         SPACE
*****************************************************************
OPENERR  DS    ØH
         MVC   OTEXT,OPEERR
         B     RETURN              AND LEAVE
         SPACE
*****************************************************************
NOTOPEN  DS    ØH
         MVC   OTEXT,NOTOPE
         B     RETURN              AND LEAVE
         SPACE
```

```
*********************************************************************
NTCLOSED DS    ØH
         MVC   OTEXT,NOTCLOSE
         B     RETURN              AND LEAVE
         SPACE
*********************************************************************
NOTFOUND DS    ØH
         MVC   OTEXT,NOTFND
         SPACE
*********************************************************************
RETURN   DS    ØH
         MVC   OTEXT+34(L'OTEXT-34),ACTIONCH
         EXEC CICS WRITE OPERATOR TEXT(OTEXT) TEXTLENGTH(TLEN)
         MVC   OFINMSG,FINMSG
         MVC   OFINMSG+15(L'IDSNAME),IDSNAME
         MVC   OFINMSG+64(L'FILENAME),FILENAME
         EXEC CICS WRITE OPERATOR TEXT(OFINMSG) TEXTLENGTH(FLEN)
         EXEC CICS RETURN
         SPACE
*********************************************************************
START    DS    ØH
         SPACE
*********************************************************************
*  PRELOAD DFHEISTG USER FIELDS                                     *
*********************************************************************
         MVC   WORKSTGX,NULLS    INITIALIZE
         MVC   WORKSTG,BLANKS    INITIALIZE
         B     BEGIN
         EJECT
*********************************************************************
*        CONSTANTS                                                  *
*********************************************************************
         SPACE
COMPARE  DS    ØF
         CLC   Ø(Ø,R7),Ø(R5)        EXECUTE COMPARE OF DSNAMES
TEXTMOVE DS    ØF
         MVC   Ø(Ø,R7),Ø(R5)        EXECUTE MOVE OF DSNAME
         SPACE
BLANKS   DC    CL188' '
NULLS    DC    XL14'ØØ'
PREFIX   DC    XL2'ØØØ7' CONTAINS XACT(4) + BLANK + ACTIONCHAR + BLANK
         SPACE
TLEN     DC    XL4'ØØØØØØ23'
FLEN     DC    XL4'ØØØØØØ48'
         SPACE
OKTEXTC  DC    CL35'CSOPCP-Ø1 FCLOSE CLOSED    ACTION=N'
INPERR   DC    CL35'CSOPCP-Ø2 INPUTERR         ACTION=N'
CLOERR   DC    CL35'CSOPCP-Ø3 FCLOSE ERROR     ACTION=N'
NOTFND   DC    CL35'CSOPCP-Ø4 FILE NOTFOUND    ACTION=N'
NOTOPE   DC    CL35'CSOPCP-Ø5 FCLOSE NOTOPEN   ACTION=N'
```

```
NOTCLOSE DC    CL35'CSOPCP-Ø6 FCLOSE NOTCLOSED ACTION=N'
OKTEXTO  DC    CL35'CSOPCP-Ø7 FOPEN  OPENED    ACTION=N'
OPEERR   DC    CL35'CSOPCP-Ø8 FOPEN  ERROR     ACTION=N'
INVACHAR DC    CL35'CSOPCP-Ø9 INVALID ACTION   ACTION=N'
FINMSG   DC    CL72'CSOPCP-99 DSN: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX*
               XXXXXXXX DD: XXXXXXXX'
         SPACE
         EQUREG                    REGISTER EQUATES
         SPACE
         LTORG
         SPACE
         DC    C' '
         END   CSOPCP
```

Dieter Maier
CICS Systems Programmer (Germany)

CICS TS 3.1 – modern-day COMMAREAs

CICS programs have always used the communication area (COMMAREA) to exchange data between themselves, and the 32KB limit of the COMMAREA has been a major constraint, especially in handling distributed application requirements. IBM has now come up with channels and containers as an alternative to the COMMAREA in CICS TS 1.3. In addition to the fact that channels and containers can be used to transfer amounts of data larger than 32KB, there are also other advantages. According to IBM, channels are modern-day COMMAREAs that are meant to meet the demands of the new era.

WHAT ARE CHANNELS AND CONTAINERS?

A container is a named block of data that is used to pass information between programs (it can be thought of as a named COMMAREA). Any number of containers can be passed between programs. This is achieved through the use of channels. A channel is nothing more than a group of containers.

Similar to a COMMAREA, a channel can be used with the LINK, XCTL, START, and RETURN commands. A channel option is valid on the EXEC CICS RETURN COMMAND only if it is a pseudo-conversational RETURN with the TRANSID specified, or if RETURN to CICS is issued from the program at the highest logical level.

Only one channel can be passed to a program. Unlike a COMMAREA, there is no size limit for a container, which is constrained only by the amount of storage available. Also there is no limit on the number of containers that can be added to a channel.

HOW DOES IT WORK?

Let us take an example where program A passes a channel with two containers to program B. Program A can create the channel using the following PUT command:

```
EXEC CICS     PUT CONTAINER(container-name1)
              CHANNEL(channel-name) FROM (data-area)
```

The first PUT command creates the channel and adds the container to it. The second container can be added by using another PUT container command. Since the channel already exists, the second container gets added to it.

Program A can pass this channel to program B by using the CHANNEL(channel-name) option of the EXEC CICS LINK command. Program B can now access the channel and container by using the EXEC CICS GET CONTAINER command:

```
EXEC CICS     GET CONTAINER(container-name1)
              INTO (data-area)
```

If the channel name is not specified in the GET CONTAINER command, it is assumed to be the channel that was used to invoke program B (also referred to as the current channel).

Since program B was invoked using the LINK command, it can also return containers to program A – either by creating new ones or by reusing existing ones.

CREATING AND DELETING CHANNELS

Channels can be created in a program in many ways:

- As stated earlier, channels can be created using the EXEC CICS PUT CONTAINER command. The first PUT command creates the channel, if it does not already exist, and adds the container to it.
- If a channel that doesn't exist is specified in EXEC CICS LINK, XCTL, START, or RETURN commands, an empty channel is created.

Containers can be added to the channel by using PUT CONTAINER commands multiple times.

Additionally, there is a MOVE CONTAINER command that can be used to move containers from one channel to another.

```
EXEC CICS     MOVE CONTAINER(container-name)
              AS(container-new-name)
              CHANNEL(channel-name1) TOCHANNEL(channel-name2)
```

According to IBM, MOVE CONTAINER is also a more efficient way to transfer data compared with the PUT and GET CONTAINER commands.

Containers can be deleted from a channel using the EXEC CICS DELETE CONTAINER command.

Channels and containers are visible only to the program that creates them and to the ones they are passed to. When these programs terminate, the containers and their storage are automatically destroyed by CICS. In other words, when no program in the link stack can access a channel, it is deleted (this could occur following either EXEC CICS RETURN or XCTL).

Channel containers are not recoverable. BTS containers have to be used in the case of recoverable container requirements.

CURRENT CHANNEL

A program's current channel is the one with which it is invoked

(note that it is not mandatory to have a current channel because the program may be invoked without a channel). Though the program can create other channels, the current channel for a particular invocation of a program cannot be changed.

If the channel name is not specified in the PUT CONTAINER, GET CONTAINER, or DELETE CONTAINER commands, the current channel is assumed.

A program can be invoked with different channels (under different conditions) and it can determine which channel it has been invoked with by using the EXEC CICS ASSIGN CHANNEL command. Based on the channel it has been invoked with, it can tailor its own processing.

A program can modify its current channel by either adding containers to it or deleting containers from it.

DISCOVERING CHANNELS AND CONTAINERS

Typically, programs that exchange data – both client and server – are aware of the possible channels that they can be invoked with as well as the number of containers they have and the names of the containers in each of the channels. Notwithstanding this, IBM provides an option to discover the channels as well as the containers at run-time, providing more flexibility with the processing.

As stated earlier, the EXEC CICS ASSIGN CHANNEL command can be used to discover the current channel (returns a 16-character channel name). If there is no current channel, the command returns blanks. Similarly, the program can also get the container names in its current channel by browsing by using the browse commands:

```
EXEC CICS STARTBROWSE CONTAINER BROWSETOKEN(data-area)

EXEC CICS GETNEXT CONTAINER(data-area) BROWSETOKEN(token)

EXEC CICS ENDBROWSE CONTAINER BROWSETOKEN(token)
```

Having discovered the channels and the containers – from the data that has been passed to it – the server program can adjust its processing. In the past, this could be achieved with a COMMAREA by using a set of function codes and redefining the COMMAREA to interpret the data differently. But with identifiable channels and containers, this is achieved in a straightforward manner that provides better 'loose coupling' between the client and server programs.

It should be noted that because the called program can modify channels by adding/deleting containers, the channel may have a different set of containers in the calling program before and after the invocation of a program. Discovering containers is useful in such cases.

DATA CONVERSION IN CHANNELS

Data conversion is required when the character is passed across platforms with different encoding standards (eg ASCII to EBCDIC).

With a COMMAREA, such data conversion is handled by the system programmer by using the DFHCNV conversion table, the DFHCCNV conversion program, and, optionally, the DFHUCNV user-replaceable conversion program. With channels, this data conversion is much simplified and can be handled by the application program itself, using API commands.

CICS recognizes two type of data:

1 CHAR – character data or text strings. Data of this type can be converted to the code page of the application.

2 BIT – all non-character data. This type of data cannot be converted.

By default, the data type of the container is assumed to be a bit (and hence not eligible for conversion). To make it eligible for conversion, DATATYPE(DFHVALUE(CHAR)) has to be specified in the PUT CONTAINER command.

CICS can automatically handle the data conversion of the CHAR type container, depending on the code page of the application that retrieves it. This should be sufficient for standard ASCII to EBCDIC conversion and *vice versa*.

In addition to this, there is an option for handling explicit conversion from one code page to another by using the FROMCCSID option of the PUT CONTAINER command, and the INTOCCSID option of the GET CONTAINER command.

The INTOCCSID option of the GET CONTAINER command can also be used to prevent automatic conversion to the code page of the application. For example, the client application has put a message into a container in UTF8 format and the backend CICS TS would like to process the message in UTF8 format itself. It can do so by using:

```
EXEC CICS GET CONTAINER(input_msg) INTO(msg) INTOCCSID(utf8)
```

Similarly, if the CICS TS program wants to put a message into a container in UTF8 format and also not allow data conversion, then it can be done by using:

```
EXEC CICS PUT CONTAINER(output) FROM(output_msg) FROMCCSID(utf8)
```

CICS READ-ONLY CONTAINERS

CICS can create channels and containers for its own use, and pass them to user programs. Some of these containers are marked as read only. Read-only containers cannot be overwritten, moved, or deleted by the user programs. Hence, if you specify a read-only container on a PUT CONTAINER, MOVE CONTAINER, or DELETE CONTAINER command an INVREQ condition occurs.

At this stage, user programs cannot create read-only containers.

RECOMMENDED BEST PRACTICES

Performance related

A channel can be passed to a remote program or transaction,

but passing large amounts of data may affect performance – more so if it is an ISC connection. Though there is no longer the 32KB restriction, it is always better to constrain the amount of data passed so that it doesn't have an adverse impact on the storage available for other applications.

It is also worth noting that at the end of a DPL call, input containers that have not been changed by the server program are not returned to the client. Only input containers whose contents have been changed by the server program, and containers created by the server program, are returned.

IBM recommends the following best practices for optimal DPL performance:

- Use separate containers for input and output data (allowing smaller container flow in each direction).
- Allow the server program to create the output container (avoiding passing empty containers from client to server).
- Use separate containers for read-only and read-write data (because read-only data containers will not be returned by the server).
- Use separate containers for optional structures (containers can be passed only when the structure is present).
- Use dedicated containers for error information (so that error containers can be passed only in the case of error).
- Use separate containers for each structure (allowing users to pass a subset of the channel to subcomponents using the MOVE CONTAINER command).

Maintenance related

The best practices mentioned above also have a positive impact on the maintenance aspect of the application, as shown below:

- Having separate containers (for input/output, read-only/ read-write, error information, and each structure) allows

for better encapsulation of data as well as simplification of the copybook structure.

- When individual structures are to be changed, only the impacted programs need to be re-compiled, not all the components.
- The presence of an error can be checked just by issuing a GET CONTAINER (known-error-container-name) command. If the command returns a NOTFOUND condition, it indicates no error.

In addition to this, using separate structures for different data types (CHAR/BIT) is recommended because it improves the ability to handle code page conversions.

For improved maintenance, it is recommended that the names of the containers used and the data fields that map to the containers are stored in a copybook. The same copybook can be used in both the calling and called programs.

JCICS AND CHANNELS

For Java programs to use channels, the following three additional JCICS classes are provided:

1 com.ibm.cics.server.Channel

2 com.ibm.cics.server.Container

3 com.ibm.cics.server.ContainerIterator.

Channels and container can be handled in Java programs as follows:

- Use the createContainer() method of the channel class to create containers.
- Use the Container.put() method for putting data into the container.
- Use the link() and xctl() methods of the program class to pass channels to other programs.

- Use the getCurrentChannel() method of the task class to receive the current channel.
- Use the Container.get() method to get the container data.
- Use the ContainerIterator object to browse the current channel.

CICS also provides the following exception classes for handling errors:

1 com.ibm.cics.server.CCSIDErrorException

2 com.ibm.cics.server.ChannelErrorException

3 com.ibm.cics.server.ContainerErrorException.

BENEFITS

Channels and containers provide several advantages compared with the COMMAREA. Some of them are listed below:

- Unlike COMMAREAs, channels are not limited in size to 32KB.
- Since a channel can have multiple containers, the data passed can be well-structured compared with the former scheme of using a COMMAREA, which is a monolithic block of data.
- Unlike COMMAREAs, the programs that use channels are not required to know the exact size of the data returned.
- Channels can be used by CICS application programs written in any of the CICS-supported languages. For example, a Java client program on one CICS region can use a channel to exchange data with a COBOL server program on a back-end AOR.
- A server program can be written to dynamically discover the channels and containers it is invoked with, and vary its processing according to the kind of data passed.

- Components can be built from sets of related programs with simplified external interfaces whereby the client programs can just invoke one interfacing program with channels.
- The loose coupling between clients and components permits easy evolution of programs, where clients and components can be upgraded at different times. For example, first a component could be upgraded to handle a new channel, then the client program upgraded (or a new client written) to use the new channel.
- Since CICS automatically destroys containers (and their storage) when they go out of scope, the programmer need not have storage management concerns.
- While the data conversion is controlled by the system programmers in a COMMAREA application, in channel applications it is controlled by the application programmer, using simple API commands.
- Programs that use containers can be called from both channel and BTS applications.
- As channel containers are similar to BTS containers, converting non-BTS applications to use containers can be the first step in the migration route to full BTS applications.

CONCLUSION

CICS application programs that use traditional COMMAREAs will continue to exchange data as before. Although the simplest way of migrating from COMMAREAs to channels is to convert the monolithic block of COMMAREA into a channel with a single container, it is not preferred. To exploit the benefits of channels, it is better to design the use of channels to be in line with the best practices recommended.

Sasirekha Cota
System Software Group
Tata Consultancy Services (India)

Monitoring transaction flow started from an LU

Some applications are used through LU6.2 connections and sometimes, although the connection is seen as acquired, the users cannot start transactions in a CICS region because of a problem. In such cases, CICS monitoring tools may not produce any alerts for the application because the CICS system works fine and all the other applications can be used without any problem by the end users. If the application is a critical one for business and does have a high usage during certain times at the day, a small outage may cause high user disappointment. So, in order to identify such a problem quickly, the transaction flow should be monitored – and when this flow is interrupted, an alert should be produced.

To accomplish this, the POVI (Programmerless Open VTAM Interface) facility of AF/OPERATOR can be used with Omegamon/CICS. Such a POVI program is listed below. This program is written to monitor an LU6.2 connection in a CICSPlex environment. It assumes that the CICSPlex has only two TORs and each runs in a different MVS LPAR. The following program interacts with Omegamon/CICS by using the POVI facility and performs the required actions. If there is something that is not normal an alert message is displayed on the operator console.

Since the connection can be acquired at any of the TORs, it monitors the two TORs for the transactions that are started from the connection. If the LU is not acquired by any of the TORs, or no transaction is started from the LU in both TORs for a specified period of time (this period is coded in the program), an alert is produced.

On Omegamon/CICS, the LU should be defined for response time monitoring in the global data area modules of the two TORs. The group number of this definition is used to control the task history from the POVI program.

The CICS IDs of the TORs should also be defined on the POVI

interface. Those CICS IDs are also used in the program.

The program is invoked by giving a user-defined command from the console. It should be invoked in the two MVS LPARs on which the TORs run. AF/OPERATOR captures the user-defined command and runs the following program. When the program is invoked, the connection ID and the group number for the connection are passed to the program as parameters. After controlling the task history for the TORs, the program is set up to re-run itself after a calculated time. This time is calculated according to the end-time of the last transaction that was started from the LU. This process continues until the time coded in the program is reached.

Sample group definitions for an LU in a global data area module:

```
*
  <<GROUP>>
  GROUP_NUMBER=27
  GROUP_NAME=(LUXX        )
  GROUP_TYPE=LU
  RESPONSE_TIME_THRESHOLD=20
*
*
  <<ID>>
  LU=NETNLUXX
  ELIGIBLE_GROUPS=(27)
  FIXED_DISPLAY=NO
  TOTAL_RESPONSE_THRESHOLD=20
  HOST_RESPONSE_THRESHOLD=10
  NETWORK_RESPONSE_THRESHOLD=10
*
```

An AF/OPERATOR trap for invoking a POVI program looks like:

```
TRAP DEL(RESLUXX)
TRAP ADD(RESLUXX) +
 CMD('RESLUXX') ENA +
 ACT('EX RESXXXXR ''LUXX 27''')
```

POVI PROGRAM RESXXXXR

```
/* REXX */
```

```
/* When a related command is captured by AF/OPER, this program is
   invoked; and as parameters, the connection ID and the group number
   for the connection is passed. By using this group number,
   Omegamon/CICS task history is controlled for the transactions
   started on the CICSPlex TORs from the given LU. If no transactions
   are started from the LU in last 10 minutes, an alert message
   is displayed on the operator console.
*/

/* Note1: This program does the required task history controls
          only between certain periods of time during the day and
          re-activates itself till the end of that period of time.
*/

/* Note2: This program assumes that the CICSPlex has only two TORs
          and each runs on different MVS LPARs.
*/


Parse Arg conn_name, omeg_grp_num

Address mvs

NEWSTACK

trap_cmd = 'RES' || conn_name

RET = SYSVGET('AOSYSID')
say 'Sysid is:' AOSYSID
sysid_suff_local = substr(AOSYSID,3,2)

if (sysid_suff_local = 90) then sysid_suff_remote = 91
else sysid_suff_remote = 90

/* Decide applid of POVI address space and Omegamon/CICS session id
   to be used according to the sysid on which this program runs. */

/* For test CICSPlex */
if AOSYSID = 'TD90' then do
   povilu = 'povit90p'
   omegcics = 'omcitst1'
end
if AOSYSID = 'TD91' then do
   povilu = 'povit91p'
   omegcics = 'omcitst2'
end
/* For production CICSPlex */
if AOSYSID = 'PR90' then do
   povilu = 'povip90p'
   omegcics = 'omcipst1'
```

```
end
if AOSYSID = 'PR91' then do
   povilu = 'povip91p'
   omegcics = 'omcipst2'
end

/* Check whether the time is in the correct interval of the day. */
/* The time intervals to do the controls are                     */
/*                          09:00-12:30 or 12:30-18:00           */
IF ((time(m) >= 540 & time(m) <= 750) |,
    (time(m) >= 810 & time(m) <= 1080)) then do
   wait_threshold = 600 /* This is the period of time (10 minutes)
                           to check if any txn. is started from
                           the LU  */
   wait_time = wait_threshold

   call Omega_Logoff   /* Logoff from POVI sessions if logged on */
   call Omega_Logon    /* Logon Omegamon/CICS through POVI */
   call Check_Connection conn_name /* Check if the connection
                            acquired on the TOR which is on the
                            current MVS LPAR  */
   if (conn_acquired = 0) then do
      /* set flag */
      call Set_Conn_Inact_At_Loc_Flag conn_name sysid_suff_local
      /* check if the connection acquired on the other TORs which
         is on the other MVS LPAR */
      call Check_if_Conn_Acq_Anywhere conn_name sysid_suff_remote
   end
   if (conn_acquired = 1) then do
      /* set flag */
      call Set_Conn_Act_At_Loc_Flag conn_name sysid_suff_local
      /* Get Omegamon/CICS task history data for the connection */
      call Get_Omega_History omeg_grp_num
      /* if history data is not empty then check the last txn. time
         started from the connection */
      if (hist_empty = 0) then call Check_Txn_Time conn_name
      else do
         /* If history data not found then display alert message
            on the operator console. */
         Address Afhost
         $WTO 'WARNING - No Omegamon/CICS data for $,
         $connection: $ conn_name $.' DESC(1)$
         Address mvs
      end
   end
   call Omega_Logoff   /* Logoff from POVI sessions */

   say 'Trap wait has started..'
   Address Afhost
   /* wait till the next start */
```

```
   $WAIT SECONDS($wait_time$)$
   /* re-activate the program to do the same controls */
   $OPER 'RO $AOSYSID$,$trap_cmd$'$
   Address mvs
   say 'Trap wait has ended..'

end
else do
   shrvar = 'TRAP_RESP.' || conn_name || '.' || sysid_suff_local
   ttt=SHARVDEL('SYSPLEX',shrvar)
   say 'The following shared variable has been deleted. ',
       'Ret.Code ==>' ttt
   say 'Shrvar ==>' shrvar
end

exit Ø


/*********************************************************************/
/*********************************************************************/
/*********************************************************************/


Omega_Logon:
   /* Subroutine for logging on Omegamon/CICS through POVI        */
   /* POVI Playback Manager ACB                   : povilu       */
   /* Omegamon/CICS session id that is defined
                                          on POVI  : OMEGCICS     */
   /* Username for logging on Omegamon/CICS       : xxuserid     */
   /* Password for logging on Omegamon/CICS       : xxpass       */
   /*    Userid and password are hardcoded below and the user
         should be able to perform authorized actions.           */

   toolsrc =POVILGON(povilu,omegcics)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc =POVIFIND(TOP,$ENTER USERID ==> $,)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVISEND($xxuserid$,ENTER,$PASSWORD ==> $)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVISEND($xxpass$,ENTER,$ZMENU$,,,CONFIDENTIAL)
   IF (toolsrc <> Ø) then call error_routine
return

Omega_Logoff:
   /* Subroutine for logging off from POVI */
   DELSTACK
   toolsrc =POVILGOF()
```

```
return

Get_Omega_History:
   /* Subroutine for getting Omegamon/CICS task history data
      for the connection */

   arg txn_group_number

   toolsrc=POVILOC(ROWCOL,1,2)
   toolsrc=POVISEND($ZONDV$,ENTER,$ZONDV$)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVILOC(ROWCOL,1,2)
   toolsrc=POVISEND($ZONDT$,ENTER,$ZONDT$)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVISEND($$,PF8,$INC-GROUPS=$)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVILOC(ROWCOL,6,16)
   toolsrc=POVISEND(txn_group_number,ENTER,,
                    $INC-GROUPS='$txn_group_number$ $)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVILOC(ROWCOL,1,2)
   toolsrc=POVISEND($ZONDV$,ENTER,$ZONDV$)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVIDATA(8,1,42)
   IF (toolsrc <> Ø) then call error_routine
   parse pull line_text
   hist_data_found_str = '+                    Transaction Overview: '

   /* History data found for the connection */
   IF (line_text = hist_data_found_str) then do
      hist_empty = Ø

      toolsrc=POVIDATA(8,1,8Ø)
      IF (toolsrc <> Ø) then call error_routine
      parse pull line_text1 line_text2 line_text3 line_text4,
                 line_text5
      first_txn_date = line_text4

      toolsrc=POVIDATA(12,1,8Ø)
      IF (toolsrc <> Ø) then call error_routine
      parse pull line_text1 line_text2 line_text3
      first_txn_time = line_text2
   end

   /* History data not found for the connection */
```

```
   IF (line_text <> hist_data_found_str) then do
      hist_empty = 1
      Say 'No data..'
   end

return

Check_Connection:
   /* Subroutine for checking if the connection is acquired
      on the TOR which is on the current MVS LPAR  */
   arg connection_id

   toolsrc=POVILOC(ROWCOL,1,2)
   toolsrc=POVISEND($ZTCT$,ENTER,$ZTCT$)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVILOC(ROWCOL,1Ø,2)
   toolsrc=POVISEND($TABL TCT,ID=$connection_id,ENTER,$Terminal$)
   IF (toolsrc <> Ø) then call error_routine

   toolsrc=POVIDATA(14,8,34)
   IF (toolsrc <> Ø) then call error_routine
   parse pull line_text
   conn_acq_str = 'Session status  . . . . : Acquired'
   conn_rel_str = 'Session status  . . . . : Released'
   IF (line_text = conn_acq_str) then do
      conn_acquired = 1
   end
   else if (line_text = conn_rel_str) then do
           conn_acquired = Ø
        end
        else if (line_text = conn_rel_str) then do
                conn_acquired = -1
             end

return

Check_if_Conn_Acq_Anywhere:
   /* Subroutine for checking if the connection is acquired
      on the other TORs which is on the other MVS LPAR */
   arg connection_id sysid_suff_rem

   shrvar = 'TRAP_RESP.' || connection_id || '.' || sysid_suff_rem
   ttt=SHARVGET('SYSPLEX',shrvar)
   if ttt = Ø then do
      if (TRAP_RESP.connection_id.sysid_suff_rem = Ø) then do
         Address Afhost
         /* Display alert message on the operator console that
            the connection is not aquired anywhere. */
         $WTO 'WARNING - $ connection_id $  is not acquired at $,
```

```
         $ any TOR..' DESC(1)$
         Address mvs
      end
   end

return

Set_Conn_Act_At_Loc_Flag:
   /* Subroutine for setting the flag that connection is acquired
      on the TOR which is on the current MVS LPAR  */
   arg connection_id sysid_suff_loc

   TRAP_RESP.connection_id.sysid_suff_loc = 1
   shrvar = 'TRAP_RESP.' || connection_id || '.' || sysid_suff_loc
   ttt=SHARVDEL('SYSPLEX',shrvar)
   ttt=SHARVPUT('SYSPLEX',shrvar)
return

Set_Conn_Inact_At_Loc_Flag:
   /* Subroutine for setting the flag that connection is not acquired
      on the TOR which is on the current MVS LPAR  */
   arg connection_id sysid_suff_loc

   TRAP_RESP.connection_id.sysid_suff_loc = Ø
   shrvar = 'TRAP_RESP.' || connection_id || '.' || sysid_suff_loc
   ttt=SHARVDEL('SYSPLEX',shrvar)
   ttt=SHARVPUT('SYSPLEX',shrvar)
return

Check_Txn_Time:
   /* Subroutine for checking the last txn. time started from
      the connection */
   arg connection_id

   /* calculate total seconds of the system time since midnight */
   system_date = date(u)
   system_hour  = left(time(),2)
   system_hour  = right('Ø' || system_hour,2)
   system_minute= substr(time(),4,2)
   system_minute= right('Ø' || system_minute,2)
   system_second= substr(time(),7,2)
   system_second= right('Ø' || system_second,2)
   system_tot_second = (((system_hour*6Ø)+system_minute)*6Ø)+,
                       system_second

   /* calculate total seconds of the last txn. time since midnight */
   if LENGTH(first_txn_time) = 8 then do
      txn_hour  = left(first_txn_time,2)
      txn_hour  = right('Ø' || txn_hour,2)
      txn_minute= substr(first_txn_time,4,2)
```

```
      txn_minute= right('0' || txn_minute,2)
      txn_second= substr(first_txn_time,7,2)
      txn_second= right('0' || txn_second,2)
   end
   else do
      txn_hour  = left(first_txn_time,1)
      txn_hour  = right('0' || txn_hour,2)
      txn_minute= substr(first_txn_time,3,2)
      txn_minute= right('0' || txn_minute,2)
      txn_second= substr(first_txn_time,6,2)
      txn_second= right('0' || txn_second,2)
   end
   txn_tot_second = (((txn_hour*60)+txn_minute)*60)+,
                       txn_second

   /* calculate the time difference */
   time_diff = system_tot_second - txn_tot_second

   /* If the time difference is less than the given threshold,
      then calculate new wait_time for re-activating the program
      to do the same controls */
   IF (time_diff <= wait_threshold &,
       first_txn_date = system_date) then do
      SAY 'Less than 10 minutes..'
      wait_time = wait_threshold - time_diff
   end

   /* If the time difference is greater than the given threshold,
      then display an alert message on the operator console. */
   IF ((time_diff > wait_threshold & first_txn_date = system_date) |,
       first_txn_date <> system_date) then do
      SAY 'Warning - More than 10 minutes..'
      wait_time = wait_threshold
      Address Afhost
      $WTO 'WARNING - No transaction is started from $,
      $connection: $ conn_name $ for 10 minutes..' DESC(1)$
      Address mvs
   end

return

ERROR_ROUTINE:
   /* Subroutine for handling error conditions. */

   say 'Warning - An error has occurred..'
   PARSE SOURCE tso invoke_type exec_name the_rest
   CALL GLBVGET('aopvfnc')
   PARSE PULL feedback
   say 'Omeg CICS report error:'
   Say '     Trap action =>' exec_name
```

```
   Say '     Line        =>' sigl
   Say '     Function    =>' aopvfnc
   Say '     Return code =>' toolsrc
   Say '     Feedback    =>' feedback
   call Omega_logoff
   EXIT -1
return
```

Erhan Pasa
Senior Systems Programmer
Akbank TAS (Turkey)

Fly to another planet

DESCRIPTION

Many people hold the opinion that CICS offers only boring and dry dialogs. This is not correct – see Figure 1. Compile the program CSFLIG and the mapset CMFLIG, define the PPT and PCT entries and...fly to another planet. Have fun!

CMFLIG

```
        EJECT
        SPACE  3
MAPMT25 DFHMSD TYPE=MAP,TERM=3270,MODE=INOUT,STORAGE=AUTO,             *
               CTRL=FREEKB,LANG=PLI
*---------------------------------------------------------------------*
*              MAP4001 KONSTANTENAUSGABE FUER AUSWAHLBILD             *
*---------------------------------------------------------------------*
M1      DFHMDI SIZE=(24,80)
        DFHMDF POS=(1,25),LENGTH=30,ATTRB=(ASKIP,BRT),                 *
               INITIAL='START-PARAMETER FUER TASK EULE'
        DFHMDF POS=(3,10),LENGTH=02,INITIAL='IN'
INT     DFHMDF POS=(3,20),LENGTH=06,ATTRB=(UNPROT,IC)
        DFHMDF POS=(3,27),LENGTH=1
        DFHMDF POS=(4,19),LENGTH=8,INITIAL='HHMMSS'
        DFHMDF POS=(6,05),LENGTH=07,INITIAL='ODER UM'
ZEIT    DFHMDF POS=(6,20),ATTRB=(UNPROT),LENGTH=06
        DFHMDF POS=(6,27),LENGTH=5,INITIAL= '  UHR'
```

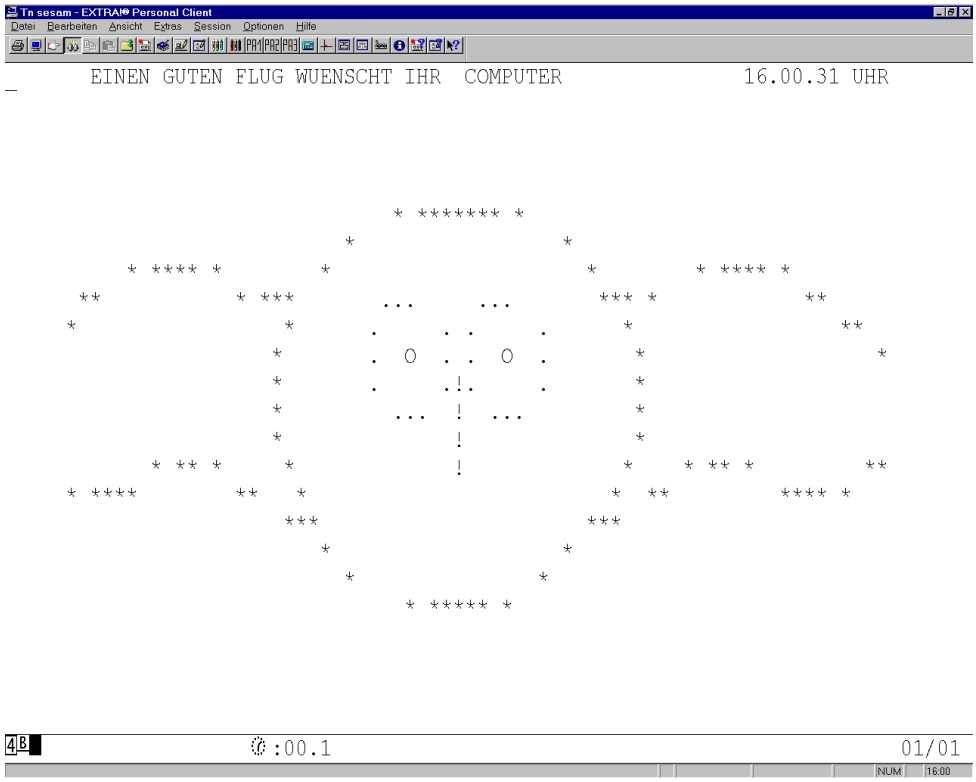


Figure 1: Screen image

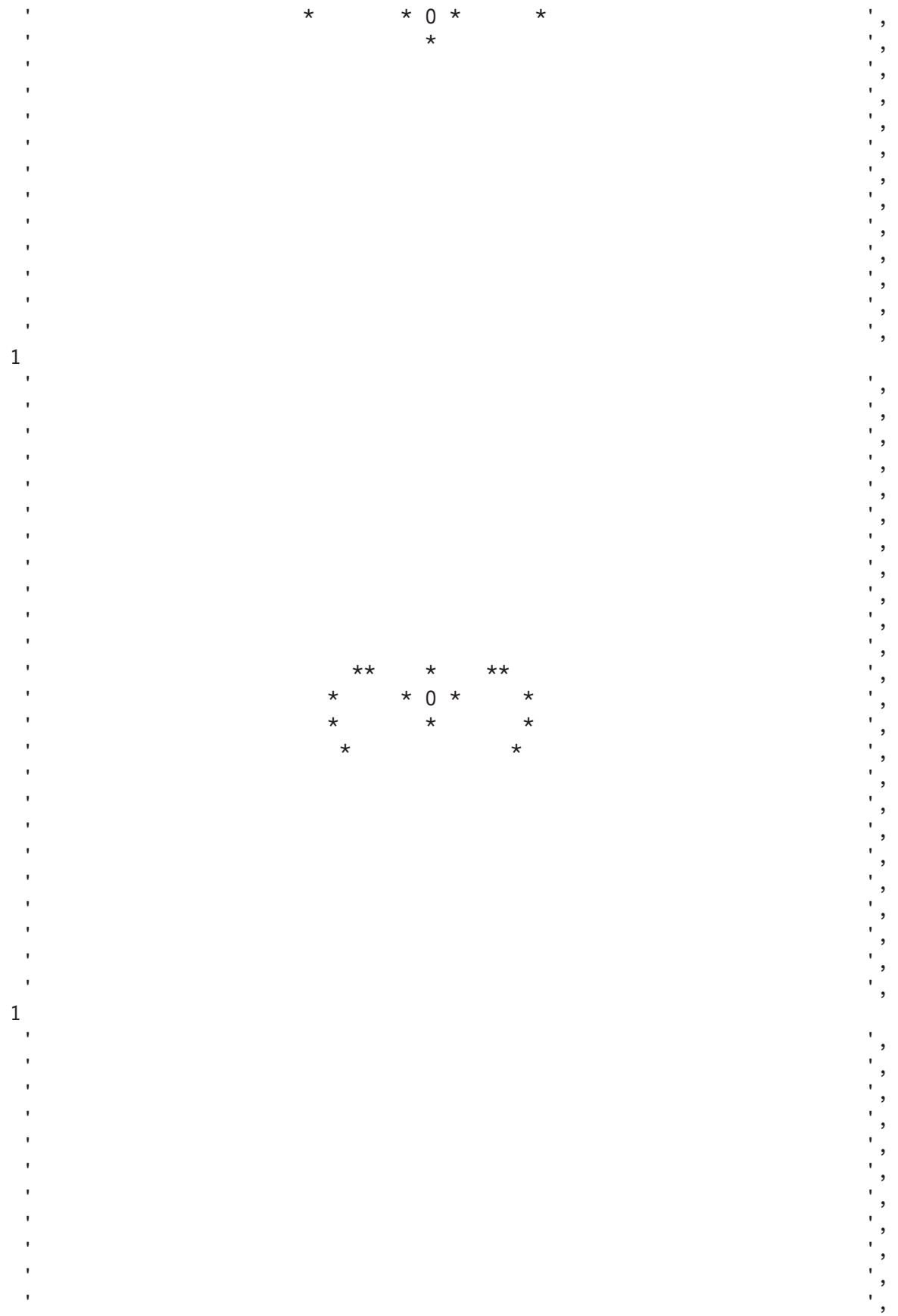
```
        DFHMDF POS=(6,40),LENGTH=12,INITIAL='ES IST JETZT'
IZEIT   DFHMDF POS=(6,53),LENGTH=09,PICOUT='Z99.99.99'
        DFHMDF POS=(6,63),LENGTH=05,INITIAL='  UHR'
        DFHMDF POS=(7,19),LENGTH=8,INITIAL='(HHMMSS)'
        DFHMDF POS=(9,05),LENGTH=06,INITIAL='AUF BS:'
BS      DFHMDF POS=(9,20),LENGTH=04,ATTRB=UNPROT
        DFHMDF POS=(9,25),LENGTH=01
        DFHMDF POS=(12,25),LENGTH=24,ATTRB=(ASKIP,BRT),                *
              INITIAL='PARAMETER FUER TASK EULE'
        DFHMDF POS=(14,4),LENGTH=11,INITIAL='INTERVALL-1'
INT1    DFHMDF POS=(14,20),LENGTH=02,ATTRB=UNPROT
        DFHMDF POS=(14,23),LENGTH=01
        DFHMDF POS=(15,19),LENGTH=04,INITIAL='(SS)'
        DFHMDF POS=(16,5),LENGTH=11,INITIAL='INTERVALL-2'
INT2    DFHMDF POS=(16,20),LENGTH=02,ATTRB=UNPROT
        DFHMDF POS=(16,23),LENGTH=01
        DFHMDF POS=(17,19),LENGTH=4,INITIAL='(SS)'
        DFHMDF POS=(18,05),LENGTH=18,INITIAL='ANZAHL DURCHLAEUFE'
ANZ     DFHMDF POS=(18,30),LENGTH=02,ATTRB=UNPROT
```

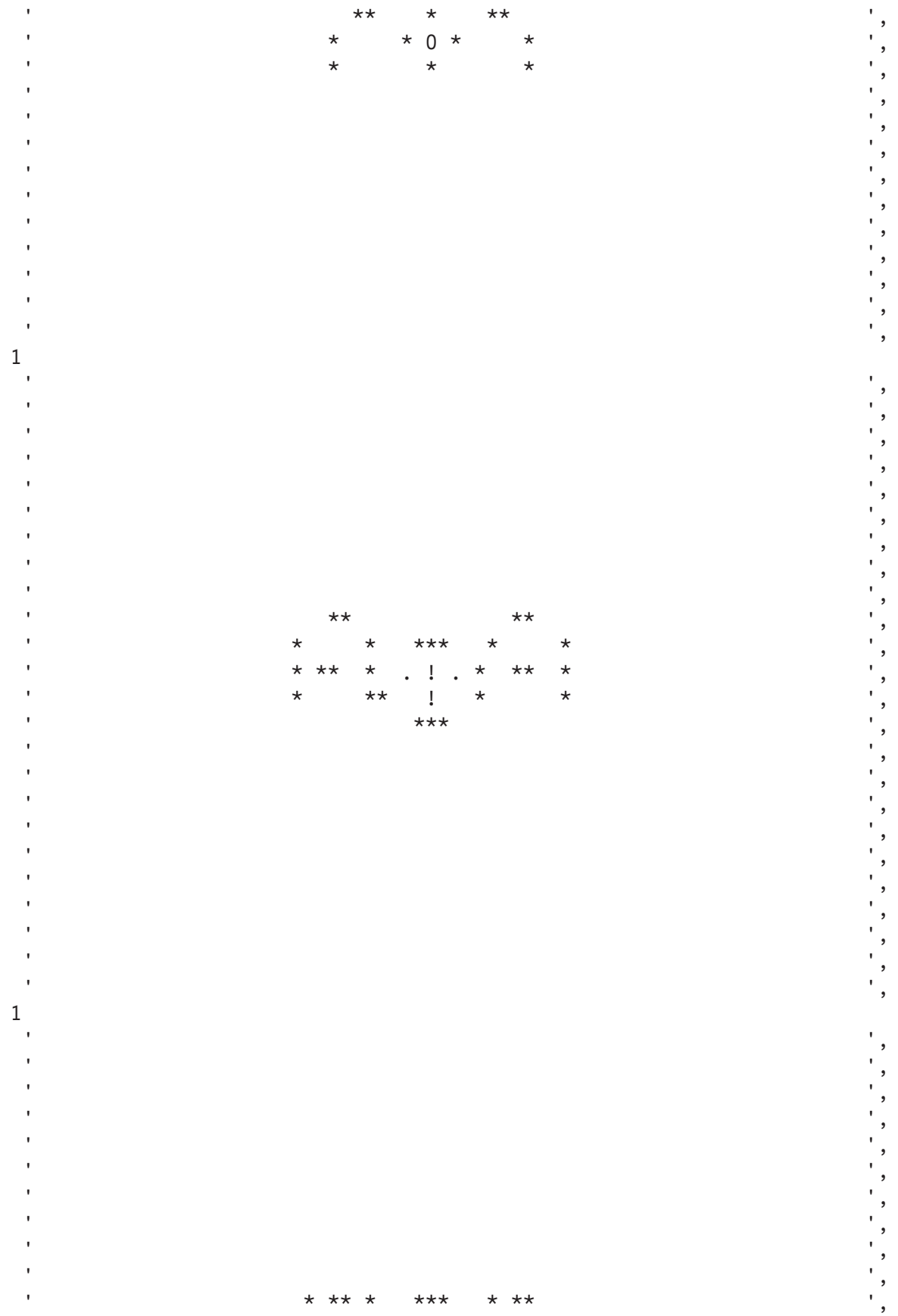
```
         DFHMDF POS=(18,33),LENGTH=1
         DFHMDF POS=(19,29),LENGTH=Ø4,INITIAL='(NN)'
         DFHMDF POS=(21,Ø5),LENGTH=11,INITIAL='ALARM (J/N)'
ALARM    DFHMDF POS=(21,3Ø),LENGTH=Ø1,ATTRB=UNPROT
         DFHMDF POS=(21,32),LENGTH=1
         DFHMDF POS=(22,Ø5),LENGTH=Ø4,INITIAL='TEXT'
T1       DFHMDF POS=(22,3Ø),LENGTH=2Ø,ATTRB=UNPROT
         DFHMDF POS=(22,51),LENGTH=Ø1
         DFHMDF POS=(22,65),LENGTH=Ø6,INITIAL='TASTEN'
T2       DFHMDF POS=(23,3Ø),LENGTH=2Ø,ATTRB=UNPROT
         DFHMDF POS=(23,51),LENGTH=1
         DFHMDF POS=(23,65),LENGTH=11,INITIAL='CLEAR/ENTER'
FM       DFHMDF POS=(24,Ø1),LENGTH=6Ø,ATTRB=(BRT,ASKIP)
M2       DFHMDI SIZE=(24,8Ø)
Z        DFHMDF POS=(1,1),LENGTH=79,OCCURS=24,ATTRB=ASKIP
ML       DFHMDI SIZE=(24,8Ø)
         DFHMDF POS=(1,1),LENGTH=Ø1,ATTRB=(UNPROT,IC)
         DFHMSD TYPE=FINAL
         END
```

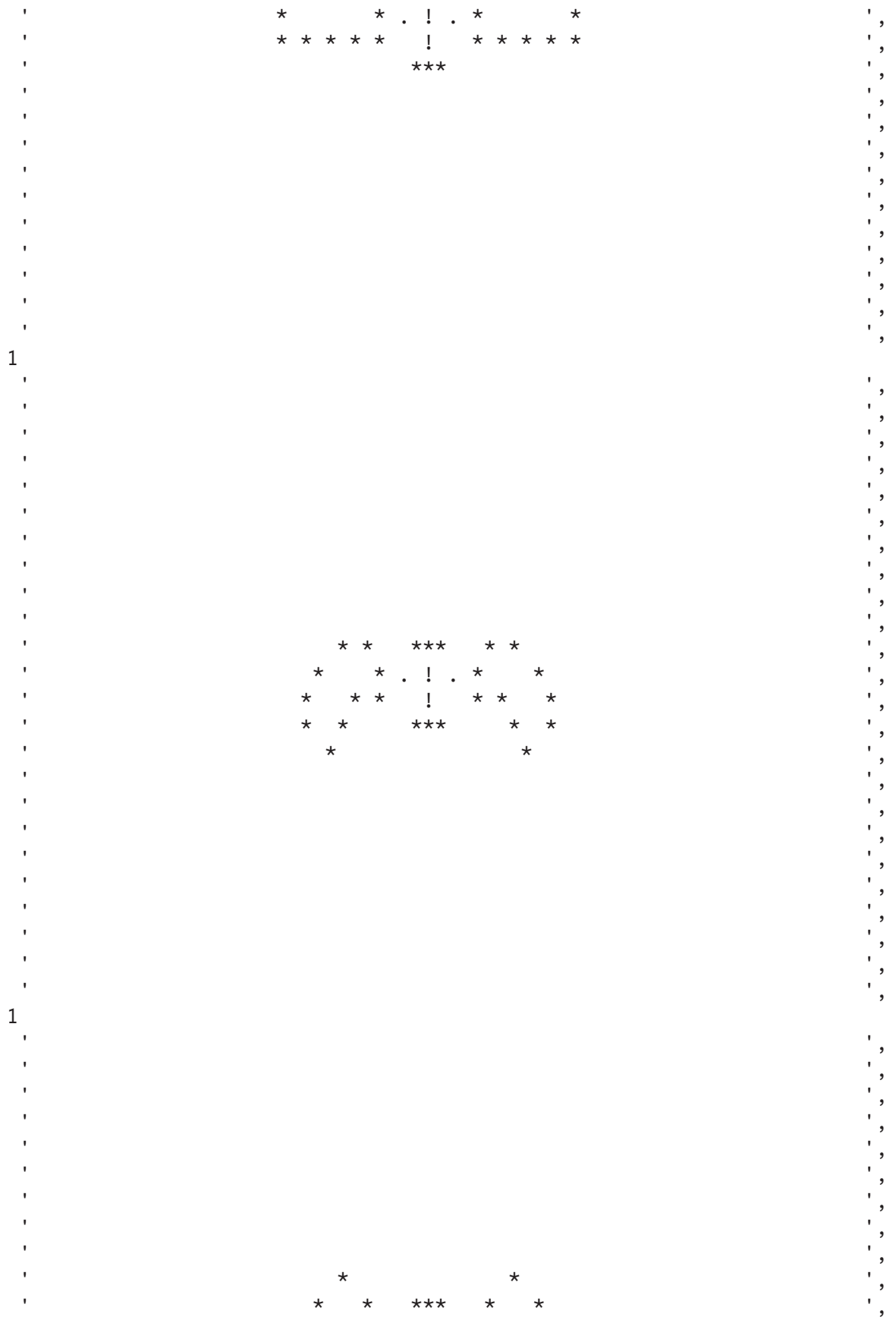
CSFLIG

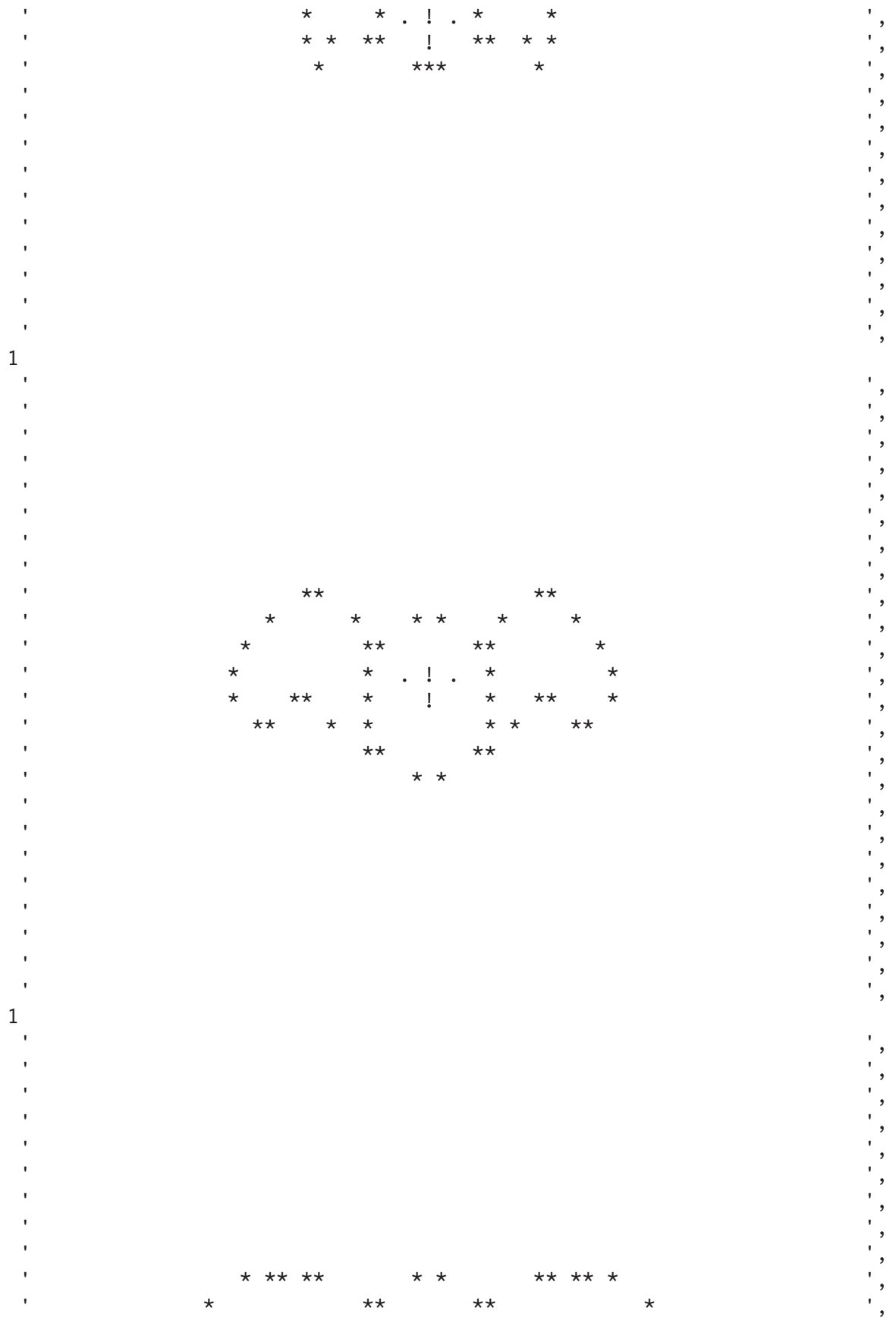
```
* PROCESS CICS(SOURCE MARGINS(2,72,1)),INCLUDE,LIST;
 DFHFLIG:      PROC OPTIONS (MAIN);
         DCL PLIXOPT CHAR (2Ø) VAR STATIC EXTERNAL
             INIT('ISASIZE(1ØØØØ) STAE');
 %INCLUDE DFHAID;
             DCL PANLEVEL CHAR(8) INIT ('&PANLEVL');
             DCL PANDATE  CHAR(8) INIT ('&PANDATE');
             DCL PANTIME  CHAR(8) INIT ('&PANTIME');
Ø        DCL   1 PARAM,
                2 INT1            FIXED(7) INIT(Ø),
                2 INT2            FIXED(7) INIT(Ø),
                2 ANZ             BIN FIXED(15) INIT(2),
                2 ALARM           CHAR(1) INIT('N'),
                2 T1              CHAR(2Ø) INIT(' '),
                2 T2              CHAR(2Ø) INIT(' ');
         DCL   MS                 CHAR(1) INIT('1');
         DCL   BS                 CHAR(4);
         DCL   INT                FIXED(7) INIT(Ø);
         DCL   LNG                BIN FIXED(15) INIT(51);
         DCL   FEHLTAB(5)         CHAR(6Ø) INIT(
               'FELD NICHT NUMERISCH',
               'EINGABE NICHT N ODER J',
               'ENTWEDER INTERVALL ODER ZEIT EINGEBEN',
               'BILDSCHIRM NICHT VORHANDEN',
               '                          ');
         DCL   NUM                CHAR(1Ø) INIT('1234567891Ø');
         DCL   ZWPIC              PIC'Z99.99.99';
```

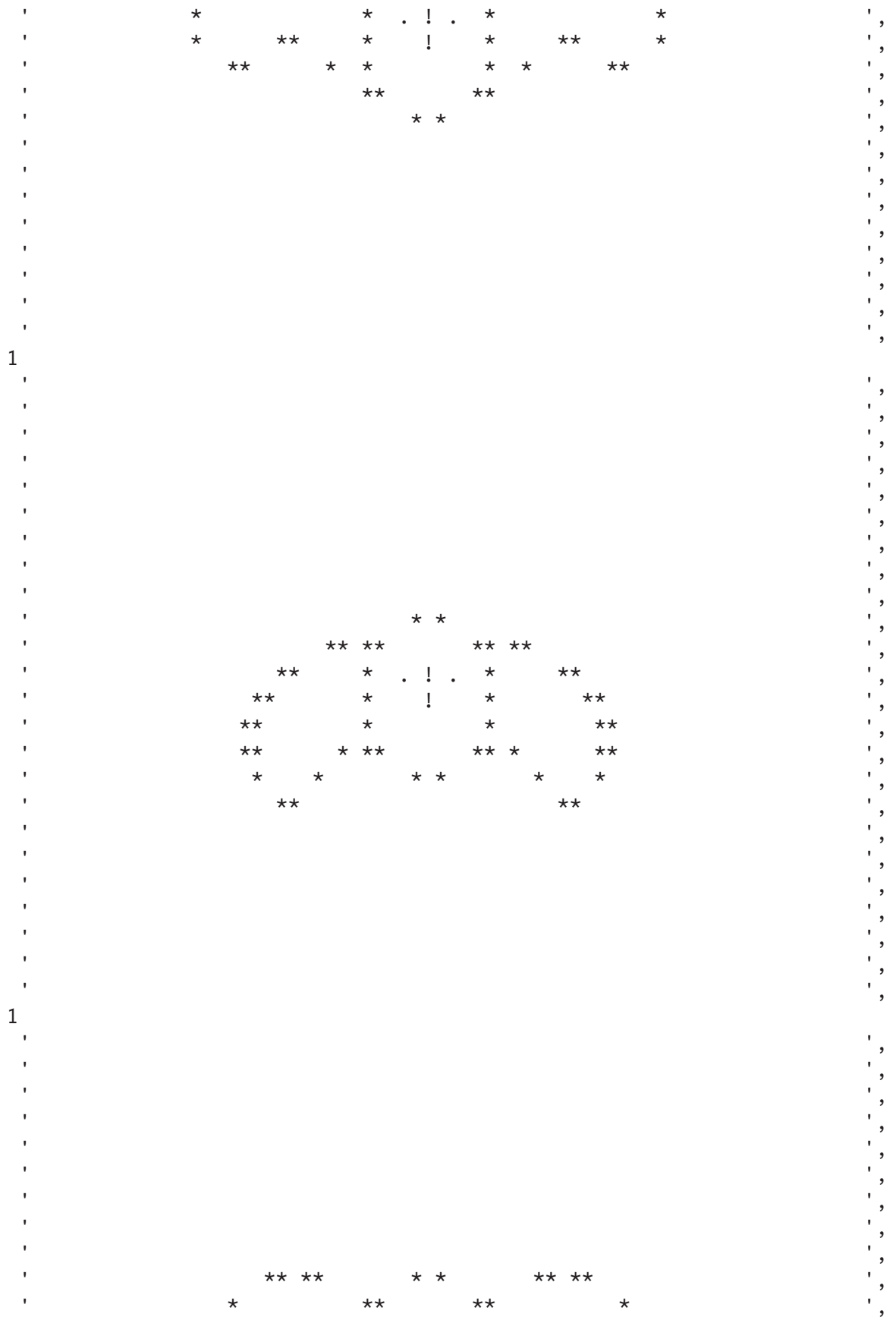
```
         DCL   DUMMY             CHAR(1Ø) INIT('DUMMYDUMMY');
 %INCLUDE CMFLIG;
         DCL   M1RED             CHAR(3ØØØ) BASED(ADDR(M1O));
         DCL   M2RED             CHAR(3ØØØ) BASED(ADDR(M2O));
         DCL   MLRED             CHAR(3ØØØ) BASED(ADDR(MLO));
         DCL   INTIR             PIC'(6)9' BASED(ADDR(INTI));
         DCL   INT1IR            PIC'(2)9' BASED(ADDR(INT1I));
         DCL   INT2IR            PIC'(2)9' BASED(ADDR(INT2I));
         DCL   ZEITIR            PIC'(6)9' BASED(ADDR(ZEITI));
         DCL   ANZIR             PIC'(2)9' BASED(ADDR(ANZI));
Ø/*    E U L E   */
         DCL   MA(24,24)         CHAR(68) INIT(
1
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                              O                                   ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
1
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                                                                  ',
 '                      * **    *    ** *                          '
```

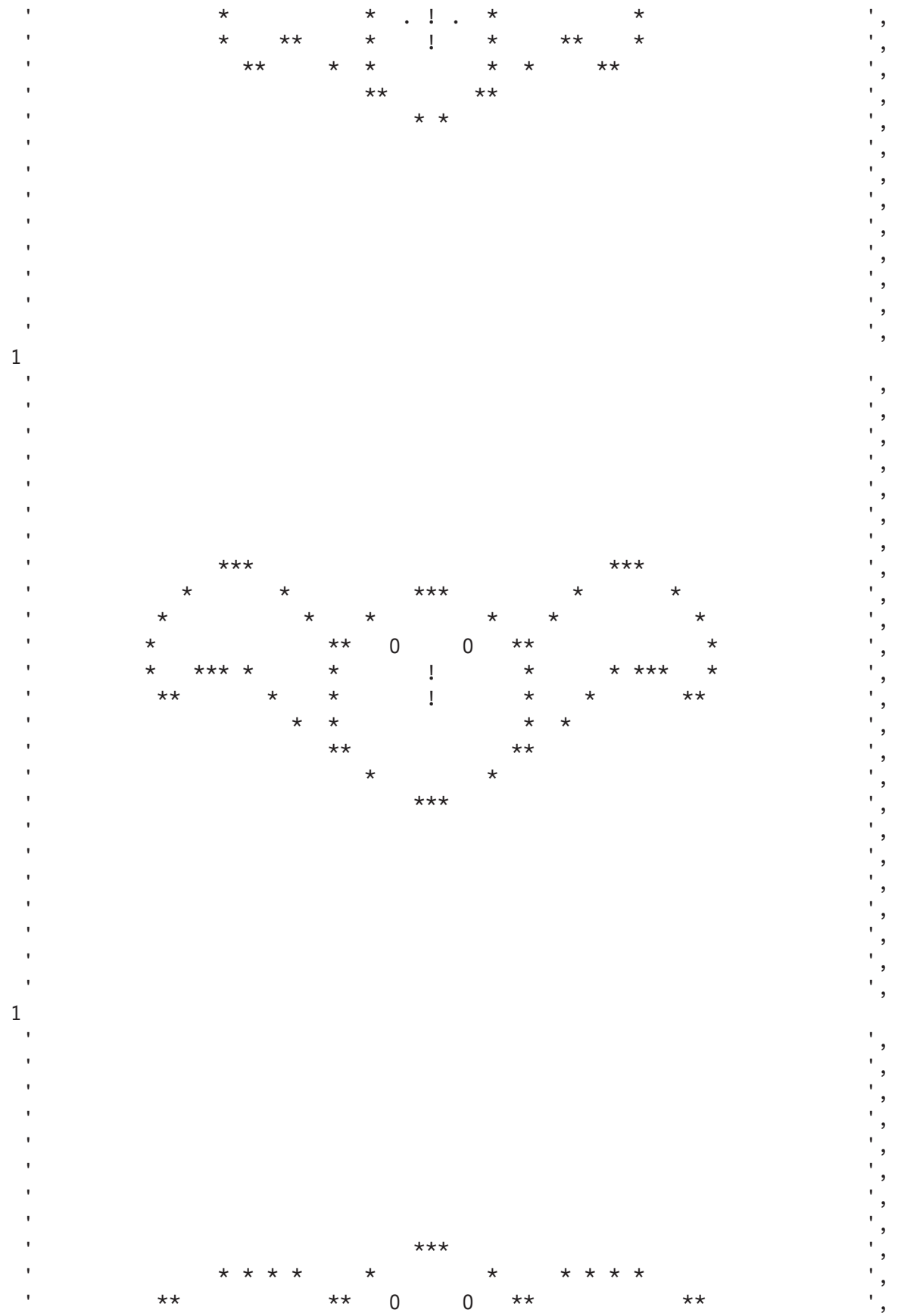


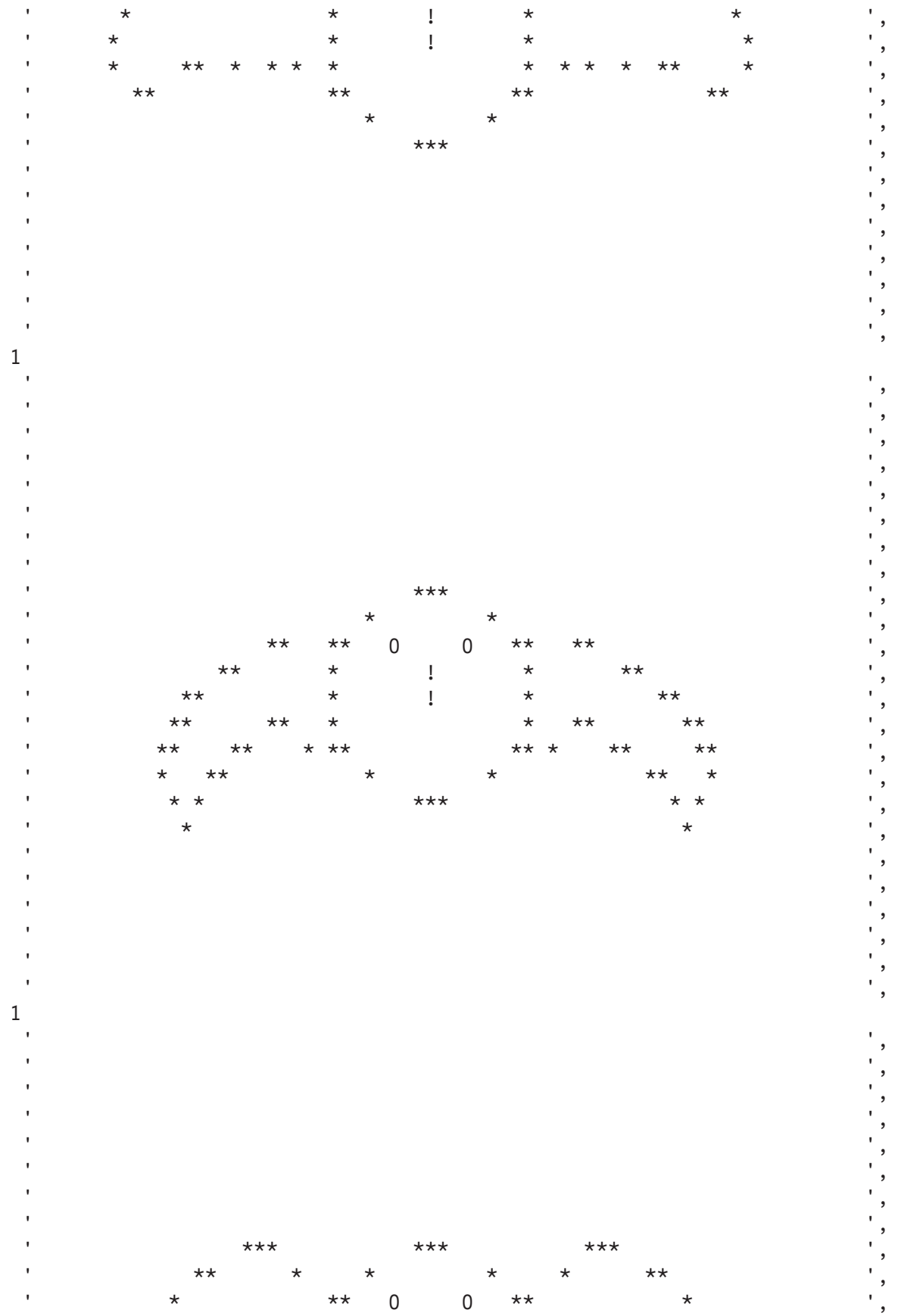


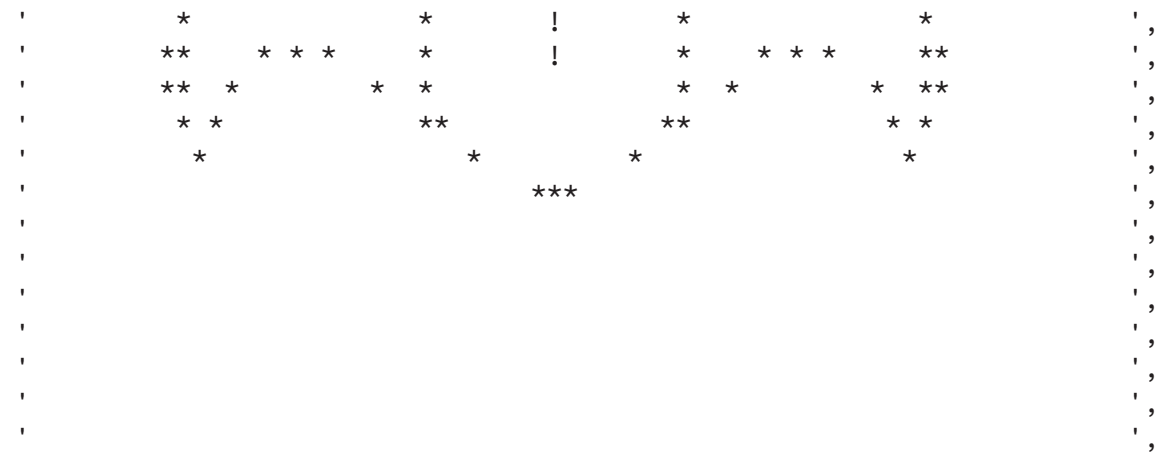












Editor's note: this article will be concluded next month.

Dieter Maier
CICS Systems Programmer (Germany)

Please note that the correct contact address for Xephon Inc is PO Box 550547, Dallas, TX 75355, USA. The phone number is (214) 340 5690, the fax number is (214) 341 7081, and the e-mail address to use is info@xephon.com.

Acucorp has announced Version 7 of extend, which it claims allows companies to enhance their legacy applications and reduce their maintenance costs by taking advantage of new interoperability and compatibility features. These features include expanded support for distributed CICS; facilities for integrating COBOL with Java, C, and C++; and improved compatibility with other COBOL dialects.

Using extend7, developers can exploit the flexibility and cost savings inherent in distributed environments while preserving their CICS investments. The product supports both IBM's TXSeries for Multiplatforms running on AIX, HP-UX, Windows, and Solaris, as well as Sun Microsystem's Mainframe Transaction Processing (MTP) and Mainframe Batch Manager (MBM) running on Solaris. Companies can make the most of their CICS skills, and enhance those applications with Internet deployment and portability.

For further information contact:
URL: www.acucorp.com/company/press/releases/2005/2005_pr_5.php.

* * *

NEON Systems has announced support for CICS Transaction Server Version 3.1 through NEON's mainframe Web services solution, Shadow z/Services.

The company claim that Shadow RTE can simplify the deployment of Service-Oriented Architectures (SOA) for organizations using CICS to run their mission-critical business services. Shadow's ease of development, implementation, and operation, the company says, can be used to extend all the new facilities and features available with CICS TS 3.1.

For further information contact:
URL: www.neonsys.com/newsroom/press_releases/2005/20050627.asp.

* * *

NetManage has announced that Xcalia has embedded NetManage Librados adapters into their Xcalia Intermediation Platform. Xcalia dynamically integrates heterogeneous data, including legacy and mainframe data, with new and existing services including both Web (SOAP, etc) and legacy (CICS, IMS, WebSphere MQ, etc) to deploy mission-critical, high-performance, transactional composite applications.

Xcalia offers several service modules as part of the intermediation platform. NetManage Librados JCA Plus CICS and JCA IMS adapters are incorporated into the Xcalia Mainframe Services module and other adapters are also used in the Packaged Application Services module. With the added connectivity benefits of NetManage Librados adapters, Xcalia can now provide customers end-to-end solutions for data and service integration.

For further information contact:
URL: www.xcalia.com/news/press_release.jspf.

* * *