Getting Back to Basics

CICS Users' Group, July 7, 2015
Mary Abdill
mabdill@nyc.rr.com
www.CreativeDataMovers.com

Topics for Today

Flow

- Flow of control - XCTL, LINK, CALL, START

Web Services

- Converting CICS programs to Web Services

MQ

- CICS programs that handle MQ messages

Pass data

- Passing data between CICS programs
- Passing data between Java web programs
- COMMAREAs or Channels/containers that work

Very Basic Basics

Most CICS programs are written in COBOL

COBOL celebrates its 56th year

- May 28, 2009, COBOL became 50 years old

More and more programs that access CICS programs are written in Java

Web Service calls, Servlet front ends, MQ triggers, etc.

CICS language (API) embedded in COBOL, ASM,PL/I, C/C++, REXX, Java, PHP

Translator (pre-compiler) converts CICS language into COBOL language

COBOL Code Restrictions

No direct file requests (no OPEN, CLOSE, READ, etc.)

No ACCEPT, DISPLAY, SORT, MERGE

No DYNAM, GRAPHIC, NOLIB, NORENT Compile options

Try to avoid COBOL verbs with high overhead (INITIALIZE, etc.)

continued

COBOL Code Restrictions (cont.)

COBOL Programs *may* use:

- INSPECT, STRING, UNSTRING
- SSRANGE Compile option
- Static and Dynamic CALLS to programs that may or may not contain CICS commands
- ADDRESS OF and LENGTH OF

STOP RUN does *not* crash the region any more. Recommend GOBACK.

Some CICS COBOL Code

```
MOVE LENGTH OF WS-MSG-RECEIVED TO WS-LEN
EXEC CICS RECEIVE
          INTO(WS-MSG-RECEIVED)
          LENGTH(WS-LEN)
          RESP(WS-RESP-CD)
END-EXEC

EVALUATE TRUE
WHEN WS-RESP-CD =  DFHRESP(LENGERR)
   MOVE 'INPUT KEY TOO LONG'  TO WS-SEND-MSG
WHEN WS-RESP-CD NOT = DFHRESP(NORMAL)
                    AND DFHRESP(EOC)
    MOVE 'ERROR ON INPUT'
         TO  WS-SEND-MSG
 WHEN OTHER
    PERFORM FORMAT-OUTPUT
 END-EVALUATE
```

Convert

Convert

Convert

Convert

Prepare a CICS COBOL program

Translate step:
converts CICS into COBOL

Compile step:
can include translate

Binder (aka Linkage Editor)

Must be in a PDS (now PDSE) library
that CICS knows about

Additional steps for DB2, WS, MAPs, etc.

Result after translate step

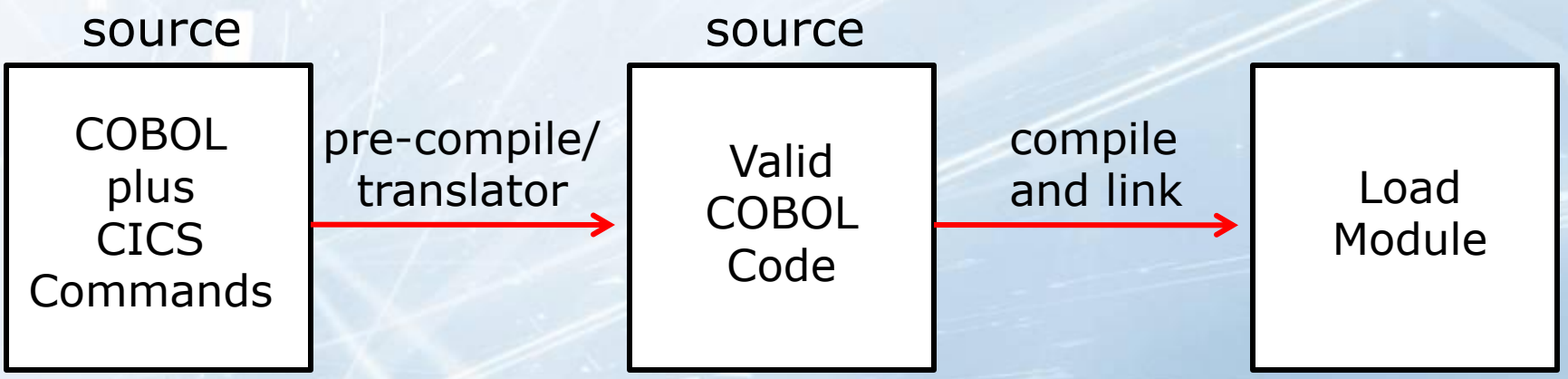
```
 MOVE LENGTH OF WS-MSG-RECEIVED TO WS-LEN
*EXEC CICS RECEIVE
*           INTO(WS-MSG-RECEIVED)
*           LENGTH(WS-LEN)
*           RESP(WS-RESP-CD)
*END-EXEC

 Call 'DFHEI1'
    using by content
   x'0402c000270000014000040000000f0f0f0f3f7404040'
    by reference WS-MSG-RECEIVED
    by reference WS-LEN
 end-call

 Move eibresp to WS-RESP-CD
```

Result after translate step, continued

```
 EVALUATE TRUE

*DFHRESP(LENGERR) = 22   ←INSERTED BY TRANSLATOR
 WHEN WS-RESP-CD =  22
      MOVE 'INPUT KEY TOO LONG'  TO WS-SEND-MSG

*DFHRESP(NORMAL) = 0    ← INSERTED BY TRANSLATOR
 WHEN WS-RESP-CD NOT = 0
*DFHRESP(EOC) = 6       ← INSERTED BY TRANSLATOR
             AND 6
      MOVE 'ERROR ON INPUT'  TO  WS-SEND-MSG

 WHEN OTHER
      PERFORM 4000-FORMAT-OUTPUT
 END-EVALUATE
```

CICS Commands and COBOL

- CICS commands are inserted into COBOL source
- Most commands become CALLs to the EIP
- EIP is a traffic cop that passes requests to the correct control programs
- The CICS translator pre-compiles the program (comments out CICS commands and inserts CALLs and data items)



Another Look at Translate

- Before Translate (precompile):

```
EXEC CICS RECEIVE INTO (IN-MSG) END-EXEC
```

- After Translate:

```
* EXEC CICS RECEIVE INTO (IN-MSG) END-EXEC
      MOVE 'DB&    ' TO DFHEIVO
      CALL 'DFHEI1' USING DFHEIVO IN-MSG
```

- Translator converts certain CICS functions to equivalent numeric values, such as:

```
IF WS-RESP-CD = DFHRESP(NORMAL) OR DFHRESP(EOC)
```

becomes:

```
IF WS-RESP-CD = 0 or 6
```

continued

Another Look at Translate (cont.)

- The translator *changes* your Procedure Division to:

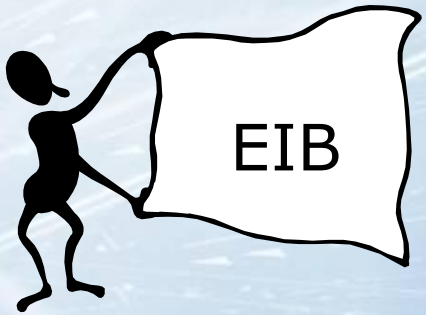
```
PROCEDURE DIVISION USING DFHEIBLK DFHCOMMAREA.
```

- Adds three data structures:


```
01  DFHEIVAR      (to Working-Storage Section)

01  DFHEIBLK      (to Linkage Section)

01  DFHCOMMAREA   (to Linkage Section)
```

Commonly Referenced EIB Items

EIBTIME	EIBAID
EIBDATE	EIBRCODE
EIBTRNID	EIBDS
EIBTASKN	EIBREQID
EIBCALEN	EIBCPOSN



Much Has Changed in 46 Years

- **July 8 - Happy Birthday CICS!**

- **CICS will be 46 Years on July 8, 2015 and Going Strong!**

Run a CICS program – Good Old Days

- Green screen terminal
- Logged onto a CICS region
- Typed in a CICS transaction

the "good old days"

Run a CICS program – Now Days

- 3270 emulation; logon and enter transaction
- HATS – gives web browser access. 3 tier.
- HOD – another 3 tier solution
- CWS - a feature of CICS TS API. 2 tier.
- Java web front end – send commarea to CICS
- Any program (usually web) – invoke CICS as a Web Service (SOAP or REST)
- Run CICS in background (using JCL), and run your CICS program in batch CICS region
- MQ triggers a CICS program
- Scheduler triggers, etc.

Tools for Development

Was Character UI	TSO	then ISPF
Now Eclipse-based GUI	RDz	z/OS Explorer/ CICS Explorer/ CICS plug-ins

Versions

IBM withdrawing support for

- CICS TS for z/OS V3.1 and V3.2 will be withdrawn from support December 31, 2015

Latest releases

- CICS TS V5.2 and CICS Explorer V5.2 GA date was June 13, 2014

Transaction, Task, and Pseudo-conversation

Transaction

A unit of work for the computer generally initialized by a user entering a Transaction Identifier (Transid)

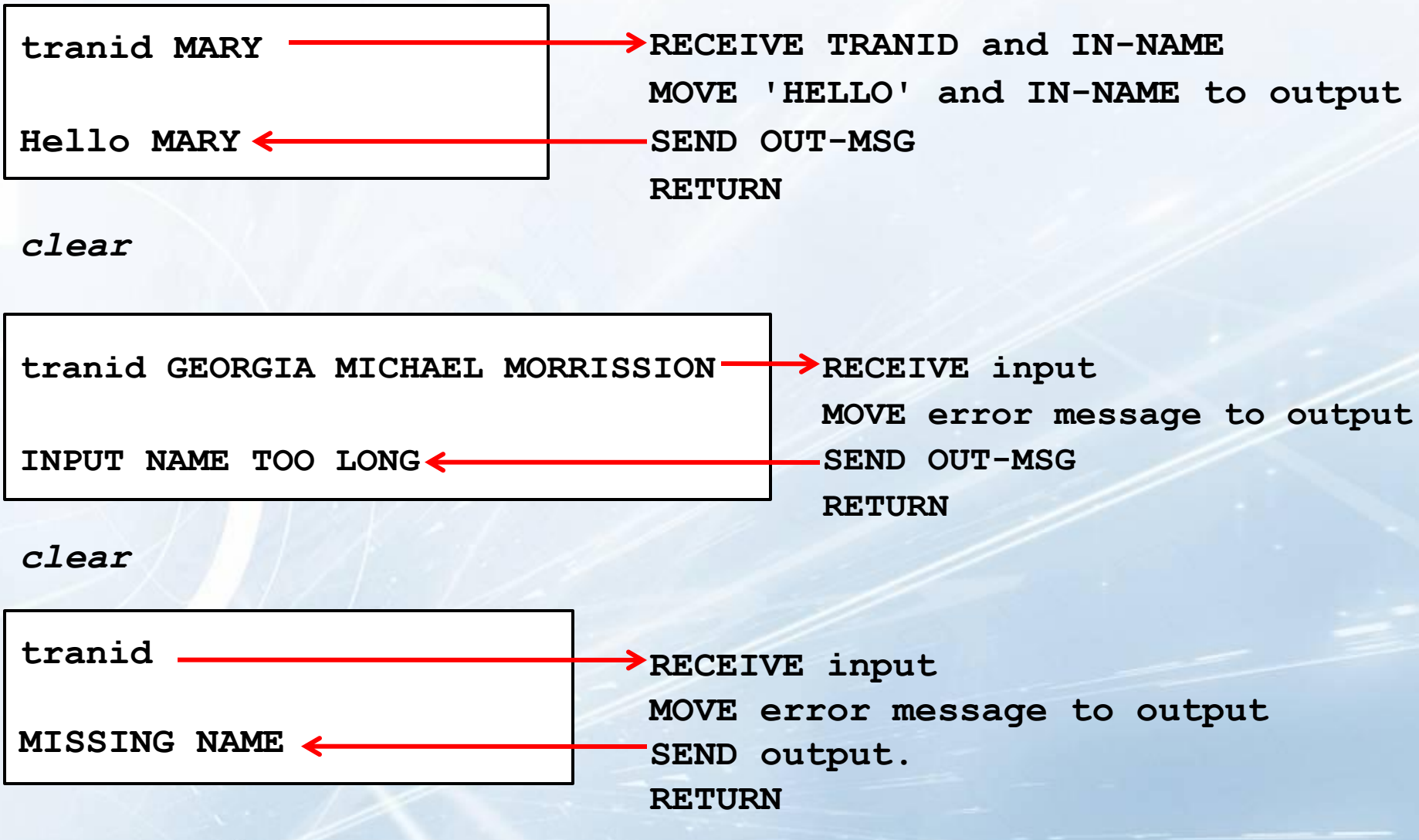
Task

Created by CICS to process a transaction

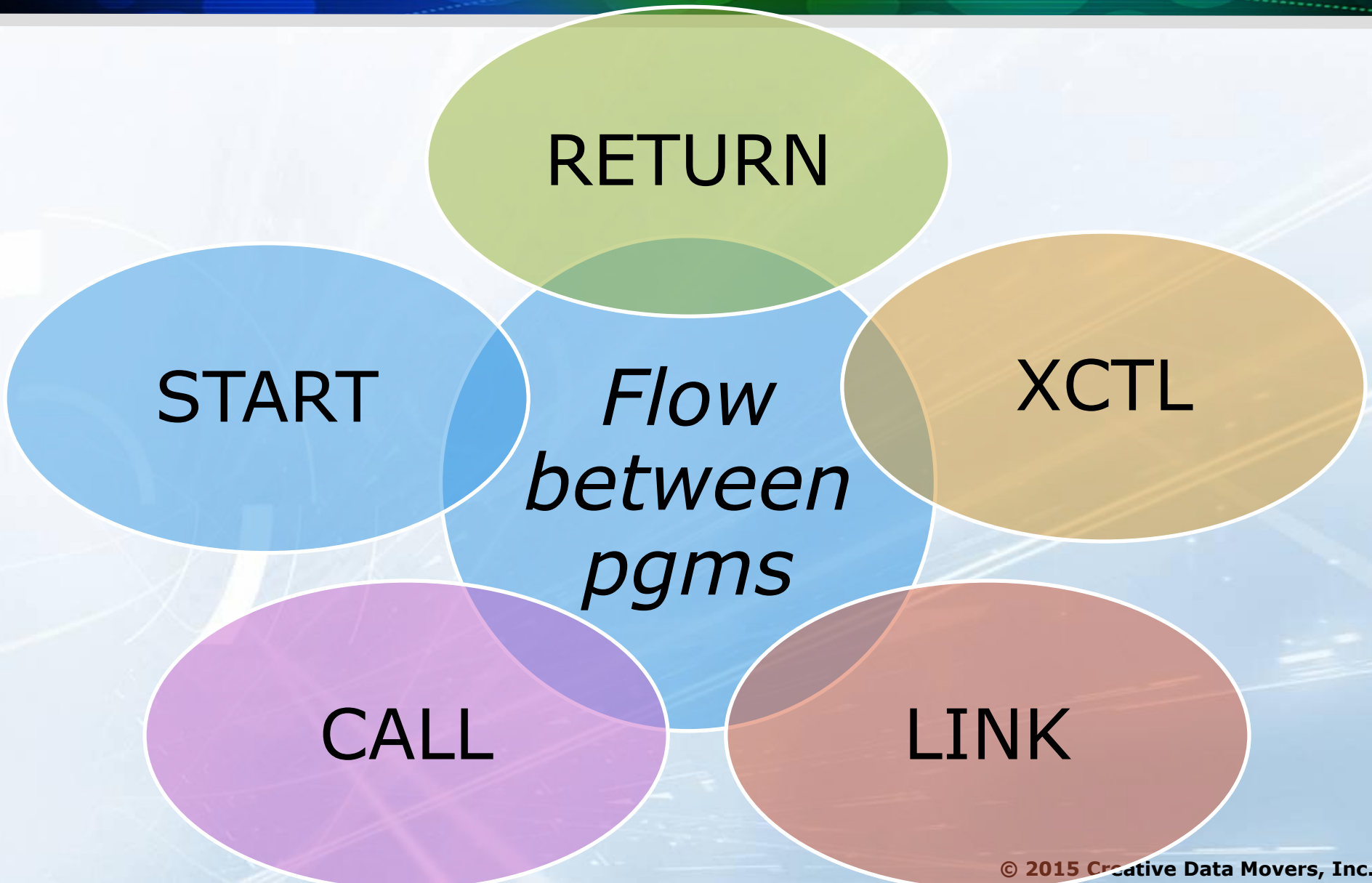
Pseudo-conversational Programming

Multiple tasks complete a transaction

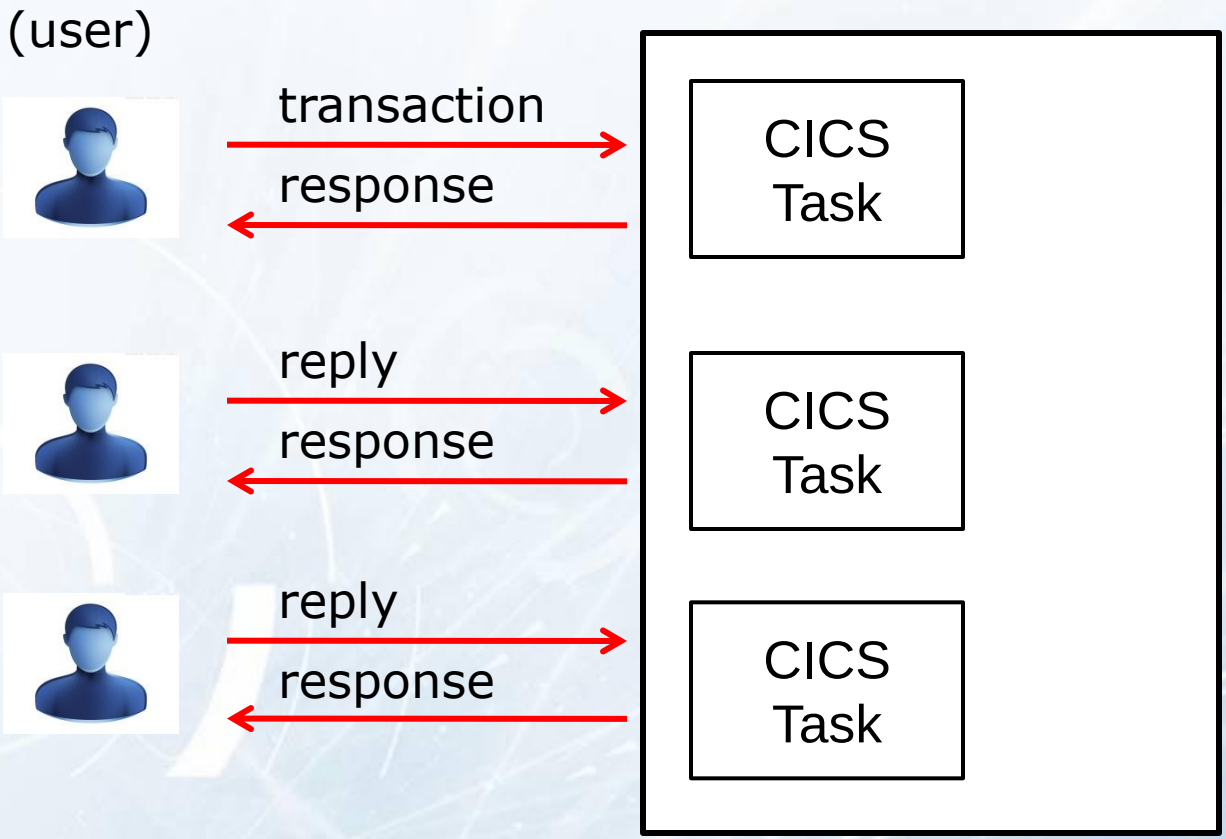
Non-Conversational Program Flow



Flow of Control



Pseudo-conversation



Pseudo-conversation completes the transaction

Pseudo-conversational Logic Flow

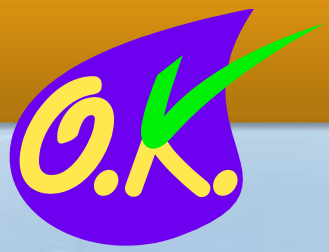
First time:

- Send the user the map so they can input their request
- RETURN with transid, AND a commarea

Subsequent times:

- RECEIVE the map and process the user's request
- SEND the user the map with their answer
- RETURN with transid AND a commarea

Last time:



- Determined when user presses CLEAR
- RETURN without transid

Pseudo-Code Framework for "Native"

```
procedure division.
   if eibaid = DFHCLEAR
      exec cics RETURN end-exec
   end-if

   if eibcalen > zero
      move DFHCOMMAREA to WS-COMMAREA
      perform receive-map-and-process
   end-if

   exec cics SEND map('MAP1')
                  MAPSET('ABCMSI')
                  ERASE
                  RESP(WS-RESP-CD)
   end-exec

   exec cics RETURN transid('CADD')
                    commarea(WS-COMMAREA)
                    length(LENGTH OF WS-COMMAREA)
                    RESP(WS-RESP-CD)
   end-exec
   goback
   .
```

First time test, if started from a trans code. Test is different when XCTL with commarea to a program. In that case, test if EIBTRNID matches that for the second program.

Pseudo-conversational program Logic

First time processing

- Send map and return (with transid and commarea)

Middle-time repeated processing

- Receive map (or test for function key pressed)
- Process input and prepare output
- Send map and return (with transid and commarea)

Last time

- Receive map, process, send map
- Return, with no transid

CICS Programs Are Essentially 3 Programs in 1

1

- The "first program" processes first-time logic.
- First-time is determined when EIBCALEN = 0 (different for XCTL). Program simply sends a map and returns with a transid and a commarea.

2

- The "second program" processes user input.
- When EIBCALEN is greater than zero (different for XCTL) the program receives map, processes input data, sends map with information and returns with a transid and a commarea

3

- The "third program" processes the user's request to end.
- For example, user presses CLEAR to terminate processing
- The program tests for this by testing EIBAID = DFHCLEAR

Determining which phase to process

- First time
  - For initial program (usually menu program)
    - If commarea length is zero. it's the first time

```
IF  EIBCALEN = 0  ...
```

  - After an XCTL, this does not work
    - If EIBTRN is not equal to program's trancode
    - If not equal, then first time
- Last time
  - User will indicate it is the last time

```
IF EIBAID = DFHCLEAR ...
```

  - No SEND; To end: code RETURN without transid
- Middle times
  - All the other times

HANDLE CONDITION vs RESPONSE

- Strongly recommend RESP(...)
- Include this phrase in every EXEC CICS

```
RESP(WS-RESP)
```

- Follow EXEC CICS with IF or EVALUATE

```
IF WS-RESP = DFHRESP(NORMAL)
   CONTINUE
ELSE
   MOVE 'ERROR SENDING MAPONLY MAP'
        TO WS-ERROR-LINE
   PERFORM 8000-COMMON-ERROR-RTN
END-IF
```

Commentary on RECEIVE MAP

```
EXEC  CICS  RECEIVE MAP('MAPADD')
            MAPSET('XXXX100')
            RESP(WS-RESP)
END-EXEC

IF WS-RESP = DFHRESP(NORMAL)
          OR  DFHRESP(EOC)
    PERFORM 3000-PROCESS-MAP
ELSE
    ...
END-IF
```

EOC is normal also

Two types of RETURN

- *Return to continue processing*

```
EXEC CICS RETURN
          TRANSID('TF00')
          COMMAREA(WS-COMMAREA)
          LENGTH(LENGTH OF WS-COMMAREA)
          RESP(WS-RESP)
END-EXEC
```

- *RETURN to end transaction*

```
EXEC  CICS
          RETURN
END-EXEC
```

Typical copybooks

- Values for PFKeys, Enter, Break, PA keys

```
COPY DFHAID.
```

- Values for attribute bits/bytes

```
COPY DFHBMSCA.
```

- Copybook for mapset/maps (symbolic maps)

```
COPY XXXXM00.
```

Stateful vs Stateless

- CICS programs are STATELESS
  - (Except for conversational programs)
  - CICS programs remember nothing.
  - Every run gets a fresh copy of Working-Storage.
- How to remember and have a conversation?
  - Commarea
  - Channels and containers
  - TS Queues (Temp storage)
  - VSAM files
  - DB2 tables
  - MQ
  - Hidden fields in sent/received maps
  - etc.

COMMAREA

- Use a copybook.
- Generally the same commarea for every program in the application
- Often a bit of extra space for possible future expansions
- Make as small as possible
  - Every user gets a copy of the commarea
  - All commarea's are stored in same CICS region

New CICS – no user I/O

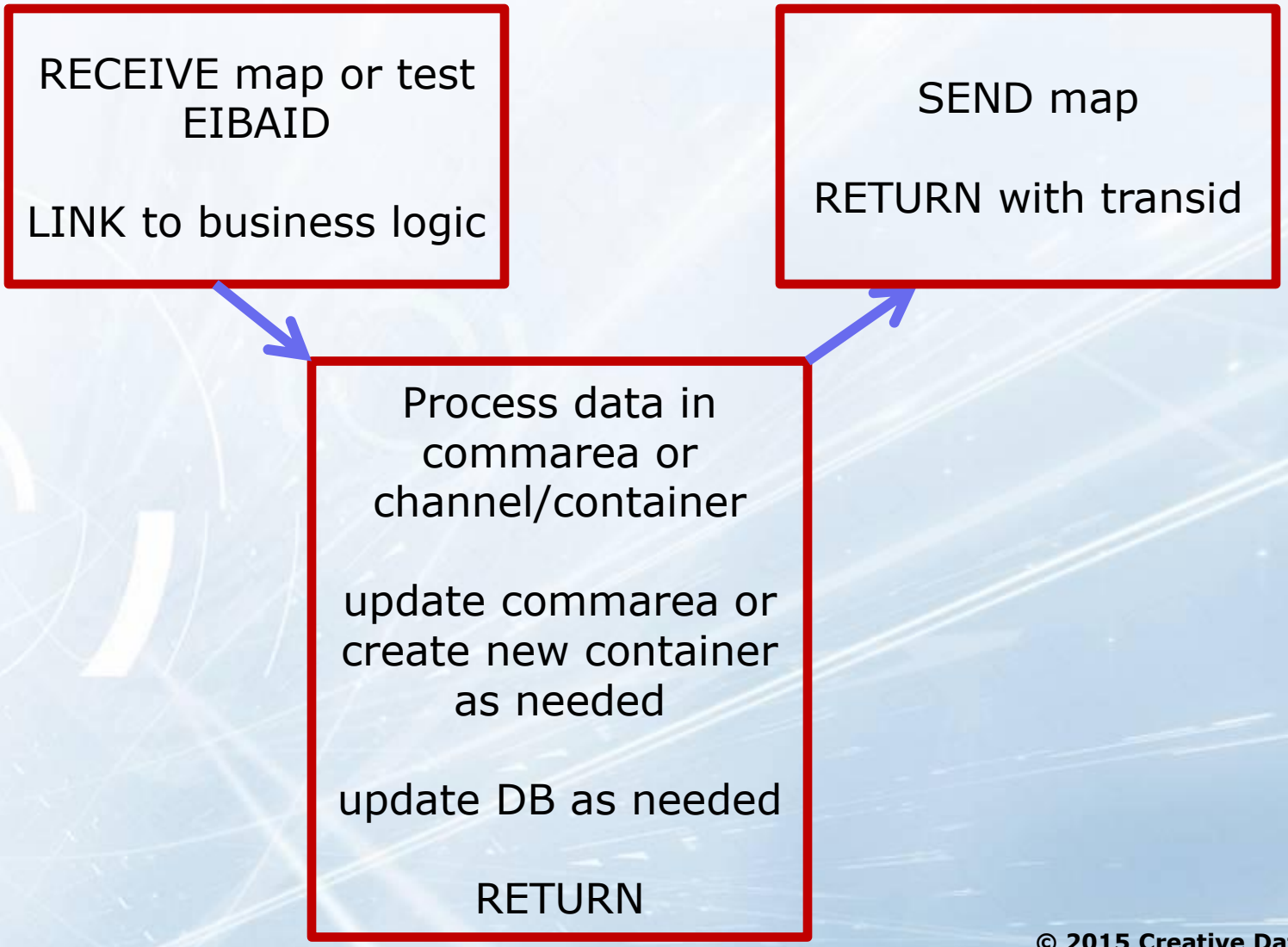
Separation of Concerns

- Often, the new CICS has no maps/mapsets
- But most build SEND/RECEIVE into their logic

- Put Control and User I/O in one program
- Put business logic in a second program
- Have the first program LINK to the second

- After testing, the second program can become a WEB SERVICE.

Main program and linked program



New ways to use CICS

- HATS replaces your MAPs with HTML and JavaScript
  - Maps are now GUI that appears in your browser
  - However, they are not the beautiful HTML that users expect to see in a browser
  - Don't send hidden fields, as now the user can see them
- Web Services
  - Single input, single output.
  - SOAP can be the vehicle enclosing your data
  - No maps/mapsets
  - User I/O is handled by the invoking program
  - Requires modularization/ separation of concerns

MQ Triggered CICS Programs

Characteristics

- No user I/O
- All you know, if your CICS program is triggered, is that there should be a message in a particular WS message queue

Logic

- Once thru code then end (or loop for more msgs)
- Get the message
- Process message
- RETURN (no transid or commarea)

Challenge to debug. Possibilities:

- Write trace messages to a TS queue, then browse
- CEDX (like CEDF, but from another logon)
- Xpediter, InterTest, Fault Analyzer, etc.

CICS Program Design Considerations

Follow the standards in place for your site

Use modular design with small functional programs

Structure to reduce maintenance but keep code concise

continued

CICS Program Design Considerations (cont.)

Minimize paging by avoiding unnecessary branching

Avoid high overhead instructions

Keep WORKING-STORAGE small

- If possible, use LINKAGE SECTION
- Code literals in PROCEDURE

Program Control

Structured programming divides applications into smaller, more manageable units

Program Control commands transfer between application programs in the CICS region

Program Control Commands require entries in the PPT

- XCTL goes to another program
- LINK performs a lower level program
- RETURN goes back up to a higher level program
- LOAD brings a load module (table) into the CICS region
- RELEASE deletes a loaded item
- CALL invokes a subprogram
- START transaction

RETURN Notes

While the transid on the RETURN can be any transid defined in a PCT, recommend using the same one as for your program

The CICS translator by default defines a one byte DFHCOMMAREA in the LINKAGE SECTION

Even if the transid specified causes the same program to be restarted, it has a FRESH COPY (all new) WORKING-STORAGE, etc.

LINK Format

```
EXEC CICS LINK
              PROGRAM ('pptname')
              [COMMAREA(dataname)]
              [LENGTH(halfword)]
              [DATALENGTH(halfword)]
              [INPUTMSG(dataname)]
                    [INPUTMSGLEN(halfword)]
              [SYSID(name)]
                    [SYNCONRETURN]
                    [TRANSID(transid)]
              [RESP (dataname)]
END-EXEC
```

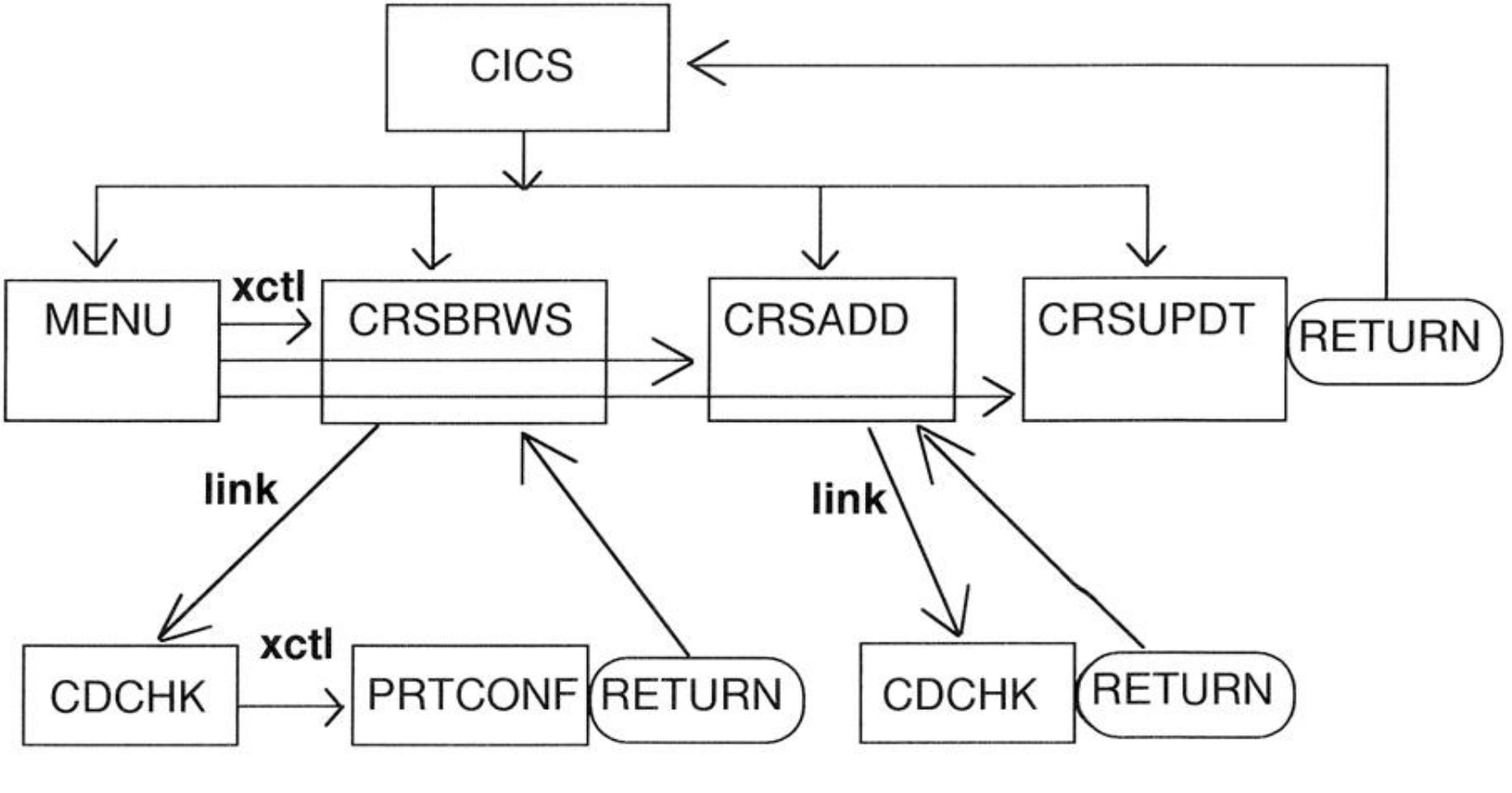
- Use GOBACK or "vanilla" RETURN to end the linked program

XCTL Format

```
EXEC CICS XCTL
              PROGRAM ('pptname')
              [COMMAREA(dataname)]
              [LENGTH(halfword)]
              [INPUTMSG(dataname)]
                 [INPUTMSGLEN(halfword)]
              [RESP (dataname)]
END-EXEC
```

XCTL transfers control to another program at the same logical level

XCTL and LINK in an Application



COBOL CALL

Like a LINK; often faster

CALL can be static or dynamic

Compile parm *must* be NODYNAM

CALLed programs may contain CICS commands

If CICS commands are used in sub-program - main program *must* pass DFHEIBLK and DFHCOMMAREA on the CALL, followed by other parameters

CALLed program can *only* be COBOL or Assembler in CICS/ESA

COBOL Call Examples

- Example of CALL Code in Main Program

```
CALL 'SUBPROG' USING DFHEIBLK
                     WS-COMMAREA
                     WS-FLD1
                     WS-FLD2
```

or:

```
CALL identifier USING DFHEIBLK
                      WS-COMMAREA
                      WS-FLD1
                      WS-FLD2
```

COBOL Call Examples (cont.)

- Example of Code in CALLed Program

```
LINKAGE SECTION.
01   DFHCOMMAREA---.
01   LS-FLD1---.
01   LS-FLD2---.

PROCEDURE DIVISION USING LS-FLD1, LS-FLD2.
```

Sharing Data Across Transactions

CICS features for sharing data across tasks and transactions

- CWA
- TWA
- TCTUA
- COMMAREA and EIB (Needed if passing to C.)
- Display Screen (Hidden or shown fields.)
- TS Queue
- TD Queue
- GETMAIN with SHARED option
- LOAD a shared table
- MQ Series

CWA - Common Work Area

Single block of data

Allocated at CICS startup

Exists until shutdown

Fixed size (specified in system initialization parameter - WRKAREA)

Almost no overhead

continued

CWA - Common Work Area (cont.)

Data *not* secured and *not* recoverable

Always use a copybook

Use for status information or other small amounts of data

To restrict write access to CWA, specify CWAKEY=CICS. Then only programs defined with EXECKEY(CICS) can write to CWA.

Program Control Summary

Program Control Commands tie application systems together

XCTL goes to another program at the same level

LINK effectively CALLs another program

continued

Program Control Summary (cont.)

CALL can be coded to programs with *or* without CICS commands

CALLs to programs with CICS commands *must* pass the EIB block and DFHCOMMAREA

SYNCPOINT Format

```
EXEC CICS SYNCPOINT
                  [ROLLBACK]
END-EXEC
```

SQL Commit - NOT

Do *not* code EXEC SQL COMMIT or ROLLBACK

Use EXEC CICS SYNCPOINT END-EXEC to commit changes to recoverable CICS, VSAM, and DB2 resources

Use EXEC CICS SYNCPOINT ROLLBACK END-EXEC to back out all changes to recoverable resources