

# Customer Information Control System

Lesson 00:

People matter, results count.



Copyright © Capgemini 2015. All Rights Reserved 1

©2016 Capgemini. All rights reserved.  
The information contained in this document is proprietary and confidential.  
For Capgemini only.

## Document History

Date	Course Version No.	Software Version No.	Developer / SME	Change Record Remarks
23-Jan-2006	CICS ver 2.2	NA	Vandana Mistry	Content creation
20-Nov-2009	CICS ver 2.3	NA	Vandana Mistry	Content Updation
14-Dec-2009	CICS ver 2.4	NA	CLS Team	Review
30-June-2011	CICS ver 3	NA	Rajita Dhumal	Content Updation
4-June-13	CICS ver 3.1	NA	Rajita Dhumal	Content Revamp as per New ELTP 60 Days Curriculum
2-Aug-16	CICS Ver 3.2	NA	Veena K	Revamp, post the integration



Copyright © Capgemini 2015. All Rights Reserved 2

## Course Goals and Non Goals

- Course Goals
  - Develop On-Line COBOL/CICS programs.
  - Execute CICS Programs.
  
- Course Non Goals
  - CICS maintenance and tuning



Copyright © Capgemini 2015. All Rights Reserved 3

## Pre-requisites

- COBOL
- Job Control Language (JCL)
- Multiple Virtual Storage (MVS)
- Virtual Storage Access Method (VSAM)



Copyright © Capgemini 2015. All Rights Reserved 4

## Intended Audience

- This course is intended for developers who want to develop CICS application programs



## Day Wise Schedule

- Day 1

- Lesson 1: Introduction to CICS
- Lesson 2: System Components
- Lesson 3: Data Communication Operation in CICS

- Day 2

- Lesson 3: Data Communication Operation in CICS... contd
- Lesson 4: Characteristics of CICS

- Day 3:

- Lesson 5: Command Level Programming



Copyright © Capgemini 2015. All Rights Reserved 6

## Day Wise Schedule

### ■ Day 4

- Lesson 6: CICS Copy Books
- Lesson 7: Application Program Housekeeping
- Lesson 8: BMS Programming Considerations
- Lesson 9: Terminal Control Operations
- Lesson 10 : Interval Control Operations

### ■ Day 5

- Lesson 11 : Program Control Operations
- Lesson 12: File Control Operations (Random Access)



Copyright © Capgemini 2015. All Rights Reserved 7

## Day Wise Schedule

- Day 6
  - Lesson 12: File Control Operations (Random Access)... contd
- Day 7
  - Lesson 13: File Control Operations (Sequential Access)
  - Lesson 14: Temporary Storage Control
  - Lesson 15: Transient Data Control
- Day 8
  - Lesson 16: Tests and Debugging
  - Lesson 17: CICS WITH DB2
  - Lesson 18: Introduction to CICS Web Services



Copyright © Capgemini 2015. All Rights Reserved 8

## Table of Contents

- Lesson 1: Introduction to CICS
  - 1.1: Batch and Online systems
    - 1.1.1: Advantages of the Online System
    - 1.1.2: Advantages of the Batch System
  - 1.2: Interactive Programs
    - 1.2.1: Data Entry Program
    - 1.2.2: Inquiry Program
    - 1.2.3: File Maintenance Program
    - 1.2.4: Menu Program
  - 1.3: Interactive System Considerations
    - 1.3.1: Shared Files (File Integrity)
    - 1.3.2: Response Time
    - 1.3.3: Security
    - 1.3.4: Recovery



Copyright © Capgemini 2015. All Rights Reserved 9

## Table of Contents

- 1.4: Elements of a Data Communication Network
  - 1.4.1: Host System
  - 1.4.2: Communication Controller
  - 1.4.3: Modem
  - 1.4.4: Communication Lines
  - 1.4.5: Terminal System
- 1.5: Introduction To CICS
- 1.6: Features of CICS
  - 1.6.1: DB/DC
  - 1.6.2: Multiprogramming
  - 1.6.3: Multitasking
  - 1.6.4: Multithreading
  - 1.6.5: Re-entrant
- 1.7: Transaction
- 1.8: Initiation of a Task



Copyright © Capgemini 2015. All Rights Reserved 10

## Table of Contents

- Lesson 2: System Components
  - 2.1: Management Modules
    - 2.1.1: File Control Program Management Module (FCP)
    - 2.1.2: Terminal Control Program Management Module (TCP)
    - 2.1.3: Program Control Program Management Module (PCP)
    - 2.1.4: Task Control Program Management Module (KCP)
    - 2.1.5: Storage Control Module
    - 2.1.6: Interval Control Module
    - 2.1.7: Journal Control Module



Copyright © Capgemini 2015. All Rights Reserved 11

## Table of Contents

- 2.2: Tables
  - 2.2.1: Program Control Table (PCT)
  - 2.2.2: Processing Program Table (PPT)
  - 2.2.3: Terminal Control Table (TCT)
  - 2.2.4: File Control Table (FCT)
  - 2.2.5: System Initialization Table
  - 2.2.6: Sign On Table (SNT)
- 2.3: Control Blocks



Copyright © Capgemini 2015. All Rights Reserved 12

## Table of Contents

- Lesson 3: Data Communication Operation In CICS
  - 3.1: Methods for Data Communication
  - 3.2: Characteristics of 3270 display
    - 3.2.1: Attribute Byte Format
  - 3.3: Basic Mapping Support
    - 3.3.1: Format Independence
    - 3.3.2: Device Independence
  - 3.4: Maps and Mapsets
    - 3.4.1: Map Generation
  - 3.5: Physical and Symbolic Map
    - 3.5.1: Physical Map
    - 3.5.2: Symbolic Map
      - 3.5.2.1: Format of Symbolic Map
      - 3.5.2.2: Example of Symbolic Map Definition
      - 3.5.2.3: Example of Customized Symbolic Map Definition



Copyright © Capgemini 2015. All Rights Reserved 13

## Table of Contents

- 3.6: Components of a Screen
  - 3.6.1: Modified Data Tag (MDT)
- 3.7: Map Definition Macros
  - 3.7.1: Format of BMS Macros
  - 3.7.2: Example of BMS Macros
  - 3.7.3: Order of Map-Code
    - 3.7.3.1: Print NOGEN
    - 3.7.3.2: END
- 3.8: DFHMSD Macro
  - 3.8.1: Format of DFHMSD Macro
- 3.9: DFHMDI Macro
  - 3.9.1: Format of DFHMDI Macro
- 3.10: DFHMDF Macro (Function)



Copyright © Capgemini 2015. All Rights Reserved 14

## Table of Contents

- Lesson 4: Characteristics of CICS
  - 4.1: Multithreading
  - 4.2: Quasi-Reentrancy
  - 4.3: Conversational Programming
  - 4.4: Pseudo-Conversational Programming
    - 4.4.1: How does a Pseudo-Conversational Program work?
  - 4.5: Passing the Data using COMMAREA



Copyright © Capgemini 2015. All Rights Reserved 15

## Table of Contents

- Lesson 5: Command Level Programming
  - 5.1: CICS Program Development
  - 5.2: CICS Command Format
  - 5.3: Command Language Translator
  - 5.4: COBOL/CICS Program Structure
- Lesson 6: CICS Copy Books
  - 6.1: Exec Interface Block (EIB)
  - 6.1.1: EIB Fields



Copyright © Capgemini 2015. All Rights Reserved 16

## Table of Contents

- 6.2: DFHAID- Attention Identifier (AID)
  - 6.2.1: DFHAID
  - 6.2.2: Using AID information in a program (Example)
- 6.3: DFHBMSCA - Standard Attribute Byte List
- 6.4: FACDEFN
  
- Lesson 7: Application Program Housekeeping
  - 7.1: ABEND Codes
  - 7.2: Exception Handling
  - 7.3: Handle Condition



Copyright © Capgemini 2015. All Rights Reserved 17

## Table of Contents

- 7.4: IGNORE Condition
- 7.5: NOHANDLE Option
- 7.6: HANDLE AID Command
- 7.7: EIBAID
  
- Lesson 8: BMS Programming Considerations
  - 8.1: DFHBMSCA – Standard Attribute Byte List
  - 8.2: FACDEFN



Copyright © Capgemini 2015. All Rights Reserved 18

## Table of Contents

- 8.3: Cursor Positioning Techniques
  - 8.3.1: Static Cursor Positioning
  - 8.3.2: Symbolic Cursor Positioning
  - 8.3.3: Relative Cursor Positioning
- Lesson 9: Terminal Control Operations
  - 9.1: Functions of CICS Command for BMS
  - 9.2: RECEIVE MAP Command
  - 9.3: SEND MAP Command
  - 9.4: RECEIVE Command (Unformatted Data Transfer)
  - 9.5: SEND TEXT Command (Unformatted Data Transfer)



Copyright © Capgemini 2015. All Rights Reserved 19

## Table of Contents

- Lesson 10: Interval Control Operations
  - 10.1. Introduction
  - 10.2. ASKTIME Command
  - 10.3. FORMATTIME Command (Function)
- Lesson 11: Program Control Operations
  - 11.1: Program Control Operations
  - 11.2: Application Program Levels
  - 11.3: RETURN Command
  - 11.4: LINK Command



Copyright © Capgemini 2015. All Rights Reserved 20

## Table of Contents

- 11.5: XCTL Command
- 11.6: XCTL versus LINK
- 11.7: Exceptional Conditions
- 11.8: Data Passing through COMMAREA
  
- Lesson 12: File Control Operations (Random Access)
  - 12.1: Supported Access Methods
  - 12.2: Commands for File Handling
  - 12.3: Special Services of File Control
  - 12.4: READ Command Using Full Key



Copyright © Capgemini 2015. All Rights Reserved 21

## Table of Contents

- 12.5: READ Command Using GENERIC Option
- 12.6: READ Command Using GTEQ Option
- 12.7: WRITE Command
- 12.8: READ/UPDATE and REWRITE Commands
- 12.9: UNLOCK Command
- 12.10: DELETE Command
  
- Lesson 13: File Control Operations (Sequential Access)
  - 13.1: Introduction
  - 13.2: STARTBR Command



Copyright © Capgemini 2015. All Rights Reserved 22

## Table of Contents

- 13.3: Start Browse at the Beginning of the File
- 13.4: Start Browse at a Specific Record
- 13.5: Start Browse at the End of the File
- 13.6: READNEXT Command
- 13.7: READPREV Command
- 13.8: READNEXT/READPREV-NOTE
- 13.9: Changing Direction of Browse
- 13.10: ENDBR Command
- 13.11: RESETBR Command
- 13.12: Multiple Browse Operations



Copyright © Capgemini 2015. All Rights Reserved 23

## Table of Contents

- Lesson 14: Temporary Storage Control
  - 14.1: Temporary Storage Control
  - 14.2: Characteristics of TSQ
  - 14.3: TSQ in MAIN Storage
  - 14.4: TSQ in Auxiliary Storage
  - 14.5: Naming Convention for TSQ Queue Id
  - 14.6: TSQ Access
  - 14.7: Item and Item numbers
  - 14.8: Commands used in Temporary Storage Control
  - 14.9: Write Queue
  - 14.10: WRITEQ TS Command with REWRITE Option
  - 14.11: Read Queue
  - 14.12: DELETEQ TS Command



Copyright © Capgemini 2015. All Rights Reserved 24

## Table of Contents

- Lesson 15: Transient Data Control
  - 15.1: Introduction
  - 15.2: Types of TDQ
    - 15.2.1: Intra-partition TDQ
    - 15.2.2: Extra-partition TDQ
  - 15.3: WRITEQ TD
  - 15.4: READQ
  - 15.5: DELETEQ TD Command
  - 15.6: Intra-partition TDQ
  - 15.7: Automatic Task Initiation (ATI)
  - 15.8: Indirect Destination



Copyright © Capgemini 2015. All Rights Reserved 25

## Table of Contents

- Lesson 16: Tests and Debugging
  - 17.1: Commands for Tests and Debugging
  - 17.2: Abend Exit
  - 17.3: HANDLE ABEND Command
  - 17.4: ABEND Command
- Lesson 17: CICS with DB2



Copyright © Capgemini 2015. All Rights Reserved 26

## References

- List as bullets references (books, URL)
  - CICS Handbook By YUKIHISA KAGEYAMA
  - CICS for the Cobol Programmer by Doug Luw



Copyright © Capgemini 2015. All Rights Reserved 27

## Next Step Courses

- DB2



Copyright © Capgemini 2015. All Rights Reserved 28

## Other Parallel Technology Areas

- ADS/O



Copyright © Capgemini 2015. All Rights Reserved 29

# **Customer Information Control System**

Lesson 1: Introduction to CICS

## Lesson Objectives

- In this lesson, you will learn about:
  - Batch and Online systems
  - Interactive Programs
  - History of CICS
  - Features of CICS
  - Task and Transaction Initiation



1.1: Batch and Online Systems

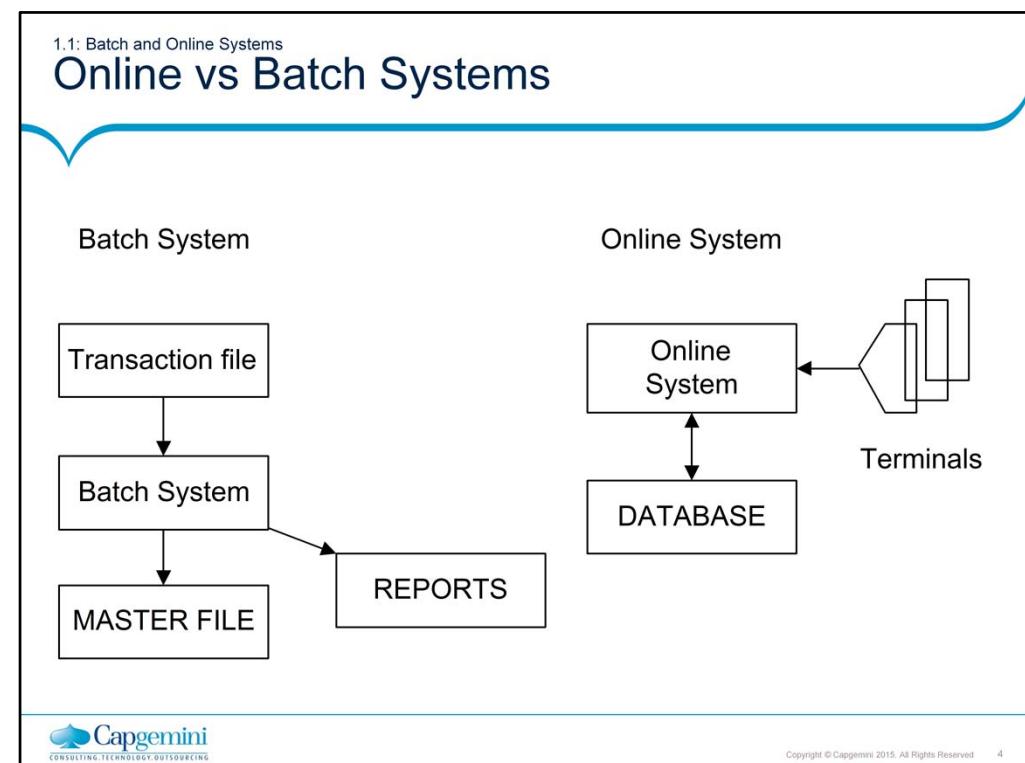
## Features in Batch and Online Systems

- Following is a list of features in Batch and Online systems:

Feature	Batch	On-Line
Data Collection	Off-Line	On-Line
Input	Sequential In Batch	Random
Job Schedule	At specific intervals	Instantaneous
Resources	Not Sharable	Sharable
Response Time	Not Critical	Critical
Output	In Printed Format	On the Terminal
Security	Simple	Complex
Recovery	Simple	Complex
Information	Not always current	Always current
Updation	In Batch	Immediate

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3



1.1: Batch and Online Systems

## Advantages Of Online System

- These days, the online system is so common that virtually every large mainframe installation has at least one online application system.
- This is because the online system has advantages over the batch systems.
  - Up-to-date file (information) can be shared among many users simultaneously and instantly.
  - Data validation and editing can be done at data entry time.



Copyright © Capgemini 2015. All Rights Reserved

5

1.1: Batch and Online Systems

## Advantages Of Batch System

- If the information need not be updated on a real time basis, then batch system can be used.
- When massive file updates have to be done, they should be done in batch mode. This is because online processing of these areas can be resource intensive.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

### Advantages of the Batch System:

- On the other hand, in spite of the ever-increasing popularity of the online system, the batch system has not died out. It still contributes to the large portion of data processing requirements. This is because the batch system has advantage over the online system. Some of the advantages of the batch system are as follows:
  - Certain information does not have to be updated or displayed on a real-time base. Users can wait until the next day, the end of a week, or the end of a month. In this case, the batch system is sufficient for the purpose.
  - If massive file updates or lengthy calculations are involved, then the batch system should be used because the online processing in these areas tends to be very costly in terms of resource consumption.

1.2: Interactive Programs

## Concept Of Interactive Programs

- An online system also called as an interactive system is a collection of computer programs that lets end users access mainframe computers via terminal devices.
- Following are the four types of application programs:
  - Data Entry Program
  - Inquiry Program
  - File Maintenance Program
  - Menu Program

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

1.2: Interactive Programs

## Data Entry Programs

- The operator enters the data for one transaction at the terminal.
- The program updates any related master files.
- The program writes the transaction to the transaction file.

```
graph LR; Terminal[Terminal] -- 1 --> DEP[Data entry programs]; DEP -- 2 --> MasterFile[Master file]; DEP -- 3 --> TransFile[Trans file]
```

The flowchart illustrates the Data Entry Programs process. It starts with a 'Terminal' box on the left, which has an arrow labeled '1' pointing to a 'Data entry programs' box in the center. From this central box, two arrows labeled '2' and '3' point to two circular cylinders representing files: 'Master file' above and 'Trans file' below. The 'Master file' cylinder is positioned higher than the 'Trans file' cylinder.

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 8

1.2: Interactive Programs

## File Maintenance Programs

- The operator enters the record key.
- The program reads the appropriate record from the master file.
- The program displays the record at the terminal.
- The operator enters any required changes to the record.
- The program rewrites the record to the master file.

```
graph LR; Terminal[Terminal] -- 1 --> FM((File maintenance)); FM <-- 2 --> MF((Master File)); Terminal <-- 3 --> FM; FM -- 4 --> Terminal; FM <-- 5 --> MF;
```

The diagram shows three main components: a rectangular box labeled "Terminal", a cylinder labeled "File maintenance", and another cylinder labeled "Master File". Arrows numbered 1 through 5 indicate the flow of data: 1) from Terminal to File maintenance; 2) from File maintenance to Master File; 3) from Terminal to File maintenance; 4) from File maintenance back to Terminal; and 5) from File maintenance to Master File.

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 9

1.2: Interactive Programs

## Menu Programs

- The menu program sends a list of selections to the terminal.
- The operator chooses one of the selections.
- Control is transferred to the appropriate program.

```
graph LR; Terminal[Terminal] <-->|1| MP[Menu program]; MP -->|2| Terminal; MP -->|3| P1[Program 1]; MP -->|3| P2[Program 2]; MP -->|3| P3[Program 3]
```

The diagram illustrates the interaction between a Terminal and a Menu program. The Terminal and Menu program are connected by a double-headed arrow labeled '1'. The Menu program is also connected to the Terminal by a single-headed arrow labeled '2'. Additionally, the Menu program is connected to three separate boxes labeled 'Program 1', 'Program 2', and 'Program 3' by single-headed arrows labeled '3'.

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 10

1.3: Interactive System Considerations

## Factors Affecting Interactive Systems

- There are a few considerations of an online system that a programmer must contend with, while designing and developing these programs:
  - Shared Files
  - Response Time
  - Security
  - Recovery



Copyright © Capgemini 2015. All Rights Reserved 11

1.3: Interactive System Considerations

## Significance Of Shared Files

- A batch program typically has complete control of the file it uses. So there is no chance that another program can interfere with its processing.
- In an interactive system, many terminal operators use the same system simultaneously, and all must have access to the file they require.
- When an interactive system lets users shared files, it must coordinate all file updates to insure file integrity such that no two users are able to update the same record at the same time.



Copyright © Capgemini 2015. All Rights Reserved 12

1.3: Interactive System Considerations

## Significance Of Response Time

- Response time is the amount of time for which a user must wait while a transaction is being processed.
- Many factors affect the response time, namely:
  - The number of users on the system
  - The size of the CPU
  - The speed of disk drives
  - The speed of communication lines
  - The manner in which the application programs are written



Copyright © Capgemini 2015. All Rights Reserved 13

1.3: Interactive System Considerations

## Significance Of Security

- The main security technique used in today's interactive system is the logon procedure.
- In addition to the logon procedure, most interactive systems have a multilevel security system that allows some users to access certain files.
- Each user is allowed to access only those files s/he is authorized to use.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 14

**Security:**

CICS provides simple security features and many CICS installations use other security management systems that provide tighter control. Although security is an important consideration in an online system, it does not affect the way application programs are developed.

1.3: Interactive System Considerations

## Significance Of Recovery

- All interactive systems must provide for recovery in the event of a system failure.
- Recovery in an interactive system should be properly planned, since it involves hundreds of users.



Copyright © Capgemini 2015. All Rights Reserved 15

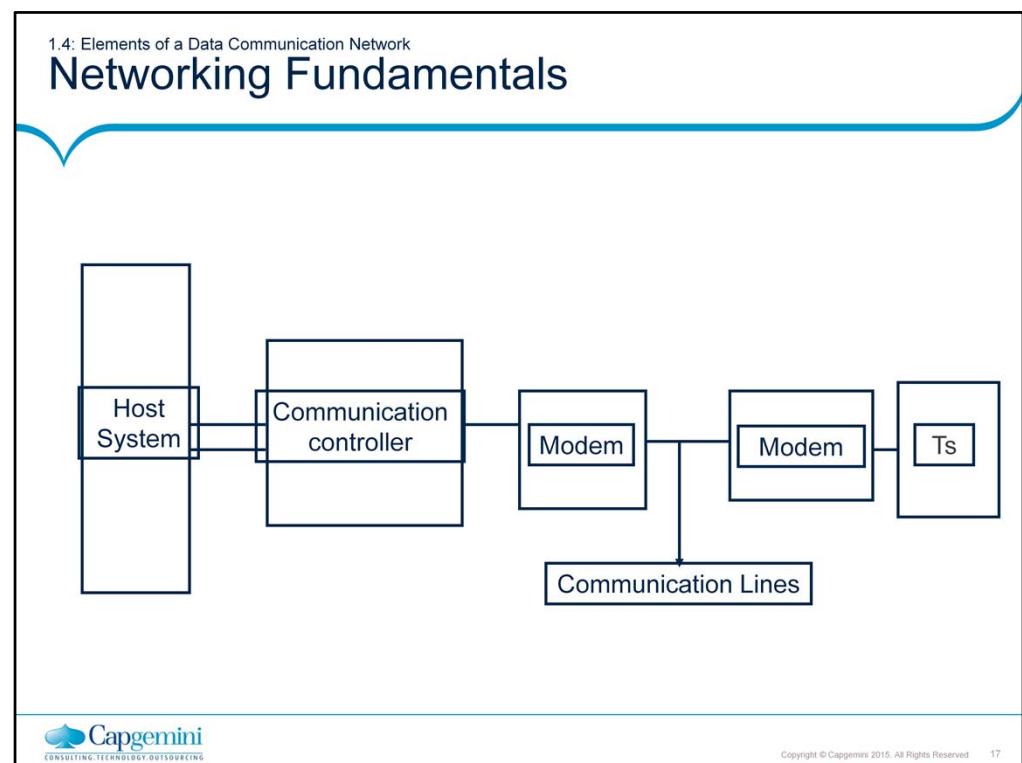
1.4: Elements of a Data Communication Network

## Networking Fundamentals

- A data communication network comprises the following components:
  - Host System
  - Communication Controller
  - Communication Lines
  - Modem
  - Terminal controller
  - Terminals

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16



1.4: Elements of a Data Communication Network

## Significance Of A Host System

- To control communication systems, the host system uses telecommunication access methods.
- The common methods are as follows:
  - BTAM (Basic Telecommunication Access method)
  - TCAM (Telecommunication Access Method)
  - VTAM (Virtual Telecommunication Access method)



Copyright © Capgemini 2015. All Rights Reserved 18

1.4: Elements of a Data Communication Network

## Significance Of Communication Controller

- The modem of the Telecommunication line is connected to the host system via a communication controller.
- The communication controller controls the functions necessary to control the transmission of data over the communication line.
- One of the main functions of communication controller is data conversion. It implies converting data into a form that the modem can process.
- Another main function of a communication controller is data-link control.
  - Data-link control ensures successful transmission of data over the communication line.
  - Some of the functions required for data-link control are:
    - Synchronizing the host with the terminal
    - Identifying the source and destination of transmission
    - Detecting and correcting transmission errors
- Popular models of communication controller are 3704 and 3705.



Copyright © Capgemini 2015. All Rights Reserved 19

1.4: Elements of a Data Communication Network

## Significance Of Modem

- The function of the modem is to:
  - Convert digital signals to audio frequency signals that can be sent over the phone line
  - Reconvert audio frequency signals back into digital signals that can be processed by the computer system



Copyright © Capgemini 2015. All Rights Reserved 20

1.4: Elements of a Data Communication Network

## Significance Of Communication Lines

- One of the major components of a communication network is a phone connection, often called a communication line.
- The TC line can be set through private telephone system or may be privately owned.



Copyright © Capgemini 2015. All Rights Reserved 21

1.4: Elements of a Data Communication Network

## Significance Of A Terminal System

- A terminal system consists of a terminal controller plus one or more display stations (Terminals).
- A popular terminal system in M/F is 3270.



Copyright © Capgemini 2015. All Rights Reserved 22

1.5: Introduction to CICS

## The CICS Concept

- CICS acts as an interface between the Operating system and Application programs.

```
graph TD; AP[Application Program] <--> CICS[CICS]; CICS <--> OS[Op. System]
```

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 23

1.5: Introduction to CICS

## The CICS Concept

- CICS is an interface between application program in an interactive system and the host OS.
- The application program communicates with CICS.
- In turn CICS communicates with access method through the host OS
  - Then the access methods (such as VTAM,BTAM,VSAM or ISAM) communicate with the system devices (such as terminals, disk drives, or tape drives.)

```
graph LR; subgraph OS [Operating system]; AP[Application programs] --> CICS[CICS]; CICS --> DA[Data access<br/>(VSAM, DB2, IMS)]; CICS --> CA[Communication access<br/>(VTAM, SNA, TCP/IP)]; end; DA --> DS[Disk storage]; CA --> UI[User interface]
```

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 24

1.5: Introduction to CICS

## The CICS Concept

- As far as the operating system is concerned, CICS is an application program.
- CICS runs as a JOB in one of the system's partitions or regions.
- The region or partition may be a real address space but more often is a virtual address space.
- CICS behaves like an OS within an OS.
- CICS is given the most favored status in an installation by making it non-swappable and is given the highest priority.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 25

1.5: Introduction to CICS

## CICS and OS Scheduling

- CICS runs in one 'region' or 'partition' or 'address space' and handles scheduling for programs running under it.
- Since CICS is itself a batch job, it is scheduled by the OS and will compete with other batch jobs for CPU time. Therefore CICS is usually run with high priority.

**OS/390**

Copyright © Capgemini 2015. All Rights Reserved. 26

1.6: Features of CICS

## Salient Features

- Following are the prime features available in CICS:
  - DB/DC
  - Multiprogramming
  - Multitasking
  - Scheduling



Copyright © Capgemini 2015. All Rights Reserved 27

1.6: Features of CICS

## Significance Of DB/DC

- Since CICS controls both database operations as well as data communication operations it is often called as DB/DC system.



Copyright © Capgemini 2015. All Rights Reserved 28

1.6: Features of CICS

## Significance Of Multiprogramming

- Multiprogramming means that the operating system allows several programs to execute at the same time.
- The key to understanding multiprogramming is that at any given time only one program can have the control of the CPU.
- As far as the operating system is concerned, CICS is an application program. It implies that CICS runs as a job in one of the system's partition or region, which is a virtual address space.



Copyright © Capgemini 2015. All Rights Reserved 29

1.6: Features of CICS

## Significance Of Multiprogramming

- Therefore CICS takes part in multiprogramming environment of the operating system.
- CICS is a longer running job. Therefore it remains up during the day time collecting on-line transactions, and at night time, it executes batch jobs for master file updation, etc.



Copyright © Capgemini 2015. All Rights Reserved 30

1.6: Features of CICS

## Significance Of Multitasking

- Multitasking implies that a program running in a single partition or region allows multiple tasks to execute simultaneously.
- A task is an execution of a programs or programs for a specific user.
  - Example: If user1 is running PGM A, then user1 has created a task.
- Multitasking is the same as multiprogramming, however, it is one level down.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 31

1.6: Features of CICS

## Multiprogramming and Multitasking

DOS/VSE	
Supervisor Area	
Background	COBOL compiler
Foreground 4	CICS Test partition
Foreground 3	Payroll application
Foreground 2	Unused
Foreground 1	CICS production partition
	User 1 Order Entry
	User 2 Customer Enquiry
	User 3 Order Entry
	User 4 Inventory Inquiry
	User 5 Customer File maintenance
	User 6 Order Entry

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 32

1.6: Features of CICS

## Multitasking

- CICS can accept input from many terminals, and when the currently executing task completes, or has to wait for file I/O, there is another task ready to execute.
- A task is the execution of an application program for a specific user and CICS handles multitasking internally, within its own address space.
- CICS decides which ready-to-run task is to be given control in a process known as task dispatching. The highest priority ready task is dispatched.

**CICS address space**

Task 1 Order entry program (user 1)	Task 2 Customer inquiry program (user 2)	Task 3 Master menu program (user 3)	Task 4 Customer maintenance program (user 4)	Task 5 Order entry program (user 5)
--	--	--	--	--

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 33

1.6: Features of CICS

## Significance Of Multithreading

- In the above figure user1, user3, and user6 are accessing the same application program: Order entry.
- This would waste valuable storage space if the same program was loaded into storage at three different storage locations.
- Under CICS, however, a concept called multithreading is used so that only one copy of the program is loaded into the storage.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 34

**Multithreading:**

Multithreading means that the area of storage containing a program is not allocated to a specific user. Instead all users running the program have access to the same storage locations.

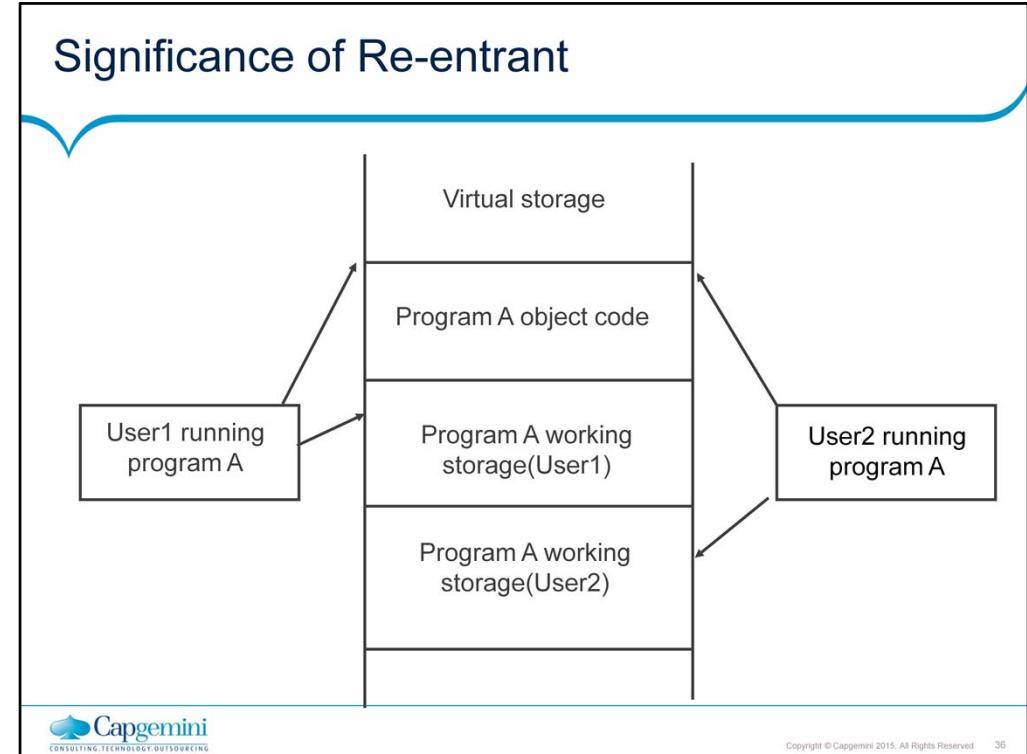
1.6: Features of CICS

## Significance Of Re-entrant

- For multithreading to work, a program must be re-entrant.
  - A re-entrant program cannot modify data in working storage.
  - As a result a user can enter a re-entrant program at any point without affecting the users who are running it.



Copyright © Capgemini 2015. All Rights Reserved 35

**Re-entrant:**

- In the example in the above slide, the two users share the same application program, that is PROGRAM A.
- They share the same storage for the program's object code, that is the procedure division, but each is given a separate working storage area. This way each uses the working-storage in a normal fashion.
- A Quasi-reentrant program is a reentrant program under CICS environment. That is a Quasi-reentrant program is a CICS program which does not modify itself.
- When you write a command-level CICS program, you do not have to worry about making it quasi-reentrant. CICS automatically handles that for you.

1.7: Transaction

## Concept Of Transaction

- A transaction is a pre-defined unit of work that a user can invoke.
- When a transaction is invoked, a specific application program is loaded into storage (if it is not already in storage) and a task is started.
- The difference between a task and transaction is that while several users may start the same transaction, each will be given its own task.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 37

Examples of Transaction:

- Displaying a record on screen
- Entering a customer's order into the system
- Entering fields needed for computation
- Receiving the result of that computation

Each CICS transaction requires many tasks

Multiple users can invoke the same transaction but a different task is created for each user

1.7: Transaction

## Concept Of Transaction

- Each transaction is identified by a four-character code called transaction identifier (transaction id).
- An operator initiates a transaction by entering the transaction id on to the terminal.
- A special CICS table, called the program control table (or PCT), defines each transaction.
- Each trans-id is paired with the name of the program CICS will load and execute when the transaction is invoked.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 38

1.7: Transaction

## Concept Of Transaction

- Another CICS table, called the processing program table (or PPT), contains a list of all valid program names.
- The PPT indicates each program's location – a storage address if the program has already been loaded or a disk location if the program has not been loaded.
- CICS uses PPT to determine whether it will load a new copy of the program when the transaction is invoked.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 39

1.8: Initiation of a Task

## Concept Of Initiation Of A Task

- The operator starts a transaction by entering a transaction-id.
- CICS searches the PCT to find the program to be executed.
- CICS then searches the PPT to determine the location of the program.
- If the program is not currently in storage, then the disk location is returned.



Copyright © Capgemini 2015. All Rights Reserved 40

1.8: Initiation of a Task

## Concept Of Initiation Of A Task

- Then CICS locates the program on the DISK and loads it into storage and a task is initiated.
- If the program is already in storage, then the object code for the program is NOT retrieved from the disk unit.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

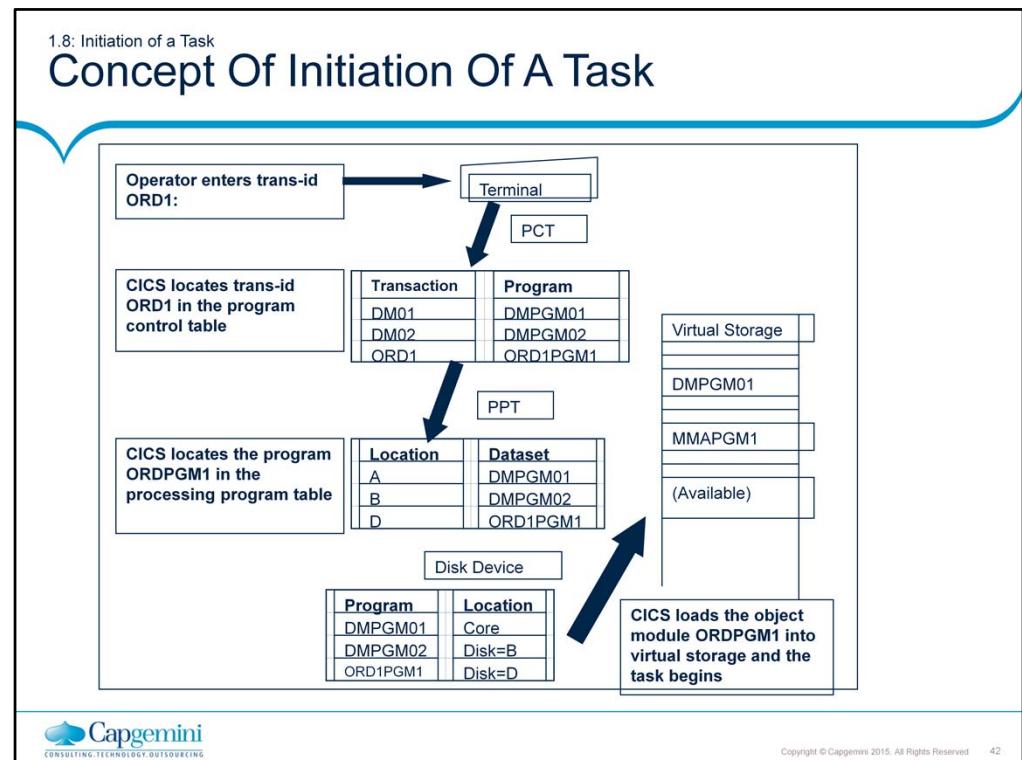
Copyright © Capgemini 2015. All Rights Reserved. 41

### What is a task?

Most elementary unit of work to CICS by OS.

Task is initiated when CICS starts executing the program

Task continues until the program gives up control to CICS and 'dies'



## Summary

- In this lesson, you have learnt about:
  - Features of Batch and Online System
  - Interactive systems
  - Features of CICS
  - Concepts of Transaction and Task



## Review Questions

- Question 1: CICS searches \_\_\_\_ to determine the location of the program.
- Question 2: \_\_\_\_ is the amount of time a user must wait while a transaction is being processed.
- Question 3: A \_\_\_\_ is an execution of a programs or programs for a specific user.



# **Customer Information Control System**

Lesson 2 : System Components

## Lesson Objectives

- In this lesson, you will learn about:
  - The Management Modules of CICS
  - Tables



2.1: Management Modules

## Concept Of Management Modules

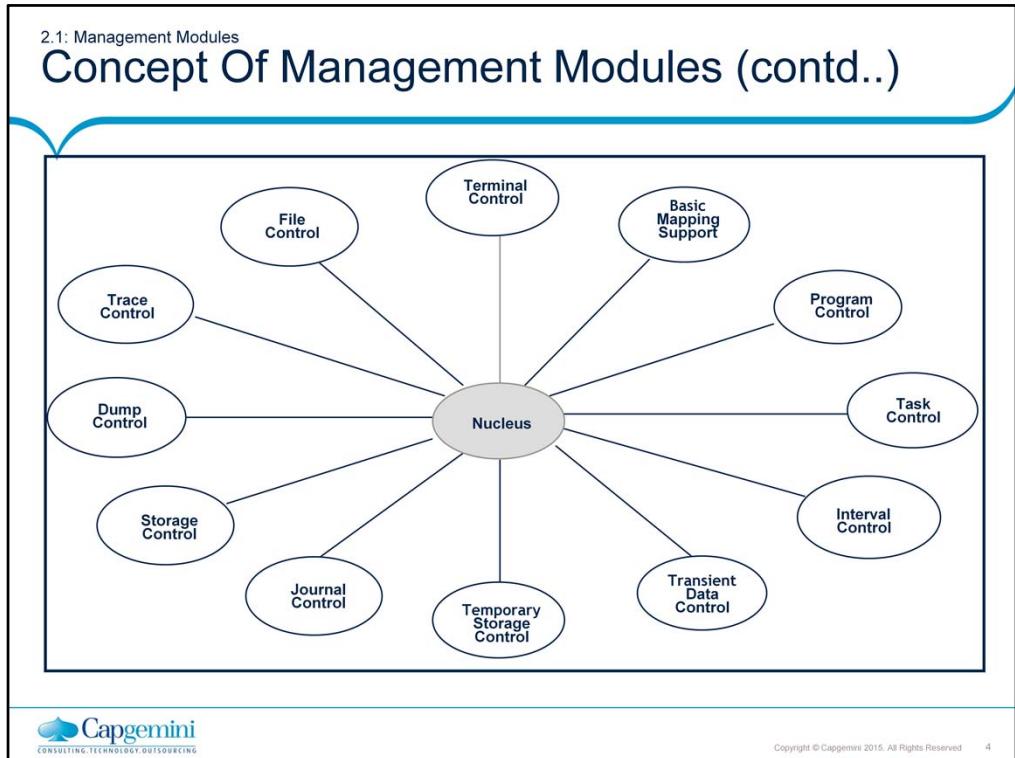
- The CICS/MVS system consists of:
  - Management Modules
  - Tables
  - Control Blocks
- Management modules are the CICS/MVS programs that interface between the operating system and the application program
- Each management module performs a particular function

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

**Introduction to Management Modules:**

**Example:** When an application program issues a request to read record, the File Control Program management module processes the request. Input/output requests are made to CICS/MVS, instead of the operating system, as they are in batch processing environment. CICS takes over the difficult part of I/O operations, leaving only the execution logic to the application program.



### Introduction to Management Modules (contd.):

At the center of the figure is the nucleus of the CICS. The nucleus handles general functions that are crucial to the operation of CICS – functions such as multitasking, multithreading, security, and recovery. The functions of the nucleus are automatic.

2.1: Management Modules

## Control Programs and Control Tables

■ Control programs	Control Tables
TCP (Terminal control program)	TCT (Terminal control table)
SCP (Storage control program)	PCT (Program control table)
KCP (Task control program)	PPT (Proc. program table)
TDP (Transient data program)	TST (Temp. storage table)
FCP (File control program)	DCT (Destination control table)
TSP (Temp. storage program)	JCT (Journal control table)
JCP (Journal control program)	
PCP (Program control program)	

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

2.1: Management Modules

## Concept Of FCP

- FCP manages file processing activities of each application program.
- FCP makes use of a table called File Control Table (FCT).
- FCT includes an entry for each file that is to be used during the operation of CICS.
- It frees application programmer from defining the physical organization and other file attributes.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

### File Control Program Management Module (FCP):

- The file control module manages the file processing activities of each application program.
- A command level CICS COBOL program does all file processing through CICS file control requests.
- Facilities like OPEN, READ, and WRITE is not allowed.
- When the file control receives a request, it passes it on to one of the host operating system's access methods such as (QSAM, ISAM, or VSAM) which in turn controls the system's secondary storage devices.

2.1: Management Modules

## Concept Of FCP

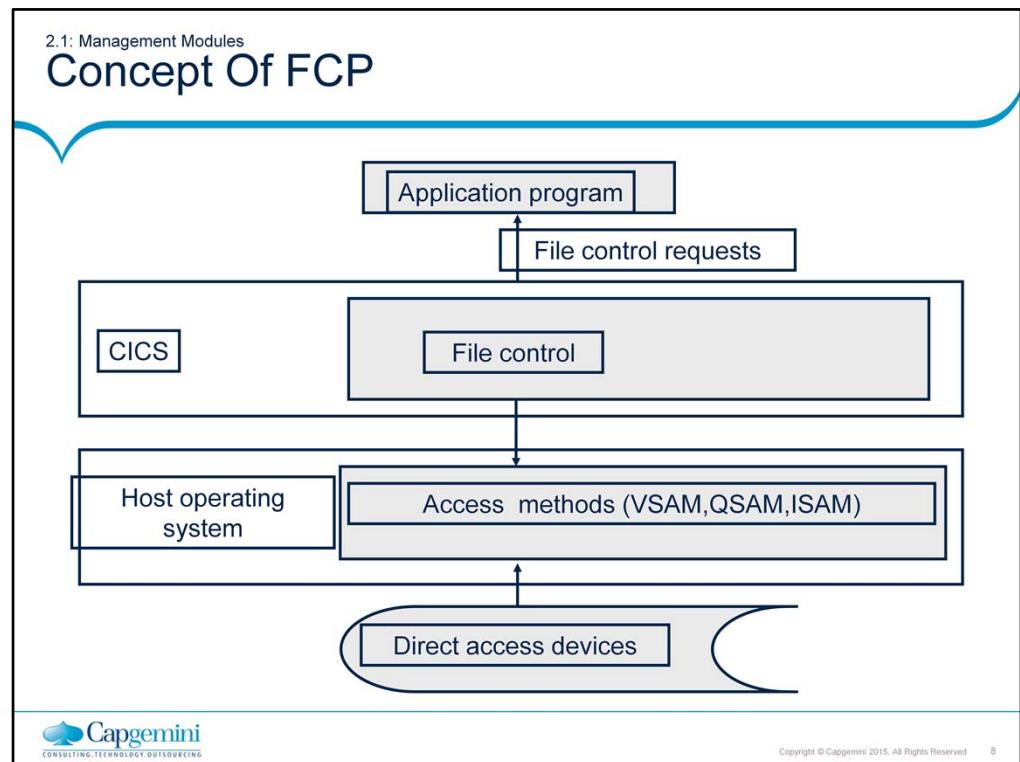
- The file control module simplifies the file processing code in a program.
- The following statements need not be coded:
  - OPEN, CLOSE
  - SELECT and FD
- The record description is coded in the working storage section.
- The appropriate file control commands are coded in the procedure division.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

### File Control Program Management Module (FCP):

- FCT includes an entry for each file that is to be used during the operation of CICS.
- Each entry in the FCT contains all the descriptive information for the file that it represents, thereby freeing the application programmer from defining the physical organization and other file attributes.
- The application programmer simply refers to the symbolic name that identifies the file entry in the FCT.



2.1: Management Modules

## Concept of TCP

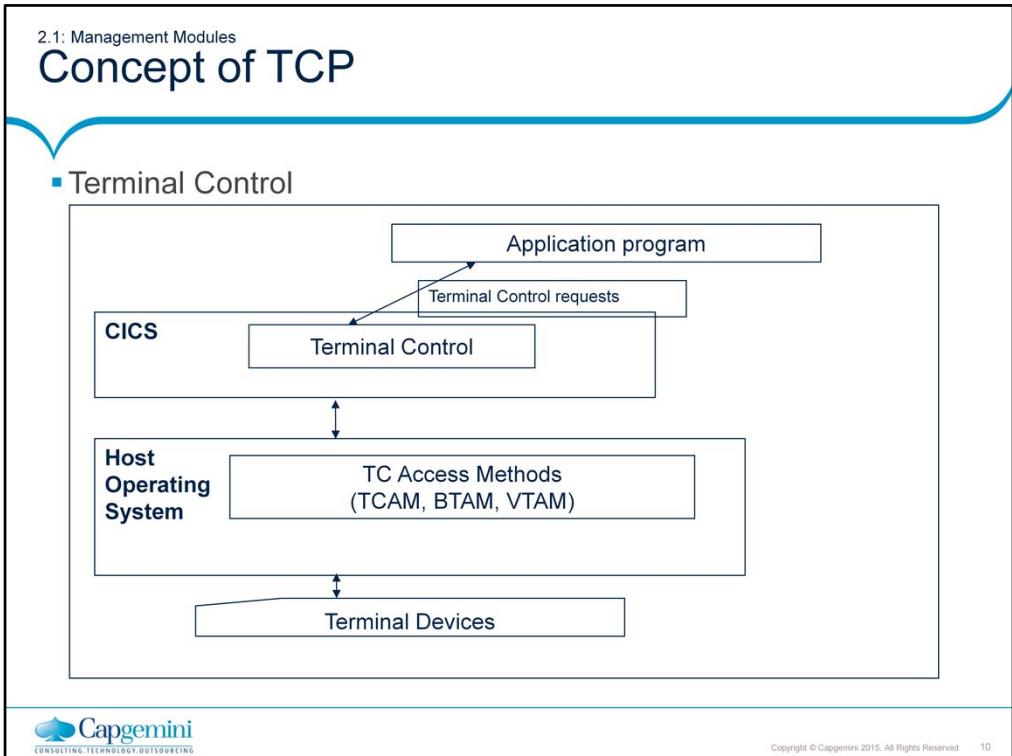
- TCP is a management module.
  - It handles communication between online application program and terminals by means of terminal control commands.
    - Example: Send and Receive
  - It allows to send text or receive text from the terminal that initiated the task.  
- Terminal control is difficult to use directly.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 9

### Terminal Control Program Management Module (TCP):

- TCP handles communication between online application program and the telecommunication access method of the operating system like VTAM, BTAM.
- TCP is CICS interface to telecommunication access methods (VTAM/BTAM).
- It transfers data between the application program and the terminal.
- Terminal control is difficult to use directly since an application program that uses terminal control directly must process:
  - Complicated strings of control characters
  - Data sent to and received from the terminal



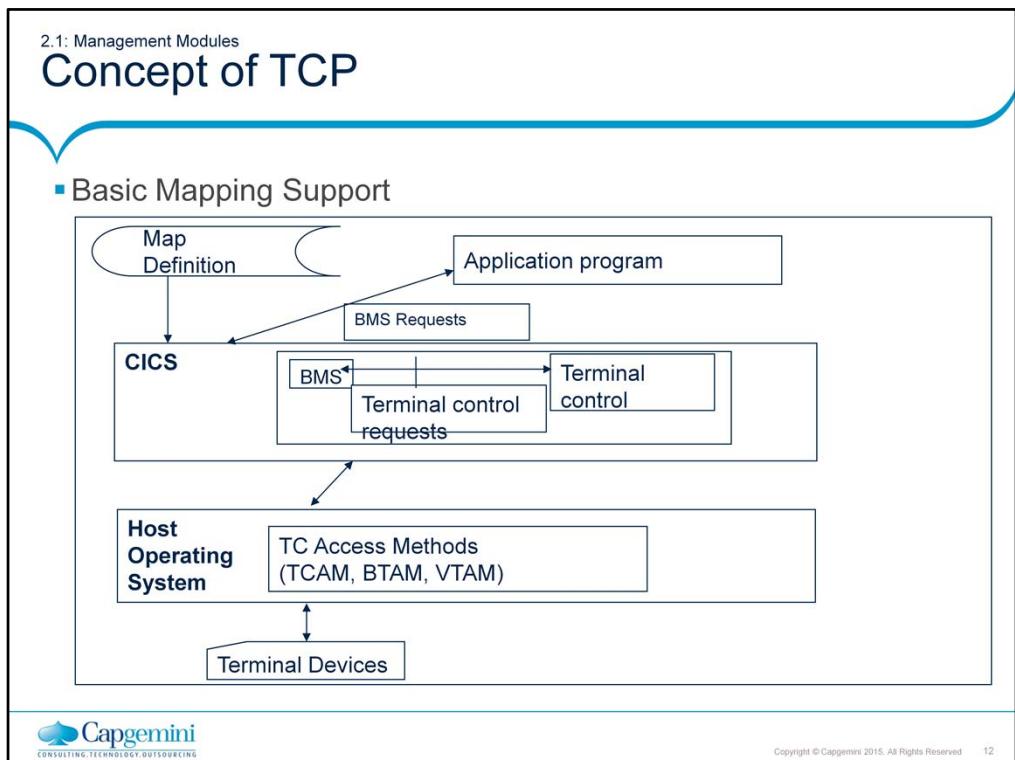
2.1: Management Modules

## Concept of TCP

- To relieve the programmer of the task of building and decoding complex string of characters and data, Basic Mapping Support (or BMS) was developed.
- BMS (a part of CICS) is an interface between the application program and the terminal control.
- To receive from or send data to a terminal, an application program issues a BMS request.
- After BMS retrieves the specified map definition, it formats a terminal control that is valid for the type of terminal that initiated the task and that matches the map definition.



Copyright © Capgemini 2015. All Rights Reserved 11



2.1: Management Modules

## TCP Functions

- Let us see some of the functions done by TCP:
  - It requests terminals to send their transactions.
  - It transfers data between the application program and the terminal.
  - All data transfers are requested by the application program but are executed by TCP.
  - It handles hardware communications requirements.
  - TCP is responsible for the actual transmission of a message over communication lines.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 13

2.1: Management Modules

## TCP Functions

- It searches the Terminal Control Table (TCT).
- Each terminal in the network is identified by a unique entry in the Terminal Control Table.
- The entry is called Terminal Control Table Terminal Entry (TCTTE).
- When data is ready to be sent to the terminal, a flag is set ON in the terminal's TCTTE.
- TCP searches the TCT periodically and causes the data to be sent to the respective terminals.
- It requests initiation of a new task.
- All functions carried out by TCP are transparent to application program.
- Programmer simply needs to code RECEIVE and SEND commands.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 14

2.1: Management Modules

## Concept Of PCP

- Under CICS, a task may consist of more than one application program.
- For example, the application program executed when a task is begun may transfer control to a different program, which in turn may invoke yet another program.
- The program control module manages the flow of control from one program to another within a task.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 15

2.1: Management Modules

## PCP Functions

- PCP manages the flow of control within task.
- It is responsible for:
  - Locating application program and if necessary load them into storage for execution.
  - Transferring control to the program and returning control back to CICS.
- PCP also facilitates one application program transferring control to another application program via LINK and XCTL commands.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 16

**Program Control Program Management Module (PCP):**

- The validity of transaction-id supplied by the user is checked from the PCT by Program Control Program management module (PCP).
- In carrying out its program control functions, PCP makes use of a table called the Processing Program Table (PPT). There is an entry in the table (PPT) for each application program along with its name, host language, and address.
- PCP knows if there are any tasks currently using the same program, from the value of USE COUNTER in PPT.
- If USE COUNTER is zero, then the storage is made available for other purpose.

2.1: Management Modules

## Concept Of KCP

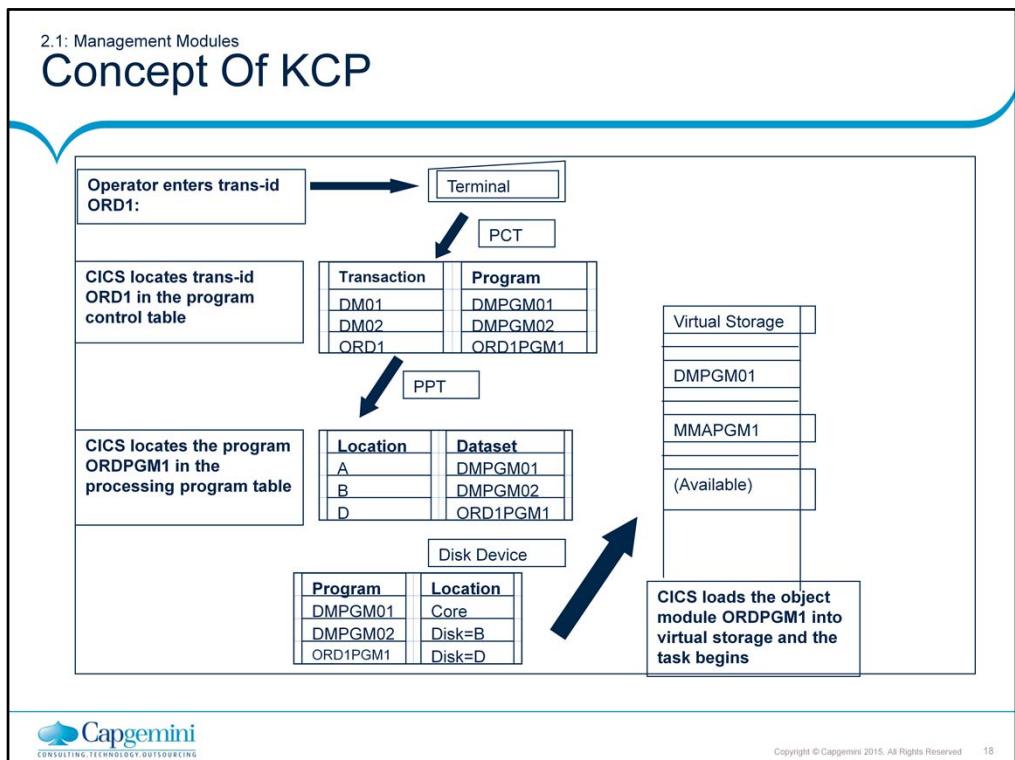
- KCP creates a task for the transaction requested.
  - It checks the validity of the transaction-id and the authority of the operator.
  - It allocates processor time among the tasks.
  - It keeps control of the status of all CICS tasks being processed concurrently.
- Transaction request can be made either:
  - By entering transaction Id at the terminal
  - By another CICS transaction

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 17

### Task Control Program Management Module (KCP):

- Task Control keeps control of the status of all CICS tasks. Many of them will be processed concurrently and task control allocates processor time among them. This is called multitasking.
- When an operator requests a transaction, normally by logging on and keying in a transaction code, CICS checks the status of the operator and the terminal. This ensures that the operator is known to the system and that the transaction is valid for that user and the terminal. Task Control then creates a task for that transaction.
- Transaction request can be made either by entering transaction-id at the terminal or by another CICS transaction.
- A transaction is processed up to an instruction involving input from a file or a terminal, for instance. Then while the transaction waits for its input, another waiting task begins or resumes execution.



2.1: Management Modules

## Storage Control Module

- The Storage Control Module allocates storage space to application programs.
- This process is automatic and there are very few application programs where we need to manage storage control functions.



Copyright © Capgemini 2015. All Rights Reserved 19

2.1: Management Modules

## Storage Control Program (SCP)

- SCP maintains full control over the virtual storage.
- It acquires, controls, and releases dynamic storage during execution of a task.
- Allocation of main storage is automatic.
- It controls requests for the main storage.



Copyright © Capgemini 2015. All Rights Reserved 20

## 2.1: Management Modules

## Interval Control Module

- Interval Control Module allows you to implement time dependent applications.
- It provides a way of starting a task that is an alternative to keying in a trans-id.
- Using the interval control functions you can specify that a task be started at a specific time or after a specific time interval has passed.



Copyright © Capgemini 2015. All Rights Reserved 21

2.1: Management Modules  
**Journal Control Module**

- Journal Control Module provides a standardized method of creating sequential files, called journals, which are used to restore master files in the event of a system failure.



Copyright © Capgemini 2015. All Rights Reserved 22

2.2: Tables

## Concept of Tables

- Tables define the resources that are controlled by CICS.
- Each table is associated with corresponding management modules.
- The tables are brought into storage when CICS is initialized.
- Example:
  - Terminal control table defines the terminals in the network.
  - File control table describes the data files that are accessed by the CICS program.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 23

2.2: Tables

## Concept of Tables

- Some of the important tables in CICS/MVS are:
  - FCT: File Control Table
  - PCT: Program Control Table
  - PPT: Processing Program Table
  - TCT: Terminal Control Table
  - DCT: Destination Control Table
  - SIT: System Initialization Table
  - SNT: Sign On table

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 24

2.2: Tables

## Concept of PCT

- A special table, called the PCT or Program Control Table, defines each transaction.
- PCT contains list of valid transaction identifiers.
- Each trans-id is paired with the name of the program.
- CICS will load and execute when the transaction is invoked.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 25

2.2: Tables

## Concept of PCT

- The PCT contains the control information to be used by CICS for identifying and initializing a transaction.
- This table is required by CICS to verify incoming requests to start transactions and to supply information about the transaction.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 26

2.2: Tables

## Concept of PPT

- Another CICS table called Processing Program Table (PPT), contains a list of valid program names.
- The PPT indicates each program's location – a storage address if the program has already been loaded or a disk location if the program has not been loaded.
- CICS uses PPT to determine whether it will load a new copy of the program when the transaction is invoked.
- The PPT defines the programs and mapsets.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 27

2.2: Tables

## Concept of TCT

- The Terminal Control Table (TCT) describes a configuration of terminals, logical units, and other CICS systems.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 28

2.2: Tables

## Concept of FCT

- The File Control Table (FCT) describes files that are processed by file control management module.
- The CICS system definition (CSD) file is defined in the FCT, in addition to your own files.
- The files defined in the FCT can be VSAM or BDAM.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 29

2.2: Tables

## System Initialization Table

- The System Initialization Table (SIT) contains parameters used by the system initialization process.
- In particular, the SIT identifies (by suffix character) the version of the CICS system control programs and CICS tables that you have specified and that are to be loaded.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 30

2.2: Tables

## Concept of SNT

- The Sign-On Table (SNT) contains a list of all authorized CICS users.
- For each user, a one to three character operator identification, an operator name, and an optional password are supplied.
- To access the system, one needs to know his/her operator name and password.
- In addition, the SNT provides a security key that is used to determine the resources that are available to each user.



Copyright © Capgemini 2015. All Rights Reserved 31

2.3: Control Blocks

## Concept of Control Blocks

- Control Blocks are dynamically created by CICS when needed.
- They are accessed by CICS modules during execution of tasks.
- They are released when no longer needed.
- For example:
  - A control block called Task Control Area (TCA) represents each transaction.
  - Information about current task is stored into control block named Execute Interface Block (EIB).



Copyright © Capgemini 2015. All Rights Reserved 32

2.3: Control Blocks

## Flow of Transaction

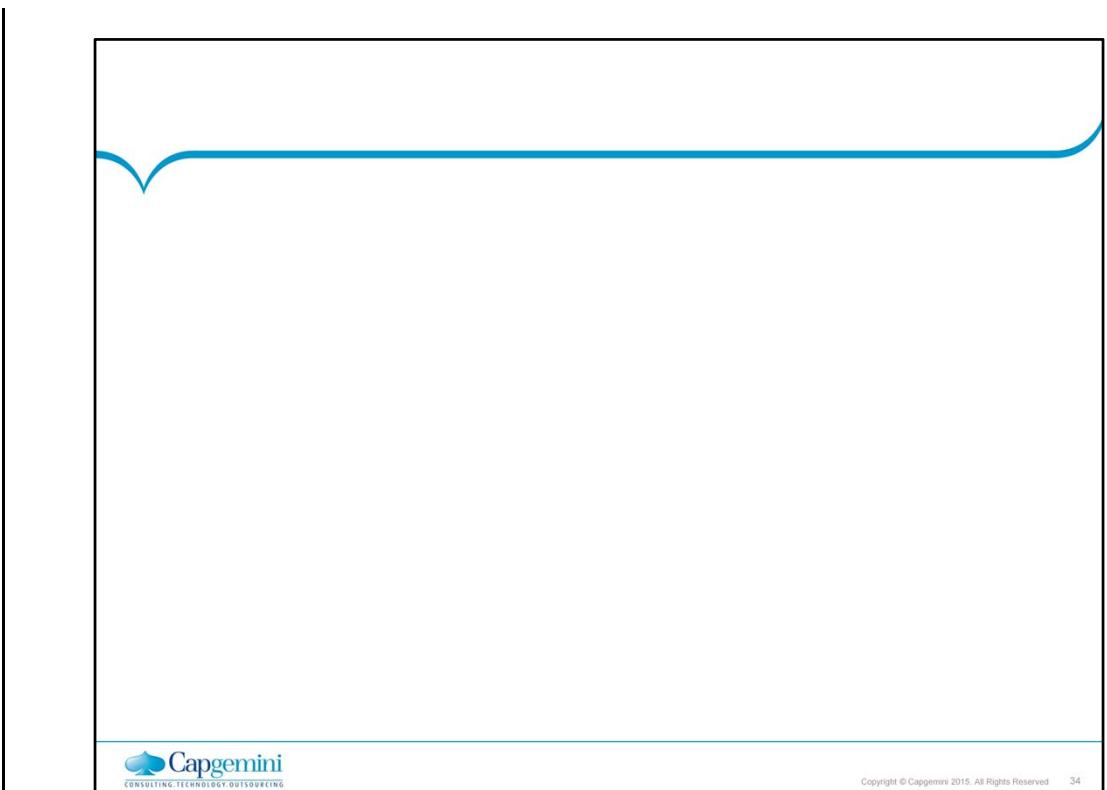
- Case: INQY 1234
- Inquiry transaction against a dataset to read a Part # 1234
  - Step-1. Read Input data from terminal
  - Step-2. Start the transaction
  - Step-3. Process the input
  - Step-4. Read the record
  - Step-5. Prepare the output
  - Step-6. End the transaction
  - Step-7. Write the output

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 33

### Step-1. Read Input data from terminal

- TCP gets control
- Validates Terminal Existence from TCT.
- Thru SCP acquires Terminal Input Output Area (TIOA) in the Dynamic Storage Area (DSA).
- Updates TCTTE with address of TIOA.
- Requests initiation of task thru KCP.
- Step-2. Start the transaction
- KCP validates transaction by searching PCT.
- Acquires any areas (TWA) thru SCP.
- Creates Task Control Area (TCA) and assigns a task sequence number.
- Links TCTTE & TCA with pointers.
- Passes control to program control.
- Step-3. Process the input
- PCP checks PPT for load module.
- Loads the program in DSA.
- Creates a copy of working storage in DSA and links to TCA.
- Application program gets control & reads input data from TIOA.



Copyright © Capgemini 2015. All Rights Reserved. 34

**Step-4. Read the record**

- Application program calls FCP
- FCP checks FCT, reads the record for part # 1234 from the file.

**Step-5. Prepare the output**

Application program calls TCP and sends the formatted data.

**Step-6. End the transaction**

Task control releases all storage associated with this task.

**Step-7. Write the output**

TCP sends output from TIOA to terminal as soon as communication line to terminal is free.

Releases TIOA.

Begins polling terminal for new transaction

## Summary

- In this lesson, you have learnt about:
  - Management modules and their types
  - Tables and their Types
  - Control blocks



## Review Questions

- Question 1: \_\_\_ module transfers data between the application program and the terminal.
  
- Question 2: A task consists of only one application program.
  - True/False
  
- Question 3: \_\_\_ module allows to implement time dependent applications.



# **Customer Information Control System**

Lesson 3: Data  
Communication Operation in  
CICS

## Lesson Objectives

- Characteristics of 3270 display
- Maps and Mapsets
- Symbolic Map and Physical Map
- Map definition Macros



3.1: Methods for Data Communication

## Types of Methods

- Three methods are available to the CICS application programmer for communicating with the terminals and logical units in the subsystems of the network that form part of the CICS system.
- The methods dealt with are:
  - Basic mapping support (BMS)
  - Terminal Control
  - Batch Data Interchange (BDI)

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

### Methods for Data Communication:

**Basic Mapping Support (BMS)** provides commands and options that can be used to format data in a standard manner. BMS converts data streams provided by the application program to conform to the requirements of the devices. Conversely, data received from a device is converted by BMS to a standard form.

**Terminal Control** is the basic method for communicating with devices; whereas both **BMS** and **Batch Data Interchange (BDI)** extend the facilities of terminal control to further simplify the handling of data streams. Both BMS and BDI use terminal control facilities when invoked by an application program.

**Batch Data Interchange (BDI)** provides commands and options that may be used possibly in conjunction with BMS commands, to communicate with batch logical units.

3.2: Characteristics of 3270 Display

## Functional Characteristics

- Before using BMS to define screen layout, the functional characteristics of 3270 display station's screen need to be understood.
- They are:
  - Field-oriented display
  - The screen consists of a number of user-defined fields
  - A special character called an attribute byte marks the beginning of a field

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

### Characteristics of 3270 Display:

The **3270 screen** is logically divided into a number of **user defined fields**.

A **screen field** is a specified area that contains a particular category of information.

Some **screen fields** allow the user to key in data into them while others are protected from data entry.

The location and characteristics of **screen fields** are determined by special characters called **attribute bytes**.

A **screen field** starts at the position immediately following its **attribute byte** and ends at the position immediately before the next field's **attribute byte**. Thus the length of the field depends on the position of the attribute byte. If there is no subsequent attribute byte, then the field continues to the end of the screen.

3.3: Attribute Byte Format

## Concept of Attribute Byte Format

- The attribute byte controls the following attributes:
  - Protection
  - Intensity
  - Modified Data tag
- It is always the first character of a field.
- It occupies a character position on the screen but appears as a blank.

Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

3.3: Attribute Byte Format

## Concept of Attribute Byte Format

Positions	Functions	Bit Settings
0-1	Information about bits 2 thru 7	
2-3	protection	00 = Unprotected alphanumeric 01 = Unprotected numeric 10 = Protected 11 = skip
4-5	Intensity	00 = Normal 01 = Normal 10 = Bright 11 = No-display
6	Must be 0	
7	MDT	0 = Field has not been modified 1 = Field has been modified

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

3.3: Attribute Byte Format

## Concept of Attribute Byte Format

- Field Protection
  - Unprotected
  - Protected
  - Auto-skip
- Field Intensity
  - Normal
  - Bright
  - No-display
- Shift

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

### Attribute Byte Format:

#### **Field Protection:**

**Unprotected:** It can enter any keyboard character into unprotected field.

Data entry fields are generally called as unprotected fields.

**Protected:** Data cannot be entered in a protected field. If the operator attempts to enter data, the keyboard is locked. Display only fields are called as protected fields.

**Autoskip:** An autoskip field is a protected field that automatically skips the cursor to the next unprotected field. They are usually used to mark the end of data entry fields.

The difference between the **protected field** and a **skip field** is that when the cursor is moved to a **skip field**, it automatically advances to the first position of the next **unprotected field** on the screen.

When the cursor moves to a **protected field** without auto-skip, it stops even though the user cannot enter data there. As a result of this, protected fields without auto-skip are called **stop fields**. If a **stop field** is used to mark the end of an **unprotected field**, the user has to press the tab key to advance to the next data entry field.

An **attribute byte** will be required to mark the end of an **unprotected field**. If the end of an unprotected field is not marked with either a **skip** or **stop** field, the screen will be unprotected beyond the intended data entry field until the beginning attribute byte of the next field.

**Attribute Byte Format:****Field Intensity:**

**Normal:** A normal intensity field displays the data at the normal operating intensity.

**Bright:** A bright intensity field displays the data at a brighter than normal intensity. This is often used to highlight keywords, errors, or operator messages.

**Nodisplay:** A nodisplay field does not display the data on the screen for operator viewing. This might be used to enter security data.

**Shift:**

It indicates whether the keyboard is numeric shift or alphanumeric shift.

In the older 3270 models, the numerals are located on the same keys as some of the letters, so the operator has to use SHIFT key on the keyboard to enter a numeric data. However, if a field's attribute byte indicates numeric shift, then the keyboard is automatically put into numeric shift so the operator does not have to press the NUMERIC SHIFT key.

3.4: Basic Mapping Support

## Concept of Basic Mapping Support

- **Basic Mapping Support (BMS):**
  - It is an interface between TCP and application program.
  - It provides commands and options that can be used to format data in a standard manner.
  - It converts data streams provided by the application program to conform to the requirements of devices.
  - It facilitates development of device and format independent programs.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 9

### Basic Mapping Support (BMS):

BMS removes the responsibility of formatting the screen from the application program. BMS places the data with necessary control characters in a TIOA\* (Terminal Input Output Area). It allows repositioning of fields without modifying application program

BMS makes the application program **Format Independent** and **Device Independent**.

It is a main storage area, used for terminal input/output operations.

3.5: Format Independence

## Concept of Format Independence

- Format Independence implies that application programs are written without concern for physical position of data fields within a display.
- In an application program, symbolic labels are used to refer the fields.
- BMS relates the label to the actual position in the display.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 10

3.6: Device Independence

## Concept of Device Independence

- Device Independence implies that application programs can be developed without any concern for physical characteristics of the terminals.
- BMS prepares the message based on the terminal type being used.



Copyright © Capgemini 2015. All Rights Reserved 11

3.6: Maps and Mapsets

## Concept of Maps and Mapsets

- A screen defined through BMS is called a “map”.
- Two ways of defining maps are as follows:
  - CICS supplied macros
  - Screen Definition Facility (SDF)

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 12

**Note:**

**Map** represents BMS coding for a screen, and **Mapset** represents a load module.

3.6: Maps and Mapsets

## Concept of Maps and Mapsets

- A “mapset” contains a single or multiple maps defined for one terminal type.
- A mapset once created must be assembled.
- A job to assemble a mapset requires two steps:
  - The first transforms your mapset into a physical map.
  - The second creates symbolic map from your mapset.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 13

**Note:**

A mapset is a special kind of assembler program, to define the format of each of the screens that the programs display.

3.6: Maps and Mapsets

## Concept of Maps and Mapsets

- A group of maps, which are link edited together is called mapset.
  - Each mapset must be registered in PPT.
  - The mapset name consists of two parts as follows:
    - Generic Name: 1 to 7 characters
    - Suffix : 1 character
  - Application program uses only the generic name.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

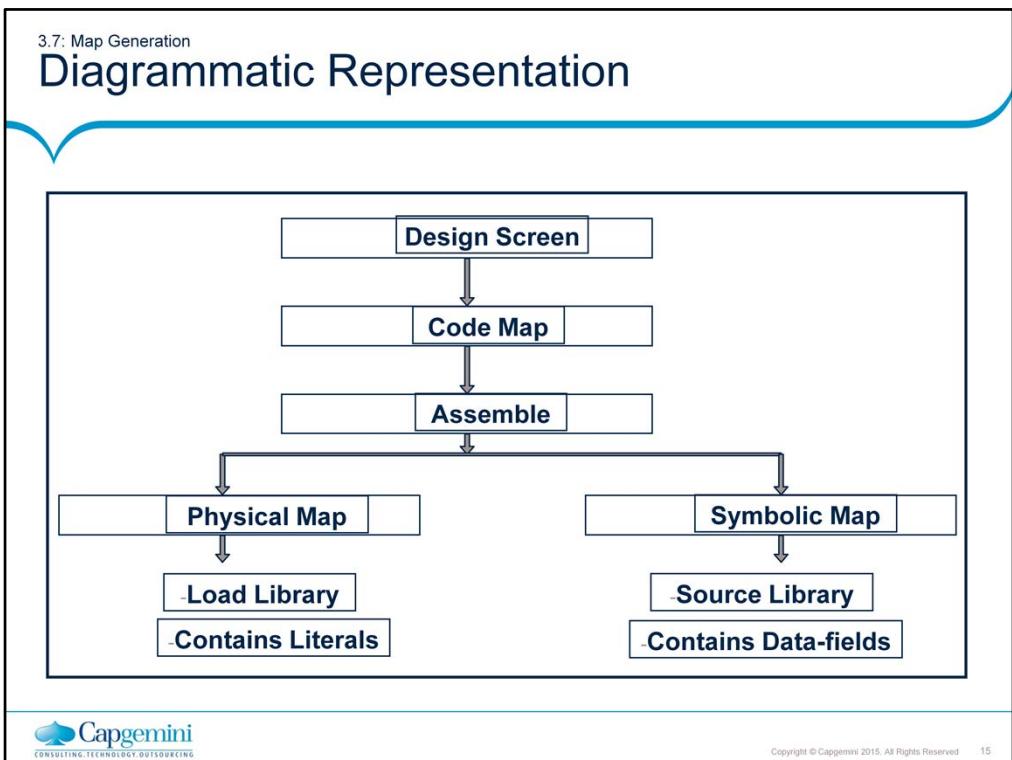
Copyright © Capgemini 2015. All Rights Reserved 14

**Note:**

The suffix is required at the time of mapset definition in order to distinguish the device types if the same mapset is used for different types of terminals.

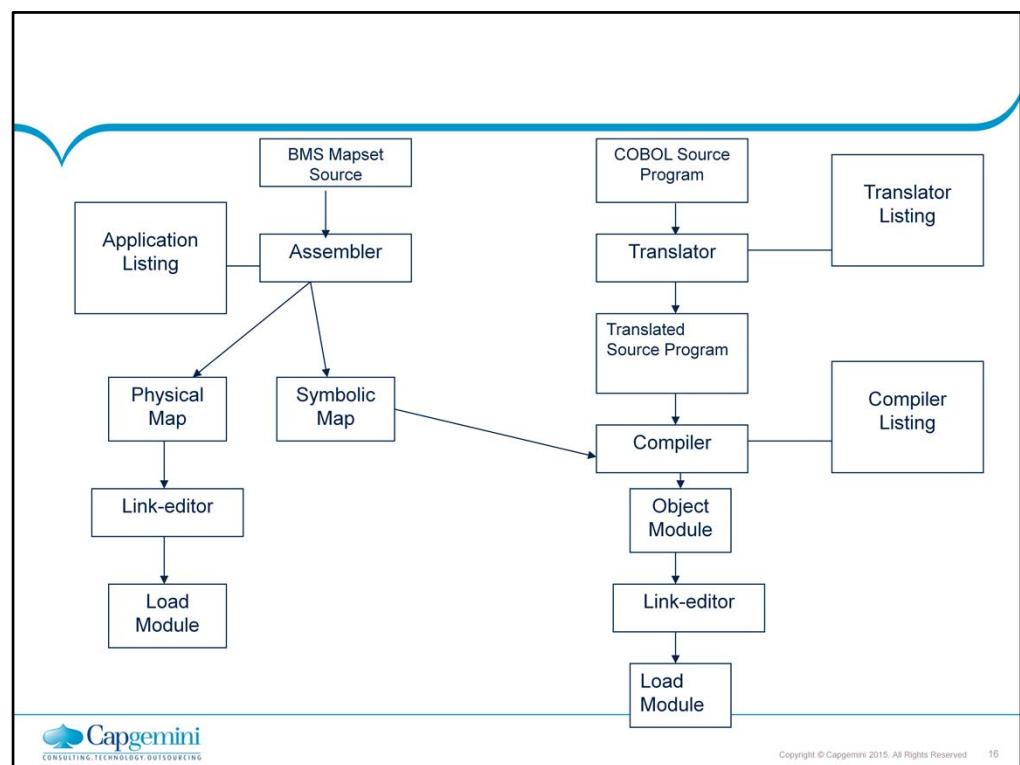
CICS automatically inserts the suffix depending on the terminal in use, thereby ensuring the device independence to the application program.

A mapset is a special kind of assembler program, to define the format of each of the screens that the programs display.



## Customer Information Control System

## Data Communication Operation in CICS



3.7: Physical and Symbolic Map

## Types of Maps

- A screen defined by CICS is called as a “map”.
- There are two types pf maps:
  - Physical Map, which is primarily used by CICS.
  - Symbolic Map, which is used by Application program.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 17

3.7: Physical Map

## Concept of Physical Map

- A physical map is a load module that contains a table which is used by BMS to determine the screen locations of the data transmitted to and received from the display station.
- Example:
  - A physical map might indicate that a particular field is displayed on the screen at column 16 of line 4.
  - A physical map also indicates attributes of each field such as protection and intensity.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 18

### Physical Map:

The primary objective of the physical map is to ensure the device independence in the application programs.

More concretely:

For **input operations**, the physical map defines maximal data length and starting position of each field to be read and allows BMS to interpret an input data stream.

For **output operations**, the physical map defines starting position, length, field characteristics (Attribute Bytes), and default data for each field, and allows BMS to add control characters and commands for output in order to construct an output data stream.

3.8: Physical Map

## Concept of Physical Map

- It describes display format for a given terminal type.
- It includes:
  - The format of input and output data
  - The fixed data such as page headings
  - Symbolic names of variable data-fields
  - The device type to which map relates
- Physical Map generation entails the following:

```
BMS macro coding      →      Assembly      →      Link-edit  
Load Module           LOADLIB          To be used by CICS
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 19

3.8: Symbolic Map

## Concept of Symbolic Map

- A symbolic map, in COBOL, is a copy library member that defines the format of data sent to or received from the terminal screen.
- When an application program requests that a map be sent to the terminal, BMS takes data from the symbolic map, formats (or maps) it according to the physical map, and sends it to the terminal.
- Likewise, when an application program requests that some data be retrieved from the terminal, BMS uses the physical map to map the data from the screen into the symbolic map.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 20

### Symbolic Map:

The primary objective of the BMS symbolic map is to ensure **device** and **format independence** to the application programs. Thus through the symbolic map, a layout change in the formatted screen can be done independent of the application program coding as long as the field name and length remain same.

It is used by the application program, which issues a COBOL COPY statement in order to include a symbolic map in the program.

3.8: Symbolic Map

## Concept of Symbolic Map

- It defines all variable data fields.
- It ensures device and format independence.
- It is used by application program which issues a COBOL COPY statement in order to include a symbolic map in the program.
- Symbolic Map generation entails the following:
  - BMS macro coding → Assembly → Symbolic map definition
  - COPYLIB → Copied into CICS application program

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 21

3.9: Format of Symbolic Map

## The Format

- A COBOL program must contain a COBOL COPY statement for each symbolic map definition in the working storage section.
- It starts with the 01 level definition of FILLER PIC X(12), which is the TIOA prefix created by TIOPFX=YES of the DFHMSD macro.
- Each symbolic map field consists of sub-fields. Each sub-field has a different suffix.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 22

### Format of Symbolic Map:

A **symbolic description map** is a source language data structure that the **assembler** or **compiler** uses to resolve source program references to fields in the map. Hence it must be copied into any application program that refers to fields in the map.

3.9: Format of Symbolic Map

## The Format

- nameL: It indicates length of the data entered in the field.
- nameF: It sets to ON when field gets modified.
- nameA: It indicates the attribute byte for input/output fields.
- nameI: It indicates the input data field.
- nameO: It indicates the output data field.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 23

### Format of Symbolic Map:

**nameL:** It indicates the halfword binary (PIC S9(4) COMP) field. For the input field, the actual number of characters typed in the field will be placed by BMS when the map is received. For the output field, this is used for the dynamic cursor positioning.

**nameF:** It is the Flag byte. For an input field, it will be X'80' if field has been modified but no data is sent (that is, the field is cleared). Otherwise, this field is X'00'.

**nameA:** It indicates the attribute byte for both input and output fields.

**nameI:** It indicates the input data field. X'00' will be placed if no data is entered.

**nameO:** It indicates the output data field.

3.10: Example of Symbolic Map Definition

## Illustration

```
01 SAMPMAPI.  
02 FILLER      PIC X(12).  
02 CHOICEL     PIC S9(4) COMP-4.  
02 CHOICEF     PIC X.  
02 FILLER REDEFINES CHOICEF.  
    03 CHOICEA    PIC X.  
02 CHOICEI     PIC X(001)  
02 MESSAGEL    PIC S9(4) COMP-4.  
02 MESSAGEF    PIC X.  
02 FILLER REDEFINES MESSAGEF.  
    03 MESSAGEA   PIC X.  
02 MESSAGEI    PIC X(030).
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 24

3.10: Example of Symbolic Map Definition

## Illustration

01 SAMPMAPO REDEFINES SAMPMAPI.  
02 FILLER                    PIC X(12).  
02 FILLER                    PIC XX.  
02 FILLER                    PIC X.  
02 CHOICEO                 PIC X(001).  
02 FILLER                    PIC XX.  
02 FILLER                    PIC X.  
02 MESSAGEO                PIC X(030).

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 25

3.11: Example of Customized Symbolic Map Definition

## Illustration

```
SAMPMAPI.  
01 SAMPLE-MAP.  
    02 FILLER      PIC X(12).  
    02 CHOICEL     PIC S9(4) COMP-4.  
    02 CHOICED     PIC X.  
    02 CHOICEA     PIC X.  
    02 MESSAGEL    PIC S9(4) COMP-4.  
    02 MESSAGED    PIC X(30).  
    02 MESSAGEA    PIC X.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 26

3.12: Components of a Screen

## The Elements

- CICS identifies screen into three components:
  - Maps
  - Mapset
    - Physical Map
    - Symbolic Map
  - Fields
    - Unnamed (Literals)
    - Named (Data fields)
    - Stopper

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 27

### Components of a Screen:

The first step in designing screen layout is to divide the screen into functional areas such as **title area**, **application data area**, and **message area**.

The title area of a screen should identify the program that displayed data. The application data area comprises the main portion of the screen. Three kinds of fields are usually found in this area:

**Keyword/unnamed/literal:** They contain constant data sent by program to identify the contents of the data field.

**Data field/named:** They contain data that the application program retrieves from files and displays. The data may appear exactly as stored in a file, or it may be edited, or it may be left blank for operator to enter data.

**Stopper field:** It is on data entry screen and restricts the length of the data field. Stopper field containing no data is used to stop operator from entering too many characters in a field.

3.12: Coding the Map

## Coding the Map using BMS

- MAPSET:
  - A group of maps is called Mapset.
  - Mapset name must be defined in the PPT. This is because CICS considers BMS mapset as a program coded in assembler.

MAP1	MAP2	MAP3
MAP4	MAP5	MAP6
MAP7	MAP8	MAPn

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 28

3.13: Modified Data Tag (MDT)

## Concept of MDT

➤ MDT is the one bit of the attribute character.

- If it is zero, it indicates that this field has not been modified by the terminal operator.
- If it is one, it indicates that this field has been modified by the operator.
- Only when it is ON, the data of the field is sent by the terminal to the application program.

➤ Effective use of MDT reduces data traffic in communication line, thereby significantly improving performance.



Copyright © Capgemini 2015. All Rights Reserved 29

3.13: Modified Data Tag (MDT)

## Concept of MDT

- Following are the three ways to set on the MDT :
  - By Data-entry
  - By setting attribute byte in the physical map
  - By moving MDT attribute in the application program

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 30

### Modified Data Tag:

When the terminal user modifies a field on the screen, MDT will be automatically set to "1" (on) by the terminal hardware.

If CNTL=FRSET is specified in the DFHMSD or DFHMIDI macro, when the mapset or map is sent to the terminal, MDT will be reset to "0"(off: that is, not modified) for all fields of the mapset or map respectively.

If FSET is specified in the ATTRB parameter of the DFHMDF macro for a field, when the map is sent to the terminal, MDT will be set to "1" (on: that is, modified) for the field regardless of whether the field has been modified by the terminal user.

3.14: Map Definition Macros

## Concept of Map Definition Macros

- To code a BMS map, BMS provides assembler macros as follows:
  - DFHMSD : To define a mapset
  - DFHMDI : To define a map
  - DFHMDF: To define a field

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 31

3.15: Format of BMS Macros

## The Format

- The BMS map definition macros are purely Assembler macros.
- Therefore the following coding must be maintained:

Col1 ↓ Symbolic  Name	Col 10 ↓ Opcode  Additional parameters	Col16 ↓ Parameter(s)	Col17 ↓ Cont'd	Col72 ↓ Char
-----------------------------------	--	----------------------------	----------------------	--------------------

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 32

### Format of BMS Macros:

**Label:** The label specifies the symbolic name for the statement and begins in column 1. For a BMS macro, the label must begin with a letter and can be 7 characters long.

**Op-code:** The op-code specifies the instructions to be executed (in a mapset usually one of the BMS macros). It begins in column 10. For a BMS macro instruction, the op-code is DFHMSD, DFHMDI, or DFHMDF. The op-codes for two assembler commands used in a mapset are **PRINT** and **END**.

**Parameters:** The parameters (sometimes called as **operands**) provide information that the instruction requires to work properly. They are separated from one another by commas with no intervening spaces. The first parameter should follow the op-code after one space. Although parameters usually begin in column 16, the first parameter following a BMS op-code begins in column 17 because the BMS macro op-codes are all six characters long and start in column 10.

**Example:** LENGTH= 16

3.16: Example of BMS Macros

## Illustration

■ Example:

1	10	16	17	72
MENU	DFHMSD	TYPE=&SYSPARM	MODE=INOUT, LANG=COBOL.....	X

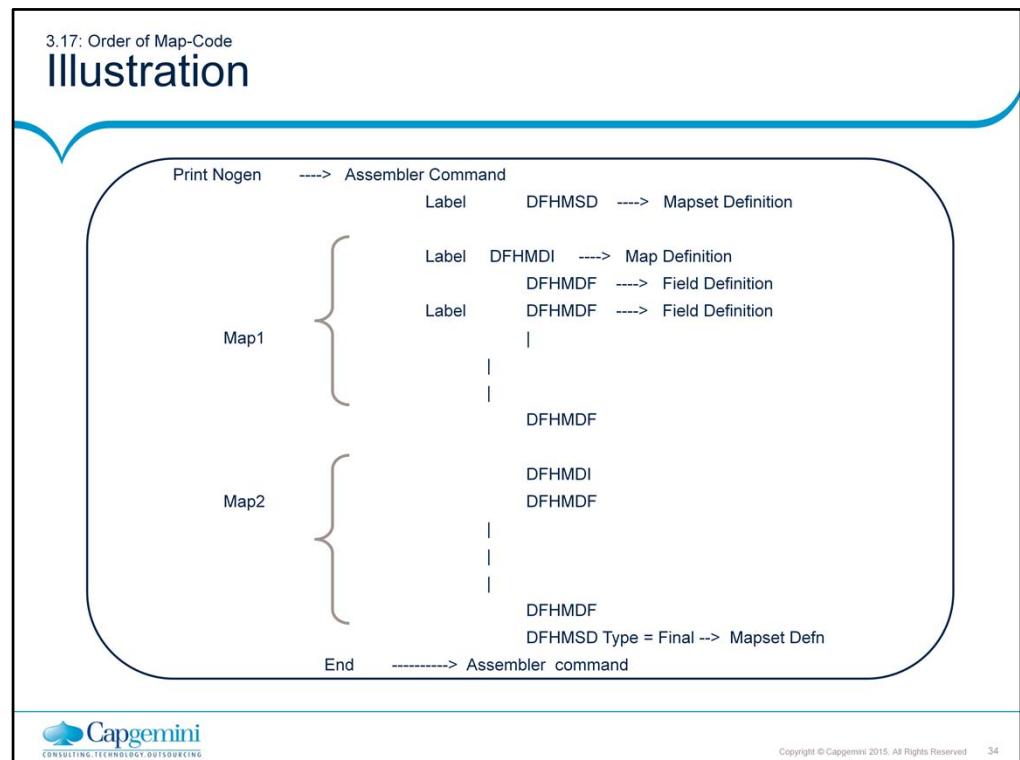
 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 33

### Example of BMS Macros:

**Continuation lines:** To continue a statement on the next line, code a comma after a parameter and place any non-blank character in column 72(for example: X). The next parameter begins in column 16 of the following line. The following line is called as a continuation line. It is recommended to code BMS macros with only parameter per line to make mapsets easier to read.

**Comment lines:** Readability of a mapset can also be improved by using comment lines to separate groups of related macros from one another. A comment line is any line with \* in column 1.

**Order of Map-Code:**

- There is rule for the order of BMS macros, which must be followed. That is, within one **mapset definition**, the **map definition** can be specified as many times as you wish. Within one **map definition**, the **field definition** can be specified as many times as you wish.

3.18: Print NOGEN

## Use of Print NOGEN

- The PRINT NOGEN, causes the assembler not to print statements generated by the BMS macro instructions. If you do not include a PRINT NOGEN command, your assembler listing will contain hundreds of lines that are not important.
- Hence it is recommended to start the mapsets with PRINT NOGEN command.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 35

3.19: END

## Use of END

- The last line in the mapset, that is END, is required.
- END, like PRINT, is an assembler command.
- Logically, it tells the assembler that there are no more source statements in the mapset program.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 36

3.20: DFHMSD Macro

## Concept of DFHMSD Macro

- Function:
  - The DFHMSD macro is used to define a mapset and its characteristics or to end a mapset definition.
  - Only one mapset definition is allowed within one assembly run.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 37

3.21: Format of DFHMSD Macro

## The Format

Label DFHMSD      TYPE=DESECT|MAP|&SYSPARM|FINAL,  
[MODE=IN|OUT|INOUT,]  
[LANG=ASM|COBOL|PLI,]  
[STORAGE=AUTO|BASE=name,]  
[CTRL=(FREEKB, ALARM, FRSET).]

[HIGHLIGHT=OFF|BLINK|REVERSE|UNDERLINE,]  
[VALIDN=MUSTFILL|MUSTENTER,]  
[TERM=TYPE|SUFFIX=n,]  
[TIOAPFX=YES|NO,]  
[DATA=FIELD|BLOCK]

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 38

3.21: Format of DFHMSD Macro

## The Format

- End of Mapset Definition:

DFHMSD      TYPE=FINAL

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 39

3.21: Format of DFHMSD Macro

## The Format

- The mapset name will appear in the PPT and in the copy statement for the symbolic map in the COBOL application program.
- TYPE: The type parameter indicates whether a Physical Map(Type=MAP) or a symbolic Map (Type=DSECT) is being generated. Type=&SYSPARM indicates that both will be generated.
- LANG: It specifies the language to be used for the symbolic map.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 40

3.21: Format of DFHMSD Macro

## The Format

- MODE: It specifies whether the map is being used for input (IN) or output (OUT) or both.
- TERM: The term parameter specifies the type of terminal that the map can be used with:
  - If you code ALL or 3270, then the map can be used with any type of terminal (even non-3270 devices).
  - Although it is a bit less flexible, it is more efficient to specify the exact terminal type.
- CTRL parameter:
  - FREEKB: To unlock the keyboard
  - FRSET: To reset MDT to zero
  - ALARM: To set an alarm at screen displaytime
  - PRINT: To indicate the mapset to be sent to the printer

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 41

### Format of DFHMSD Macro:

**Term:** The value that is specified for the term parameter affects the name CICS assigns for physical map. CICS adds one character suffix to the end of the mapset name for the physical map. The exact character used depends on the terminal selected.

**Example:** For a mapset definition names MORSET1 the symbolic map definition may be MORSET1 and a physical map definition name may be MORSET1M (since M is the suffix for 3270 model 2 terminals).

Due to the suffix, it is possible to have several physical mapsets for a single symbolic map. To use mapset MORSET1 on 3270 model1 terminals, all we need to do is change the **term parameter** to 3270-1 and reassemble the map. BMS will then generate a physical mapset named MORSETL. (L being the suffix for 3270 model1 terminals).

When the mapset is referred in the application program, the seven-character name without the suffix is used. CICS automatically retrieves the physical map that is appropriate for the terminal that the program is running on, by appending the correct suffix.

**Format of DFHMSD Macro:****Example:**

If the program is run on 3270 model1 terminal, then CICS retrieves the mapset named MORSET1L.

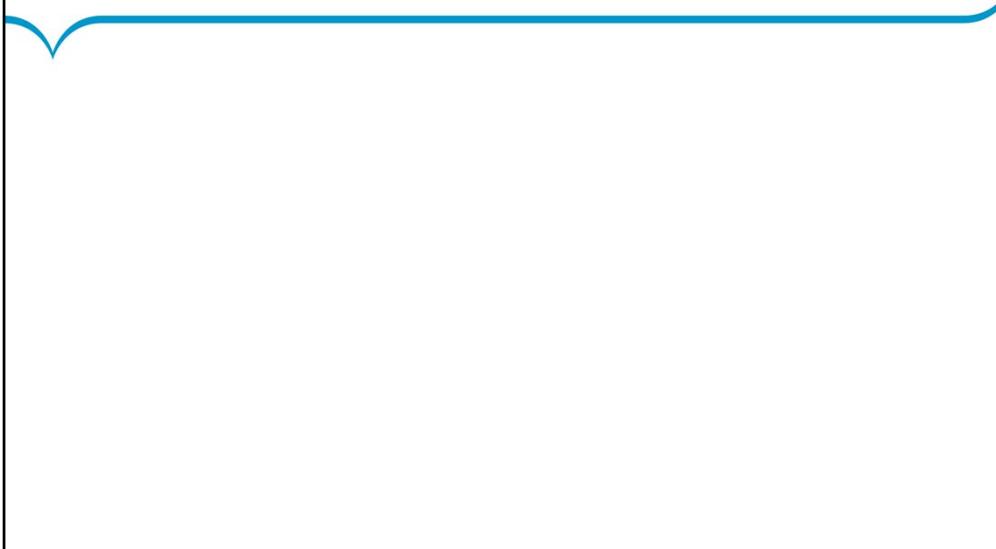
If the program is run on 3270 model2 terminal, then CICS retrieves the mapset named MORSET1M.

If CICS cannot locate the appropriate mapset, then it looks for a mapset with no suffix. An un-suffixed mapset is generated with we code TERM=ALL or TERM=3270. Such a maspset can be used for any type of terminal.

To accommodate any terminal type, however considerable run-time overhead is added to the mapset. Hence it is better to specify terminal type in the mapset.

3.21: Format of DFHMSD Macro

## The Format



 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 43

### Format of DFHMSD Macro:

- **CTRL:** The CTRL parameter specifies the control options used by the maps that a mapset defines. Alternatively, CTRL options can be specified individually for each map on the DFHMDI macro. Since the same control options apply to all the maps in a mapset, the CTRL parameter can be coded on the DFHMSD macro.
- **FREEKB:** The most common CTRL option is FREEKB. If this is specified, then the keyboard is unlocked whenever a map is sent to the terminal. If FREEKB is not specified, then the keyboard is locked and the operator must press the REST key to enter data.
- **ALARM:** The ALARM option causes the audio alarm to sound whenever the map is sent to the terminal. In some installations, the audio alarm is used to warn the operator of an error condition.
- **FRSET:** Suppose CTRL=FRSET is specified in the DFHMSD or DFHMDI macro for a field. Then whenever the map is sent to the terminal, MDT will be reset to "0" (off, that is, not modified) for all the fields of the mapset or map respectively.

3.21: Format of DFHMSD Macro

## The Format

- **STORAGE:** This parameter is used when more than one map set is defined in the mapset. If storage = AUTO is coded, then the symbolic maps for maps in the mapset will occupy separate storage locations.
- **TIOAPFX:** If YES, it generates a 12-byte filler item at the beginning of the symbolic map and should always be specified for COBOL maps.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 44

### Format of DFHMSD Macro:

- **Storage:** It indicates how storage will be allocated to the symbolic map. If storage=AUTO is coded, then each symbolic map will have its own storage area. If this clause is omitted, then the symbolic maps will occupy the same storage locations.
- **TIOAPFX:** If the map is to be processed, then a command level COBOL program, TIOAPFX = YES must be specified on DFHMSD macro. The TIOAPFX parameter generates a 12 byte filler item at the beginning of the symbolic map. The system uses those 12 bytes to maintain control information.

3.22: DFHMDI Macro

## Concept of DFHMDI Macro

- Function:
  - The DFHMDI macro is used to define a map and its characteristics in a mapset.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 45

3.23: Format of DFHMDI Macro

## The Format

Label      DFHMDI [SIZE=(LINE , COLUMN)]  
[ , LINE=number | NEXT | SAME]  
[ , LANG=ASM | COBOL | PLI ]  
[ , STORAGE=AUTO | BASE=name]  
[ , CTRL=(FREEKB] [, ALARM] [, FRSET])]  
[ , VALIDN=MUSTFILL | MUSTENTER]  
[ , TERM=TYPE | , SUFFIX=n]  
[ , TIOAPFX=YES | NO]

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 46

3.23: Format of DFHMDI Macro

## The Format

- The DFHMDI macro is used to define a map and its characteristics in a mapset.
  - The LABEL must be specified as the symbolic name to the DFHMDI macro.
  - SIZE(lI, cc) is used to define the size of the map by the line size (lI) and column size (cc).
  - LINE and COLUMN indicates the starting position of the map in line and column numbers, respectively.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 47

### Format of DFHMDI Macro:

The LABEL on the DFHMDI macro is the name of the map. This name will be used for the map in the CICS program. Mapset name must be unique within a CICS system. Similarly, map name must be unique within a mapset.

The SIZE parameter specifies the number of lines and columns in the map. The map can be smaller than the screen but usually the two are of the same size. Hence SIZE=(24,80) is usually coded for standard 24\*80 screen.

Generally most maps start at first column of first line on the screen. Hence LINE=1 and COLUMN =1 is usually coded.

3.24: DFHMDF Macro (Function)

## Concept of DFHMDF Macro

- The DFHMDF macro is used to define a field in a map and its characteristics.
- The DFHMDF macro can be issued as many times as you wish within one DFHMDI macro.



Copyright © Capgemini 2015. All Rights Reserved 48

3.24: DFHMDF Macro (Function)

## Format of DFHMDF Macro

```
LABEL      POS=(line | column)
           [, LENGTH=number]
           [, JUSTIFY=(LEFT | RIGHT)]
           [, ATTRB=(ASKIP | PROT | UPROT [, NUM])
           [, BRT | NORM | DRK]
           [, IC][, FSET]) ]
           [, HIGHLIGHT=OFF | BLINK | REVERSE |
UNDERLINE]           [, VALIDN=(MUSTFILL | MUSTENTER)]
           [, OCCURS=number]
           [, INITIAL='value']
           [, PICIN='value'][, PICOUT='value']
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 49

3.24: DFHMDF Macro (Function)

## Format of DFHMDF Macro

- If the field name is required, then the name of the field (1 to 7 chars) must be specified as the symbolic name to the DFHMDF macro.
- POS=(line, column) indicates the starting position of the field in the line and column number including the attribute character.\
- The actual data field follows the attribute byte, so if you want a field to start in column 5 of line 10, you code POS=(10,4).



Copyright © Capgemini 2015. All Rights Reserved 50

3.24: DFHMDF Macro (Function)

## Format of DFHMDF Macro

- LENGTH indicates the length of the field EXCLUDING the attribute character.
  - Hence if you specify LENGTH=5, you actually define six screen positions – five positions for the field itself and one position for the attribute byte.
- INITIAL defines the initial value of the field (if any).
- PICIN and PICOUT defines the PICTURE clauses of the symbolic map in COBOL, which is useful for numeric field editing.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 51

### Format of DFHMDF Macro:

DFHMDF POS=(5,13),  
LENGTH=14

To calculate the next available position, add 14 plus 1 to the starting column 13. In this example, the next available position is column 28 of line 5.

3.24: DFHMDF Macro (Function)

## Format of DFHMDF Macro

- ATTRB specifies the characteristics of the attribute byte.
  - Example: ATTRB=(ASKIP,BRT)
- The NUM option of the ATTRB parameter specifies a numeric protected field.
- FSET causes the MDT to be turned on before the map is send to a terminal.
  - Usually it is not specified, since the MDT bit is set automatically when an operator enters data in an unprotected field.
  - However, if a protected field is to be transmitted to the program, then FSET option has to be specified.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 52

### Format of DFHMDF Macro:

- **ATTRB:** The first set of options for the ATTRB parameter controls the intensity attribute. The following can be coded:
  - **NORM:** The field displays at Normal intensity.
  - **BRT:** The field displays at Bright intensity.
  - **DRK:** Field will not be displayed.
- If no intensity is specified, then NORM is assumed.
- The second set of options for the ATTRB parameter specifies a fields protection attribute. The following can be coded:
  - **PROT:** The field is protected. The user cannot enter data into the field. It is the normal attribute for caption and display only fields.
  - **ASKIP:** It has the same effect as PROT, except that the cursor automatically jumps over the field to the next unprotected field.
  - **UNPROT:** Field is unprotected. Data can be entered in the field.

## Summary

- Characteristics of 3270 display
- Basic Mapping support
- Mapsets and Maps
- Physical Map and Symbolic Map
- BMS Macros



## Review Questions

- Question 1: The DFHMDI macro can be issued as many times as you wish within one DFHMDF macro.
  - True/False
  
- Question 12: The map name must be registered in the PPT.
  - True/False
  
- Question 3: The \_\_\_ map indicates attributes of each field.



# **Customer Information Control System**

Lesson 4: Characteristics of  
CICS

## Lesson Objectives

- Characteristics of CICS
- Conversational programming
- Pseudo-Conversational programming



4.1: Multithreading

## Concept of Multithreading

- Singlethreading:
  - It is the execution of a program from beginning to completion.
- Multithreading:
  - It is the system environment where the tasks are sharing the same program under the multitasking environment.
  - It is a subset of multitasking, since it concerns tasks which use the same program.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 3

### Multithreading:

Under the multithreading environment, a program is shared by several tasks concurrently. For each task, the program must work as if it were executing instructions exclusively for each task. Therefore, it requires special considerations such as reentrancy.

Contrary to the multithreading environment, under the single threading environment, a program is used by only one job (or task) at a time. Processing of one transaction is completed before another transaction is started. A typical example is a batch job.

Although multithreading is not a unique concept to CICS, CICS manages multithreading of CICS tasks within its own region. That is, CICS provides the multithreading environment where more than one CICS task, which uses the same program, run concurrently.

4.2: Quasi-Reentrancy

## Concept of Quasi-Reentrancy

- In order to make multithreading possible, an application program must be “reentrant”.
- A completely reentrant program does not change itself in any way. That is to say a reentrant program cannot modify data in working storage.
- Hence, any user can enter a reentrant program at any point without affecting other users who are also running it.
- Command level CICS allows you to use working storage by treating all programs as quasi-reentrant.



Copyright © Capgemini 2015. All Rights Reserved 4

### REENTRANT PROGRAM

For Multithreading to work, a program must be reentrant

A Reentrant program does not change itself in any way, including the data in the working storage

Any user can enter a reentrant program at any point without affecting other users who are also running it

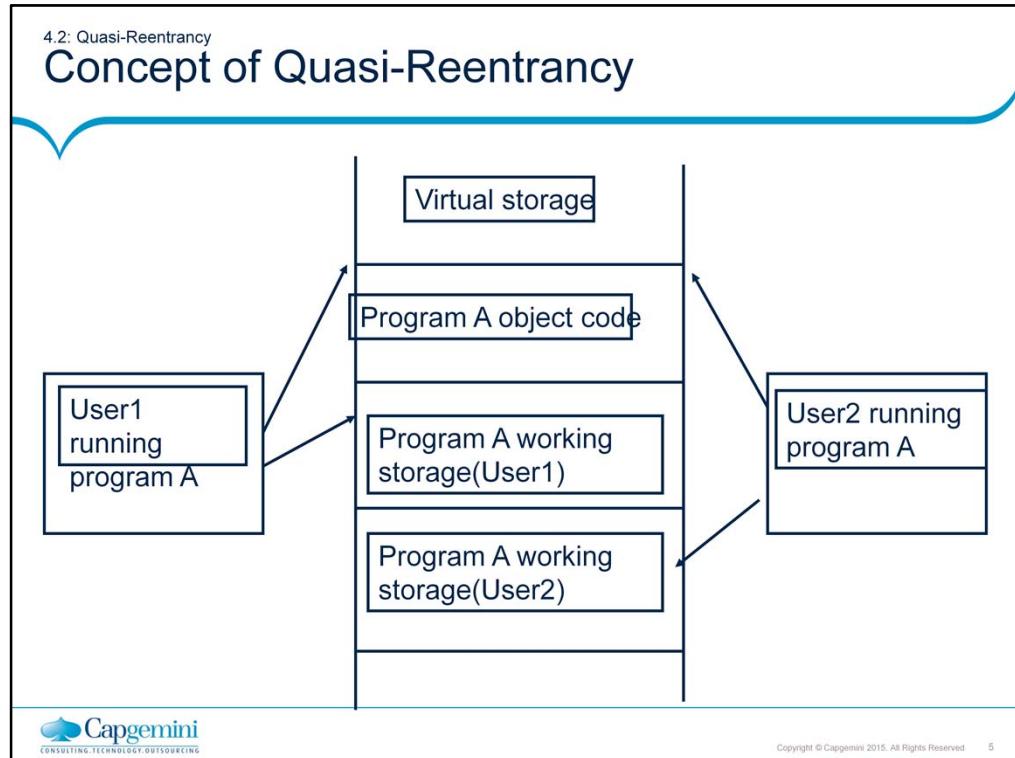
### QUASI REENTRANT PROGRAM

CICS makes program Quasi Reentrant

Multiple users share the same storage for the programs object code

Separate working storage area for each user

Also any insertion or data altered in them must be restarted



### Quasi-Reentrancy:

In the example on the above slide, the two users share the same application program, that is PROGRAM A. They share the same storage for the program's object code – that is the procedure division – but each is given a separate working storage area. That way each use the use the working-storage in the normal fashion. A Quasi-reentrant program is a reentrant program under CICS environment. That is a Quasi-reentrant program is a CICS program which does not modify itself. When you write a command level CICS program, you do not have to worry about making it quasi-reentrant. CICS handles that automatically for you.

4.2: Conversational Programming  
**Terminal Conversion**

- Online functions need to interact with the terminal users.
- Conversation : The process of sending a message to the terminal user and receiving his response.
- 3 different terminal conversation modes :
  - Non Conversational
  - Conversational Mode
  - Pseudo Conversational
- Pseudo-conversational approach recommended.



Copyright © Capgemini 2015. All Rights Reserved 6

4.2: Conversational Programming

## Non Conversation Mode

- In non conversational mode, there is no conversation which takes place between the terminal user and the program.
- They are usually single task programs like enquiry or printing reports on the terminal.

```

graph TD
    START([START]) --> READ[READ FROM FILE]
    READ --> GENERATE[GENERATE REPORT]
    GENERATE --> END([END])
  
```



Copyright © Capgemini 2015. All Rights Reserved 7

### EXAMPLE OF A NON-CONVERSATIONAL TRANSACTION

User must enter all data required in the proper format with TRANSACTION-ID  
(4 Characters in CICS)

E.G. : MBAL 5555

RECEIVE ACCOUNT NO.

READ DESIRED RECORD

IF RECORD FOUND

SEND MEMBERS CURRENT BAL TO  
TERMINAL

ELSE

SEND 'INCORRECT ACCOUNT NO.' TO  
TERMINAL

ENDIF

RETURN TO CICS

4.2: Conversational Programming

## Conversational Mode

- A Conversational Program :
  - Sends a message to the terminal
  - Waits for the user to respond
  - Receives the message
  - Acts on the received message
- This process is repeated till the user decides to quit.
- Most of the client server programs are made like this.

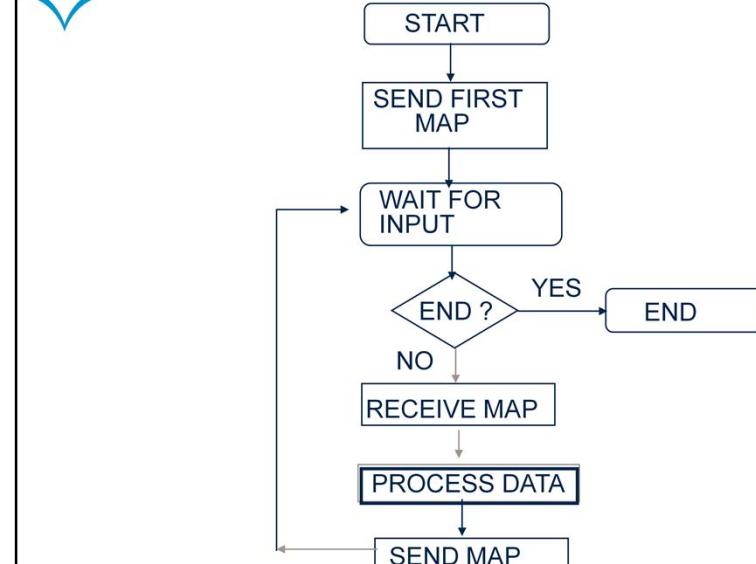
 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

### Conversational Programming:

Example: In a 30 users system, it might take 30 seconds for an operator to enter a transaction and 1 second to process it. In such a case, only one of the 30 users is actually processing data at any given moment. The other 29 are waiting taking up space in main storage. Hence, in multi-user environment, conversational programs can be tremendously inefficient.

On a single-user system this is not a problem. However, with the multi-user system it is a problem.



Copyright © Capgemini 2015. All Rights Reserved 9

#### EXAMPLE OF A CONVERSATIONAL TRANSACTION

ENTER TRANS-ID ONLY.

CODE :

SEND 'DESIRED ACCOUNT NO.?' TO TERMINAL

RECEIVE ACCOUNT NO.

READ DESIRED RECORD

IF RECORD FOUND

SEND CURRENT BAL TO TERMINAL

ELSE

SEND 'INCORRECT ACCOUNT NO.' TO TERMINAL

ENDIF

RETURN TO CICS

4.4: Concept of Pseudo-Conversational Programming

## Pseudo-Conversational Programming

- In Pseudo-Conversational Programming:
  - Program is removed from the storage when it is waiting for data from a Terminal.
  - Program is restarted when the operator has filled in the required data.
  - Program remains in storage only when it is processing data.



Copyright © Capgemini 2015. All Rights Reserved 10

### Pseudo-Conversational Programming:

Since pseudo-conversational programs use main storage more efficiently than conversational programs, almost all CICS installations require CICS programs to be pseudoconversational.

4.5: How does a Pseudo-Conversational Program work?

## The Process

- Broadly the following steps occur during the working of a pseudo-conversational program:
  - An operator signals CICS to restart a pseudo-conversational program by pressing one of the terminal's AID keys.
  - A CICS program retrieves the data that the operator entered by issuing a CICS command: RECEIVE MAP.
  - The RECEIVE MAP command transfers data from the screen to the symbolic map in a program.
  - After the program retrieves the data that the operator entered, it processes it.



Copyright © Capgemini 2015. All Rights Reserved 11

Note: AID keys are the Enter key, the Clear key, the PF keys, and the PA keys.

4.5: How does a Pseudo-Conversational Program work?

## The Process

- Next, the program issues a CICS SEND MAP command to display the results.
- After the program issues a SEND MAP command, it issues a RETURN command.
- The RETURN command ends the program just as a STOP RUN does in a batch COBOL program.
- In addition, the RETURN command specifies what trans-id CICS should invoke the next time the operator presses an AID key.



Copyright © Capgemini 2015. All Rights Reserved 12

### Note:

Suppose the operator presses the CLEAR key to end the terminal session. Since the CLEAR key is an AID key, CICS restarts the program automatically. When program sees that the operator pressed the CLEAR key rather than the ENTER key, it sends the message to the terminal. Then it issues a RETURN command without a trans-id. That ends the pseudo-conversational cycle.

4.5: How does a Pseudo-Conversational Program work?

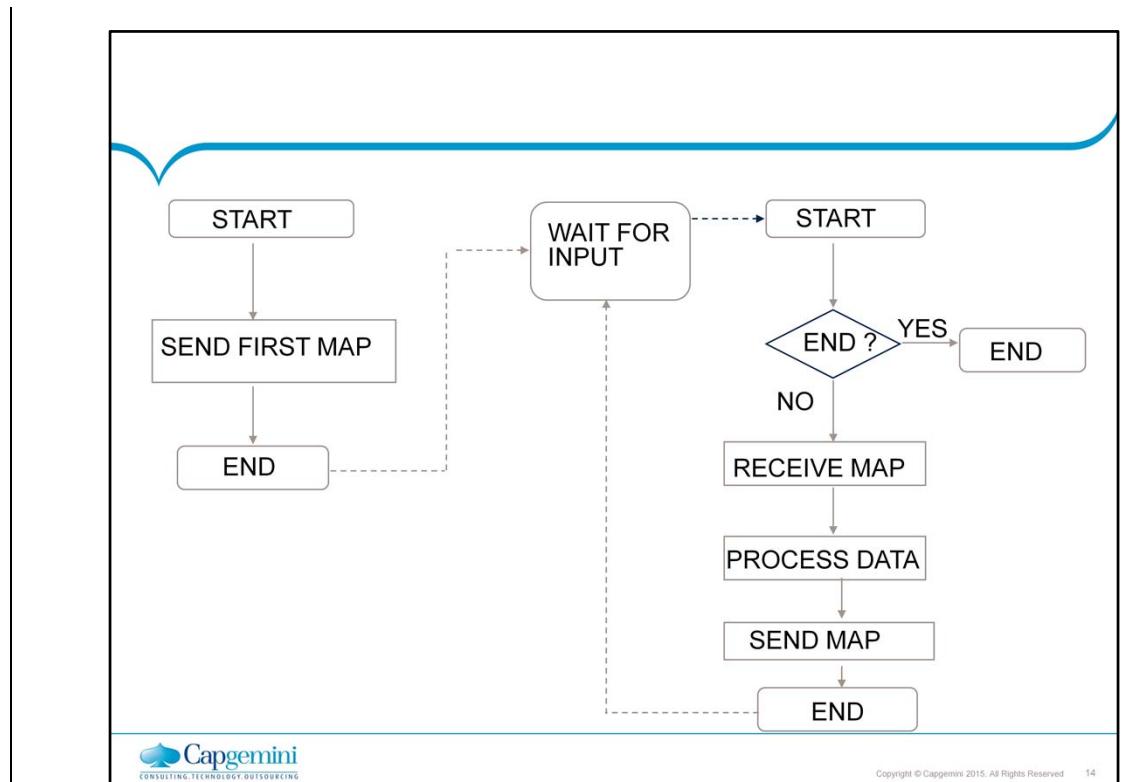
## The Process

- In the subsequent time execution:

- The map is received and the data is transferred from map variables to map variables.
- The data is validated.
- In case of an error, the appropriate error message is transferred to the error message field and the map is sent to the terminal.
- In case there is no error, the map is sent to the terminal with the appropriate.



Copyright © Capgemini 2015. All Rights Reserved 13



## EXAMPLE OF A PSEUDO-CONVERSATIONAL TRANSACTION

TRN1 :  
CASE

WHEN INITIATED FIRST TIME

SEND .....

RETURN TO CICS WITH TRN1

WHEN INITIATED SECOND TIME

RECEIVE .....

PROCESS .....

SEND

RETURN TO CICS WITH TRN1

WHEN INITIATED THIRD TIME

RECEIVE .....

PROCESS .....

SEND

RETURN TO CICS WITH TRN1

END CASE

4.6: Passing Data using COMMAREA

## Use of COMMAREA

- Communication area (COMMAREA) is used to preserve data from one program execution to the next.
- The data is received in its linkage section by the receiving program, under DFHCOMMAREA.
- The DFHCOMMAREA in the linkage section must have the same length as the communication entry in the Working-Storage section.



Copyright © Capgemini 2015. All Rights Reserved 15

### Passing Data using COMMAREA:

Each time a program is executed, a fresh copy of the working storage is executed. Hence the changes made to the contents of the working storage fields are not saved between executions of a pseudo-conversational programs. If the data needs to be preserved from one program execution to the next, it can be done using communication area. The communication area may be used in functions like RETURN/XCTL/LINK. The length of COMMAREA must be specified in the LENGTH parameter of the LINK or XCTL command in the calling program. The LENGTH parameter must be defined as a half-word binary fields (S9(04) COMP).

Even if an entry for DFHCOMMAREA is not coded, the command translator inserts a definition for one-byte communication area automatically. However, it is recommended to code your own definition of DFMCOMMAREA.

4.6: Passing Data using COMMAREA

## Use of COMMAREA

- When the program starts, EIBCALEN contains the length of the communication area passed to the program.
- If no data is passed, then EIBCALEN = 0.
- If data is passed, then EIBCALEN = Length of COMMAREA.



Copyright © Capgemini 2015. All Rights Reserved 16

### Passing Data using COMMAREA:

The command translator inserts another special field called EIB (Execute interface Block) in all CICS program after the DFHCOMMAREA entry.

The EIB contains several useful items, and one of the fields is EIBCALEN. The called program can find length of data passed using the EIB field EIBCALEN.S

4.6: Passing Data using COMMAREA

## Use of COMMAREA

- If no data is passed, then EIBCALEN is equal to zero.
- If data is passed, then EIBCALEN is equal to length of COMMAREA.

**Previous Execution**

IDENTIFICATION DIVISION.  
ENVIRONMENT DIVISION.  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 WS-COMM-AREA.  
---  
LINKAGE SECTION.  
01 DFHCOMMAREA.  
---  
---  
PROCEDURE DIVISION

**Current Execution**

IDENTIFICATION DIVISION.  
ENVIRONMENT DIVISION.  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
01 WS-COMM-AREA.  
---  
LINKAGE SECTION.  
01 DFHCOMMAREA.  
---  
---  
PROCEDURE DIVISION

Temp. Storage Area

**Capgemini**

Copyright © Capgemini 2015. All Rights Reserved 17

### Passing Data using COMMAREA:

The working-storage section definition of the communication area is the source of the data passed to the next execution of the program, while the Linkage section is the destination of the data passed from the previous execution. Thus two versions of the communication area exist during an execution of the program, one to pass data to the next execution, the other to receive data from the previous execution.

## Summary

- Characteristics of CICS
- Conversational programming
- Pseudo-Conversational programming



## Review Questions

- Question 1: It is not mandatory to code DFHCOMMAREA.
  - True/False
- Question 2: \_\_\_ is detected to evaluate first time execution of a program.
- Question 3: \_\_\_ is a subset of multitasking.



# **Customer Information Control System**

Lesson 5: Command Level  
Programming

## Lesson Objectives

- CICS command format
- Command Language Translator
- COBOL/CICS Program Structure



5.1: CICS Program Development

## The Concept

- Application programming in CICS can be done with many host languages that are supported by IBM, such as COBOL, PL/1, and Assembler.
- The coding can be done by using either of the following:
  - Macro level
  - CICS Command level library

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

### CICS Program Development:

**Macro Level Programming:** In the earlier versions of CICS, the CICS application programs were written based on CICS macros which were functionally primitive in comparison with CICS commands. Macro Level Programming involves calling macros in the host language using the program areas for accessing CICS resources. This style of programming is cryptic and complex as the programmer is required to remember all the macros and their parameters.

**Command Level Programming:** Command Level Programming is a high-level language, which provides set of CICS commands for accessing the resources. Programming in this style is comparatively easier. Command level Preprocessor translates the CICS commands into equivalent macro calls.

5.2: CICS Command Format

## The Format

- CICS commands are embedded into the Host language, for example, in the Procedure Division of COBOL program.
- CICS commands in a COBOL program are delimited by
  - EXEC CICS.....END-EXEC.
- EXEC CICS is coded in margin B of COBOL program.
- The command level translator replaces these statements by COBOL “MOVE” statements followed by COBOL “CALL” statement.
- The translated module is compiled and linked to produce an executable load module.
- The translator also includes copy books into program.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

5.2: CICS Command Format

## The Format

```
EXEC CICS Function
[option(argument)]
option(argument)
-----
-----
]
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

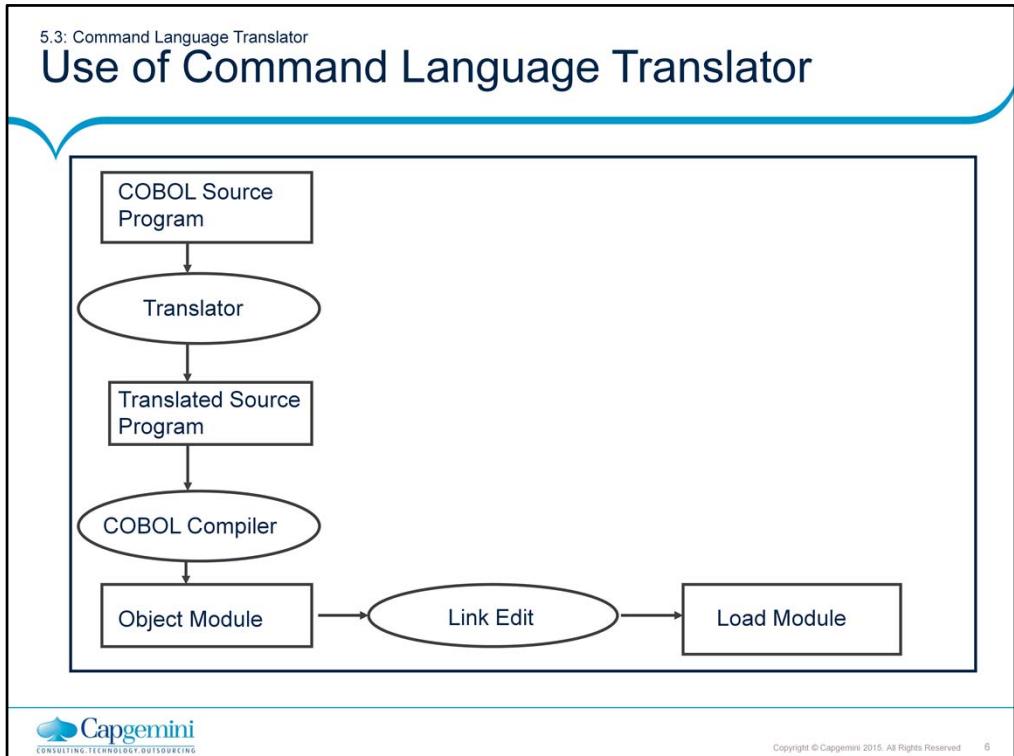
Copyright © Capgemini 2015. All Rights Reserved 5

**CICS Command Format:**

“Function” describes the operation required, for example, SEND/RECEIVE.

“Option” describes any of the many optional facilities available with each function.

Options can be coded in any order but preferably one option per line. Options may be followed by an argument in parentheses.  
“Argument” is a value, can be a data-name or literal.



### Command Language Translator:

Separate translators are available for Assembly, COBOL, and PL/I languages with embedded CICS.

The translator is executed in a separate job step.

The job step sequence is :

Translate --- Compile (or assemble) --- Link edit

Each translator is host language oriented and accepts the source program as input from SYSIN.

The translator writes the source listing and error messages to SYSPRINT.

The translated source program is accepted as input by the COBOL compiler and link edited to generate a load module.

The translator modifies the linkage section by inserting the EIB structure as the first parameter, and inserts declaration of the temporary variables that it requires into working storage section.

For COBOL application program, each command is replaced by one or more COBOL move statements followed by a COBOL CALL statement.

MOVE statement assign constants to COBOL data variables.

The constants are coded as arguments to options in the commands.

5.4: COBOL / CICS Program Structure

## The Syntax

IDENTIFICATION DIVISION.  
PROGRAM-ID.  
ENVIRONMENT DIVISION.  
DATA DIVISION.  
WORKING-STORAGE SECTION.  
Variable declaration  
File layout  
Symbolic map  
Copy books  
DFHEIVAR  
LINKAGE SECTION.  
DFHCOMMAREA  
EIBBLOCK  
BLL CELLS  
PROCEDURE DIVISION.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

5.4: COBOL / CICS Program Structure

## Prohibitions in CICS

- The following COBOL statements are prohibited in a CICS application program:
  - ACCEPT, CURRENT-DATE, DATE DAY, DISPLAY, EXHIBIT, STOP RUN, TRACE
  - Any I/O statements (OPEN, CLOSE, REPORT WRITER Feature)
  - SORT Feature

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

### COBOL / CICS Program Structure:

The CICS application program must end with the CICS RETURN command and / or GOBACK statement.

The CALL statement is allowed:

If the called program does not issue any CICS commands or inhibited COBOL statements mentioned above.

If it is written as a reentrant program.

The CALL statement, in this case, can be issued as shown in the following example:

```
CALL subprog USING xxx yyy zzz
```

## Summary

- CICS command format
- Command Language Translator
- COBOL/CICS Program Structure



## Review Questions

- Question 1: \_\_\_ statement is prohibited in a CICS program.
  - Option 1: Accept
  - Option 2: Read
  - Option 3: Write
  - Option 4: FD
  
- Question 2: EXEC CICS is coded in margin B of COBOL program.
  - True/False



# **Customer Information Control System**

Lesson 6: CICS Copy Books

## Lesson Objectives

- Execute Interface Block – DFHAID
- Copy Book DFHBMSCA



6.1: Exec Interface Block (EIB)

## Use of Exec Interface Block (EIB)

- Exec Interface Block (EIB) in Linkage section:
  - It keeps record of system related information.
  - It creates one EIB per task.
  - EIB for a task contains information about the task, which is initiated.
  - It automatically includes copy of DFHEIBLK as EIB in the Linkage section of application program during translation.
  - Program can only access data using EIB field names.
  - User should not update data in EIB fields.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

**Note:**

Execute Interface Block (EIB) is a CICS area that contains information related to the current task.

6.2: EIB Fields

## List of EIB Fields

- Here is a list of EIB fields:
  - EIBAID
  - EIBCALEN
  - EIBCPOSN
  - EIBDATE
  - EIBTIME
  - EIBDS
  - EIBRCODE
  - EIBREQID
  - EIBTRMID
  - EIBTRNID
  - EIBTASKN

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

6.2: EIB Fields

## Use of EIB Fields

- EIBAID: It is a one character field that indicates which attention key was used for the last receive command.
- EIBFN: It indicates which CICS command was executed last.
- EIBRCODE: This will contain the response code of the last CICS command. It indicates the command's completion status.
- EIBDS: It contains the name of the dataset processed by the last file control command.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

6.2: EIB Fields

## Use of EIB Fields

- **EIBCPOSN:**
  - This field supplies the screen position of the cursor.
  - The cursor position is a number from 0 to 1919, indicating the cursor's displacement on the screen.
    - Example: cursor position 255 is line 4, column 16 ( $255/80 = 3$ , remainder =15)
  - The above calculation is for terminals with 80 character lines.
  - The integer portion of the answer plus 1 is the line number, the remainder plus 1 is the column number.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

6.2: EIB Fields

## Use of EIB Fields

- **EIBTIME:**
  - It contains the time of the day that the task was started. The time is stored as a seven digit packed decimal number in the form 0HHMMSS (one leading digit followed by hours, minutes, and seconds).
  - This assumes a 24 Hr clock:
    - 2.00 pm is hour 14.
    - 38 seconds after 2:41 PM is stored as 0144138.
    - Midnight is stored as 0000000.
    - One second before is stored as 0235959.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

6.2: EIB Fields

## Use of EIB Fields

- **EIBDATE:**
  - It contains the date on which the task was started. The format of EIBDATE is 00YYDDD, where DDD represents the three digits that indicates the number of the day in the year.
    - Thus July 1 1983 is stored as 0083182 (the 182nd day of 1983).
  - Although the YYDDD format is useful for comparison, it is inappropriate for display purposes.
  - Hence if you want to display the current date, you can first convert EIBDATE into standard format.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

6.2: EIB Fields

## Use of EIB Fields

- **EIBTRNID:**
  - It contains the transaction identifier that started the task.
  - One of the common uses of EIBTRNID is to determine how your program was started.
  - This field can be used to ensure that the program is invoked only from a menu. In other words, a program cannot be started by entering a transaction identifier at the terminal.
  - Example: If EIBTRNID is anything other than MENU (the transaction identifier that starts the menu program), a return command is used to terminate the task.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 9

6.2: EIB Fields

## Use of EIB Fields

- **EIBTRMID:**
  - This field supplies the name of the terminal running the task.
  - The terminal name that appears in the EIBTRMID is not a physical device like 3270-2, rather it is a symbolic name assigned to a terminal to meet installation specific requirement.
  - These names are used for security purposes.
  - This can be used to restrict a program to certain terminals.
  - The EIBTRMID can be checked to ensure that the TERMINAL is eligible to run the task.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 10

6.2: EIB Fields

## Use of EIB Fields

- **EIBREQID:**
  - It contains the request identifier assigned to an interval control command by CICS.
- **EIBRESP:**
  - It contains a binary number corresponding to the condition that has been raised. Generally, it is used with DFHRESP.
- **EIBTASKN:**
  - It contains the task number assigned to the task by CICS.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 11

6.2: DFHAID – Attention Identifier (AID)

## Concept of DFHAID

- Attention Identifier indicates the method that the terminal operator has used to initiate the transfer of information from the terminal device to CICS.
- AID keys are PF keys, PA keys, ENTER key, and CLEAR key.
- The EIBAID field in EIB contains the AID code of the most recently used AID.
- CICS provides the standard AID list in the form of copy library member (DFHAID).

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 12

**Note:**

A program can use the standard AID list by specifying in the program:  
COPY DFHAID.

6.2: DFHAID – Attention Identifier (AID)

## Concept of DFHAID

- The DFHAID member contains AID codes as shown below:
  - DFHENTER: ENTER key
  - DFHCLEAR: CLEAR key
  - DFHPA1-3: PA1 TO PA3 keys
  - DFHPF1-24: PF1 TO PF24 keys

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 13

6.3: Using AID Information in a Program

## Illustration

- Let us see an example on using AID information:

```
IF EIBAID = DFHPF3
    PERFORM 2100-END-ROUTINE
ELSE IF EIBAID = DFHPA1
    PERFORM 2200-CANCEL-ROUTINE
ELSE IF EIBAID = DFHENTER
    PERFORM 2300-NORMAL-ROUTINE
ELSE
    PERFORM 2400-WRONG-ROUTINE.
```



Copyright © Capgemini 2015. All Rights Reserved 14

6.4: DFHBMSCA – Standard Attribute Byte List

## Concept of DFHBMSCA

- A field's attribute can be changed by moving a value to the corresponding attribute field in the symbolic map.
- This feature can be used to highlight errors detected by an edit module.
- To make it easy to modify attribute characters, IBM supplies a standard COPY member named DFMHBMSCA.
- Example:
  - COPY DFHBMSCA
  - MOVE DFHBMPRO to NAMEA

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 15

### DFHBMSCA – Standard Attribute Byte List:

Constants	Meaning
DFHBMPEM	Printer end-of-message
DFHBMPNL	Printer new line
DFHBMASK	AutoSkip
DFHBMUNP	Unprotected
DFHBMUNN	Unprotected & num
DFHBMPRO	Protected
DFHBMPRY	Bright
DFHBMDAR	Dark
DFHBMFSE	MDT set
DFHBMPRF	Protected and MDT set
DFHBMASF	Auto-Skip & MDT set
DFHBMASB	Auto-Skip & bright

To dynamically assign an attribute, the copy book "DFHBMSCA" has to be included in Working Storage section of the application program.

When the map is sent through SEND MAP command, the new attribute will be in effect on the field on subject, overriding the original attribute defined at the map definition time.

6.5: DFHBMSCA

## Example of DFHBMSCA

- Let us see an example of DFHBMSCA:

```
WORKING-STORAGE SECTION.  
-----  
COPY 'MAPSETA'.  
-----  
COPY 'DFHBMSCA'.  
-----  
PROCEDURE DIVISION.  
-----  
MOVE DFHBMBRY TO CUSTNOA.  
MOVE DFHDMPRO TO CUSTNAMEA.  
EXEC CICS SEND  
    MAP ('MAPNAME')  
    MAPSET('MAPSETA')  
    FROM(MAPSETAI)  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16

### DFHBMSCA – Standard Attribute Byte List:

#### **Dynamic Attribute Character Assignment:**

It can be used to change the default attribute character which has been defined in the BMS map.

It can be done by placing the predefined attribute character to the fieldname+A of the field to which you wish to dynamically assign the attribute character.

6.5: FACDEFN

## Concept of FACDEFN

- Using DFHBMSCA for changing attributes is cryptic.
- Also, it does not include some of the most commonly used attribute bytes. Furthermore, most of the definitions it does include are rarely used.
- Hence an improved copy member for the attribute definitions is created.
- The library member FACDEFN contains standardized definitions for attribute characters.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 17

**FACDEFN:****Field-Attribute-Definition:**

05 FAC-UNPROT	PIC X VALUE ''.	
05 FAC-UNPROT-MDT	PIC X VALUE 'A'.	
05 FAC-UNPROT-BRT	PIC X VALUE 'H'.	
05 FAC-UNPROT-BRT-MDT	PIC X VALUE 'I'.	
05 FAC-UNPROT-DARK	PIC X VALUE '<'.	
05 FAC-UNPROT-DARK-MDT	PICX VALUE '('.	
05 FAC-UNPROT-NUM	PIC X VALUE '&'.	
05 FAC-UNPROT-NUM-MDT	PIC X VALUE 'J'.	
05 FAC-UNPROT-NUM-BRT	PIC X VALUE 'Q'.	
05 FAC-UNPROT-NUM-BRT-MDT	PIC X VALUE 'R'.	
05 FAC-UNPROT-NUM-DARK	PIC X VALUE '*'.	
05 FAC-UNPROT-NUM-DARK-MDT	PIC X VALUE ')'. 05 FAC-PROT	PIC X VALUE '-'.
05 FAC-PROT-MDT	PIC X VALUE '/'.	
05 FAC-PROT-BRT	PIC X VALUE 'Y'.	
05 FAC-PROT-BRT-MDT	PIC X VALUE 'Z'.	

6.5: FACDEFN

## The Syntax

COPY BOOKS.  
..  
..  
MOVE FAC-PROT-SKIP TO CUSTNOA.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 18

**FACDEFN:**

05 FAC-PROT-DARK VALUE '%'.	PIC X
05 FAC-PROT-DARK-MDT	PIC X VALUE '_'. PIC X VALUE '0'.
05 FAC-PROT-SKIP	PIC X VALUE '5'.
05 FAC-PROT-SKIP-MDT	PIC X VALUE '8'.
05 FAC-PROT-SKIP-BRT	PIC X VALUE '9'.
05 FAC-PROT-SKIP-DARK	PIC X VALUE '@'.
05 FAC-PROT-SKIP-DARK-MDT	PIC X VALUE QUOTE

## Summary

- EIB fields
- Copy book DFHAID
- Copy book DFHEIBLK, DFHBMSCA



## Review Questions

- Question 1: The \_\_\_ field supplies the name of the terminal running the task.
- Question 2: \_\_\_ contains the request identifier assigned to an interval control command by CICS.
- Question 3: \_\_\_ contains the transaction identifier that started the task.



# **Customer Information Control System**

Lesson 7: Application Program  
Housekeeping

## Lesson Objectives

- In this lesson, you will learn about:
  - ABEND codes
  - HANDLE CONDITION
  - NOHANDLE
  - HANDLE AID



7.1: ABEND Codes

## Concept of ABEND Codes

- If an exceptional condition occurs during execution of a CICS application program, and if program does not check the exceptional condition, then by default CICS will terminate the task.
- This kind of termination is known as abnormal termination and CICS will issue the abnormal termination (ABEND) code, which is associated with the exceptional condition on subject.



Copyright © Capgemini 2015. All Rights Reserved 3

7.2: Exception Handling

## Concept of Exception Handling

- Exceptional Conditions:
  - An abnormal situation during execution of a CICS command is called an “exceptional condition”.
- Exceptional conditions can be handled using any of the following options:
  - HANDLE CONDITION command
  - IGNORE CONDITION command
  - NOHANDLE option

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

7.3: HANDLE CONDITION

## Use of HANDLE CONDITION

- The HANDLE CONDITION command can be used to specify the actions that the program should take when certain exceptional conditions are raised.
- Once a HANDLE CONDITION command request has been made, it remains active until the task terminates or another HANDLE Command for that condition is encountered or IGNORE Command for that condition is encountered

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 5

**HANDLE CONDITION:**

- The “condition” represents an exceptional condition.
  - If a “label” is specified, then the control will be passed to the labeled paragraph when the specified condition occurs.
  - If no label is specified, then the previously set HANDLE CONDITION request will be cancelled and the default action will be taken.
- The general error condition (ERROR) can be specified within the same list to specify that all other conditions cause control to be passed to the label specified.
- Not more than 16 conditions can be specified in a single HANDLE CONDITION command.
- More than one HANDLE CONDITION command can be specified in the program.
- The CICS command executed last is shown in a system field EIBFN, while the response code for the last CICS command is shown in another system field EIBRCODE.
- The exceptional condition is the interpretation of the response code in the EIBRCODE field based on the command code in the EIBFN field.

7.3: HANDLE CONDITION

## Use of HANDLE CONDITION

- The HANDLE CONDITION request is effective only within the program, which issues the HANDLE CONDITION command. That is, if the control is passed to another program, the HANDLE CONDITION request in the calling program will no longer be honored in the called program.

```
EXEC CICS HANDLE CONDITION
```

```
    condition(label)
```

```
    [ condition(label) ]
```

```
    [ERROR(label)]
```

```
END-EXEC
```

Copyright © Capgemini 2015. All Rights Reserved 6

7.3: Example of HANDLE CONDITION  
**Illustration**

```
2110-RECEIVE-CHOICE SECTION.  
  EXEC CICS HANDLE CONDITION  
    MAPFAIL(2110-MAP-FAIL)  
  END-EXEC.  
  EXEC CICS RECEIVE MAPSET('MENUMAP')  
    MAP('MENUMAP')  
    INTO(MENUMAPI)  
  END-EXEC.  
  GO TO 2110-EXIT.  
2110-MAP-FAIL.  
  MOVE 'MAP FAIL SESSION ENDED' TO WS-END-SESSION-MSG.  
  MOVE 'Y' TO WS-END-SESSION.  
2110-EXIT.  
  EXIT.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

**Example of HANDLE CONDITION:**

- The HANDLE CONDITION command is used to transfer control to the specified procedure label if the exceptional specified condition occurs.
- Once a HANDLE CONDITION command request has been made, it remains active until the end of the program or another HANDLE CONDITION request overrides it. In order to avoid the confusion over which HANDLE CONDITION request is active, it is strongly recommended that a HANDLE CONDITION request always be paired with a CICS command.

7.3: Example of HANDLE CONDITION  
**Illustration**

- The effect of a HANDLE CONDITION command can be nullified by listing the condition name without a paragraph or section name.

```
EXEC CICS
HANDLE CONDITION NOSPACE
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 8

**Example of HANDLE CONDITION:**

The command in the above slide reverses the effect of any previous HANDLE CONDITION command for NOSPACE condition. Hence if the NOSPACE condition is raised after the execution of this command, the task is terminated.

7.3: HANDLE CONDITION – ERROR

## The ERROR Condition

- A special condition-name called ERROR can be used to trap any errors that are not specifically named in a HANDLE CONDITION name.
- The routine specified for the ERROR condition will be invoked for any exceptional condition that is not specifically handled.



Copyright © Capgemini 2015. All Rights Reserved 9

7.3: HANDLE CONDITION – ERROR  
**Illustration**

- If the NOSPACE condition is raised, then the control transfers to 310-ERROR since ERROR is coded but not NOSPACE.
- However, NOTOPEN and DUPKEY are processed by their respective error-handling routines.

```
EXEC CICS  
HANDLE CONDITION DUPKEY(310-DUPKEY)  
NOTOPEN(310-NOTOPEN)  
ERROR(310-ERROR)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 10

7.4: IGNORE CONDITION Command

## Concept of IGNORE CONDITION Command

- IGNORE CONDITION causes no action to be taken if the condition specified occurs in the program.
- Control returns to the next instruction following the command, which encountered the exceptional condition.
- The request by the IGNORE CONDITION command is valid until the subsequent HANDLE CONDITION command for the same condition.



Copyright © Capgemini 2015. All Rights Reserved 11

7.4: IGNORE CONDITION Command

## The Syntax for IGNORE Condition

- The syntax for IGNORE CONDITION is shown below:

```
EXEC CICS IGNORE CONDITION  
    condition  
    [condition]  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 12

7.4: IGNORE CONDITION Command

## Example for IGNORE Condition

- Let us see an example for IGNORE CONDITION:

```
EXEC CICS IGNORE CONDITION  
  LENGERR  
END-EXEC.  
EXEC CICS READ  
  INTO (FILE-AREA)  
  DATASET ('MASTER')  
  RIDFLD (RECORD-KEY)  
  LENGTH (LEN)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved. 13

### IGNORE CONDITION command:

#### Example:

- At the end of the READ command, if the LENGERR condition occurs, the condition will be ignored and the control will be passed to the next statement after the READ command.
- You can choose an IGNORE CONDITION command if you have a program reading records that are sometimes longer than the space you provide, but you do not consider this an error and do not want anything done about it. You may, therefore, code IGNORE CONDITION LENGERR before issuing READ commands.

7.4: IGNORE CONDITION Command

## Example for IGNORE Condition

```
EXEC CICS HANDLE CONDITION  
  DUPREC(DUPRTN)  
  LENGERR  
  ERROR(ERRHANDL)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 14

### IGNORE CONDITION command:

#### **Example:**

- Since the code shown in the above slide initially ignores the condition LENGERR, nothing happens if the program raises a LENGERR condition. The application simply continues its processing. Of course, if the fact that LENGERR has arisen means that the application cannot sensibly continue, then you have a problem.
- Later in the code, you can explicitly set condition LENGERR to the system default action by naming it in a HANDLE CONDITION command without a label. When this command has been executed, the program no longer ignores condition LENGERR, and if it occurs afterwards, it causes the system default action.

7.5: NOHANDLE Option

## Concept of NOHANDLE

- The NOHANDLE option can be specified in any CICS command.
- It causes no action to be taken for any exceptional condition occurring during execution of this command.
- Example:

```
EXEC CICS SEND  
  MAP(---)  
  MAPSET(-----)  
  FROM(-----)  
  LENGTH(---)  
  NOHANDLE  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 15

7.5: NOHANDLE Option

## NOHANDLE – RESP Option

- If RESP option is specified in a command, then CICS places a response code at the completion of the command.
- The application program can check this code, then proceed to the next processing.
- If the RESP option is specified in a command, then the NOHANDLE option is applied to this command.
- Therefore the HANDLE CONDITION requests will have no effect in this case.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16

**NOHANDLE Option:****NOHANDLE - RESP Option:**

- NOHANDLE with the RESP option is an alternative to HANDLE CONDITION command.
- NOHANDLE with RESP option eliminates the need for HANDLE CONDITION command. The RESP option eliminates the need for the HANDLE CONDITION command, GO TO statements, paragraph labels to process specific exceptional conditions and sections. Instead, you simply follow each CICS command with IF statement that test the response code to determine if errors have occurred.

7.5: NOHANDLE Option

## Example of NOHANDLE – RESP Option

- Let us see an example of NOHANDLE with RESP option:

```
EXEC CICS RECEIVE  
    MAP(---)  
    MAPSET(-----)  
    INTO(-----)  
    LENGTH(---)  
    NOHANDLE  
    RESP(WS-RESP-FIELD)  
END-EXEC.  
IF WS-RESP-FIELD = DFHRESP(NORMAL)  
-----  
ELSE IF WS-RESP-FIELD = DFHRESP(MAPFAIL)  
-----
```



Copyright © Capgemini 2015. All Rights Reserved. 17

### NOHANDLE Option:

#### **Example of NOHANDLE – RESP Option:**

If RESP is specified on a CICS command, then we instruct CICS to ignore any exceptional condition that is raised by the CICS command. Instead, CICS places a numeric value that represents the condition status of the command in the response code field. It is up to the programmer to evaluate this field to make sure that the command executes properly.

7.6: RESP Option

## Procedure to Utilize RESP Option

- Here are the steps for using RESP option:
  - Define a fullword binary field (S9(8) COMP) in the Working Storage Section as the response field.
  - Place the RESP option with the response field in a command (any CICS command).
  - After command execution, check the response code in the response field with DFHRESP(xxxx), where xxxx is:
    - NORMAL for normal completion
    - Any exceptional condition



Copyright © Capgemini 2015. All Rights Reserved 18

7.7: HANDLE AID Command

## Concept of HANDLE AID Command

- HANDLE AID command is used to specify the label to which control needs to be passed when the specified AID is received.
  - This is one way of substituting the EIBAID checking approach.
  - AID keys are PA keys, PF keys, ENTER and CLEAR key.
  - CLEAR and any of the PA keys do not transmit data.
  - Format:

```
EXEC CICS HANDLE AID  
    Option(label)  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 19

### HANDLE AID Command:

- The HANDLE AID command tells the program what to do when the user presses an attention key.
- The HANDLE AID key itself does not detect the use of AID key. It is the receive map command which does that. Hence HANDLE AID key is always used alongside a RECEIVE map command.
- In effect, HANDLE AID sets up the processing to be done if the RECEIVE MAP command detects the use of an AID key.

7.7: HANDLE AID Command

## Concept of HANDLE AID Command

- Commonly used options are given below:
  - Any key name (PA1 to PA3, PF1 to PF24, ENTER, CLEAR)
  - ANYKEY (Any of the above except ENTER key)
- Use of AID key is detected by RECEIVE MAP command.
- HANDLE AID has a higher precedence over HANDLE CONDITION.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 20

### HANDLE AID Command:

- Suppose both HANDLE AID and HANDLE CONDITION are in effect when you issue a RECEIVE MAP command, and an AID key is detected. Then control will be transferred to the procedure specified for the AID key.
- Any HANDLE CONDITION command will be ignored.

7.7: HANDLE AID Command

## Example of HANDLE AID Command

- Let us see an example of HANDLE AID command:

```
EXEC CICS HANDLE AID  
    PF3(2100-END-ROUTINE)  
    PA1(2100-CANCEL-ROUTINE)  
    ENTER(2100-NORMAL-ROUTINE)  
    ANYKEY(2100-WRONG-KEY-ROUTINE)  
END-EXEC.  
EXEC CICS RECEIVE MAP('MAP1')  
    MAPSET('MAP1')  
    INTO (MAP1I)  
    LENGTH(WS-MAP-LENGTH)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved. 21

### HANDLE AID Command:

#### Example:

- To code an HANDLE AID command, you list one or more attention keys, along with the name of the paragraph that will process each key. Then when a subsequent RECEIVE MAP detects that an attention key has been used, CICS transfers control to the corresponding label specified in a previous HANDLE AID command. CICS transfers control to the appropriate LABEL with a GO TO.
- The HANDLE AID forces you to issue a RECEIVE MAP command just to detect the use of an attention key.
- In many cases, you don't need the input data retrieved by the receive map command to process the attention key. For example, if the user presses the PF3 key to end the program, then all that your program has to do is to issue an XCTL command to transfer to a menu program. In this case, receiving the map is simple a waste of time.

7.7: HANDLE AID Command

## Disadvantage of HANDLE AID Command

- Let us see a scenario in which it will be disadvantageous to use HANDLE AID command:

```
2100-RECEIVE-MAP SECTION.  
EXEC CICS HANDLE CONDITION  
    MAPFAIL(2100-MAPFAIL-ROUTINE)  
END-EXEC.  
EXEC CICS HANDLE AID  
    PF3(2100-END-ROUTINE)  
    PA1(2100-CANCEL-ROUTINE)  
    ENTER(2100-NORMAL-ROUTINE)  
    ANYKEY(2100-WRONG-KEY-ROUTINE)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 22

7.7: HANDLE AID Command

## Disadvantage of HANDLE AID Command

```
EXEC CICS RECEIVE MAP('MAP1')
  MAPSET('MAP1')
  INTO (MAP1I)
  LENGTH(WS-MAP-LENGTH)

END-EXEC.

2100-MAPFAIL-ROUTINE.
-----
2100-NORMAL-ROUTINE.
-----
..
2100-EXIT.

EXIT
```



Copyright © Capgemini 2015. All Rights Reserved. 23

7.8: EIBAID

## Concept of EIBAID

- The 3270 terminal transmits the AID character to the EIBAID field of EIB, at the time of task initiation.
- EIBAID can be used to find the AID key that has been pressed by user.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 24

7.8: EIBAID

## Concept of EIBAID

- CICS copy book DFHAID fields are used to identify the pressed key.

IF EIBAID = DFHAID(ENTER)  
OR  
IF EIBAID = DFHENTER
- EIBAID value can be checked even without receiving map.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 25

## Summary

- In this lesson, you have learnt:
  - Handling exceptions using HANDLE CONDITION
  - Use of HANDLE AID and NOHANDLE commands
  - Difference between using HANDLE AID and EIBAID



## Review Questions

- Question 1: HANDLE AID has a higher precedence over HANDLE CONDITION.
  - True/False
  
- Question 2: If the RESP option is specified in a command, then the \_\_\_ option is applied to this command.
  
- Question 3: EIBAID value can be checked even without receiving map.
  - True/False



# **Customer Information Control System**

Lesson 8: BMS Programming  
Considerations

## Lesson Objectives

- In this lesson, you will learn about:
  - Dynamic Attribute Character Assignment
  - Cursor Positioning Techniques



Copyright © Capgemini 2015. All Rights Reserved. 2

8.1: DFHBMSCA – Standard Attribute Byte List

## Concept of DFHBMSCA

- A field's attribute can be changed by moving a value to the corresponding attribute field in the symbolic map.
- This feature can be used to highlight errors detected by an edit module.
- To make it easy to modify attribute characters, IBM supplies a standard COPY member named DFMHBMSCA.
- For example:
  - COPY DFHBMSCA
  - MOVE DFHBMPRO to NAMEA



Copyright © Capgemini 2015. All Rights Reserved. 3

### DFHBMSCA – Standard Attribute Byte List:

Constants	Meaning
DFHBMPEM	Printer end-of-message
DFHBMPNL	Printer new line
DFHBMASK	AutoSkip
DFHBMUPN	Unprotected
DFHBMUNN	Unprotected & num
DFHBMPRO	Protected
DFHBMPRY	Bright
DFHBMDAR	Dark
DFHBMFSE	MDT set
DFHBMPRF	Protected and MDT set
DFHBMASF	Auto-Skip & MDT set
DFHBMASB	Auto-Skip & bright

To dynamically assign an attribute, the copy book "DFHBMSCA" has to be included in Working Storage section of the application program.

When the map is sent through SEND MAP command, the new attribute will be in effect on the field on subject, overriding the original attribute defined at the time of map definition.

8.1; DFHBMSCA – Standard Attribute Byte List

## Example of DFHBMSCA

```
WORKING-STORAGE SECTION.  
-----  
COPY 'MAPSETA'.  
-----  
COPY 'DFHBMSCA'.  
-----  
PROCEDURE DIVISION.  
-----  
MOVE DFHBMBRY TO CUSTNOA.  
MOVE DFHDMPRO TO CUSTNAMEA.  
EXEC CICS SEND  
    MAP ('MAPNAME')  
    MAPSET('MAPSETA')  
    FROM(MAPSETAI)  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

### Dynamic Attribute Character Assignment:

It can be used to change the default attribute character which has been defined in the BMS map.

It can be done by placing the predefined attribute character to the fieldname+A of the field to which you wish to dynamically assign the attribute character.

## 8.2: FACDEFN

# Concept of FACDEFN

- Using DFHBMSCA for changing attributes is cryptic.
- Also, it does not include some of the most commonly used attribute bytes. Furthermore, most of the definitions it does include are rarely used.
- Hence an improved copy member for the attribute definitions is created.
- The library member FACDEFN contains standardized definitions for attribute characters.



Copyright © Capgemini 2015. All Rights Reserved 5

8.2: FACDEFN

## Concept of FACDEFN

- Let us see an example using FACDEFN:

```
COPY FACDEFN.  
..  
..  
MOVE FAC-PROT-SKIP TO CUSTNOA
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

**FACDEFN:**

05 FAC-PROT-DARK VALUE '%'.	PIC X	
05 FAC-PROT-DARK-MDT	PIC X VALUE '_'. 05 FAC-PROT-SKIP	PIC X VALUE '0'.
05 FAC-PROT-SKIP-MDT VALUE '5'.	PIC X	
05 FAC-PROT-SKIP-BRT VALUE '8'.	PIC X	
05 FAC-PROT-SKIP-BRT-MDT	PIC X VALUE '9'.	
05 FAC-PROT-SKIP-DARK	PIC X VALUE '@'.	
05 FAC-PROT-SKIP-DARK-MDT	PIC X VALUE QUOTE	

8.3: Cursor Positioning Techniques

## Different Techniques

- CICS provides three techniques to set the cursor position:
  - Static Cursor Positioning
  - Dynamic / Symbolic Cursor Positioning
  - Relative Cursor Positioning

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

### Cursor Positioning Techniques:

When a SEND MAP command is issued, by default CICS will place the cursor in the top left corner of the screen. This forces the user to press the **Tab** key to move to the first data entry field.

#### Checking Cursor Position:

The cursor position can be known by checking the EIBCPOSN , which is a half word binary S9(4) COMP. This can be checked after the completion of the RECEIVE command. The EIBCPOSN gives the relative position of the cursor at the time of data transfer.

8.4: Static Cursor Positioning

## Concept of Static Cursor Positioning

- In the Static Cursor Positioning approach, a cursor position is defined in a map by placing "IC" in the ATTRB parameter of the DFHMDF macro for a particular field.

```
DFHMDF      POS=(3,16),
             ATTRB=(UNPROT, FSET, IC)
             LENGTH=8
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

**Static Cursor Positioning:**

When the map is sent, the cursor will appear in this field specified with the IC attribute.

IC stands for Initial Cursor.

If more than one field with IC is specified in one map, then the last IC will be honored.

8.5: Symbolic Cursor Positioning

## Concept of Symbolic Cursor Positioning

- In the Symbolic Cursor Positioning approach, a cursor is dynamically positioned through an application program using a symbolic name of the symbolic map by placing -1 into the field's length field (that is, filename+L) of the field.

```
MOVE -1 TO CHOICEL.  
EXEC CICS SEND MAP('MAP1')  
    MAPSET('MAP1')  
    CURSOR  
    ERASE  
END-EXEC
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 9

**Note:**

The SEND MAP command to be issued must have the CURSOR option (without argument). Also, the mapset must be coded with MODE=INOUT in the DFHMSD macro.

8.6: Relative Cursor Positioning

## Concept of Relative Cursor Positioning

- In the Relative Cursor Positioning approach, you dynamically position a cursor through an application program using the CURSOR (data-value) option in the SEND MAP command.
- Data-value is calculated as:  $(\text{row}-1) * 80 + (\text{column} - 1)$
- Example: If you want to place the cursor in column 17 of row 12. The correct displacement will be 896:
  - $(12-1)*80 + (17-1) = 896$

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 10

### Relative Cursor Positioning:

The map will be displayed with the cursor at the specified position, overriding the static cursor position defined at the map definition time. Direct cursor positioning has two major drawbacks:

    Cursor displacements are awkward to use.

    Direct positioning ties the program to specific screen locations. Hence if the mapset is changed by moving a field from one screen location to another, the program has to be changed as well.

8.6: Relative Cursor Positioning

## Example of Relative Cursor Positioning

- Let us see an example of Relative Cursor Positioning:

```
EXEC CICS SEND  
    MAP(-----)  
    MAPSET(-----)  
    CURSOR(896)  
    ERASE  
END-EXEC
```



Copyright © Capgemini 2015. All Rights Reserved 11

## Summary

- In this lesson, you have learnt:
  - A method to dynamically assign attributes
  - A method to dynamically position cursor



## Review Questions

- Question 1: In \_\_\_ approach, a cursor is dynamically positioned through an application program by placing -1 into the field's length field.
- Question 2: If IC is set for multiple field, the first IC will be honored.
  - True / False
- Question 3: Direct cursor positioning is the most preferred way of positioning the cursor.
  - True / False



# **Customer Information Control System**

Lesson 9: Terminal Control  
Operations

## Lesson Objectives

- In this lesson, you will learn about:
  - SEND MAP COMMAND
  - RECEIVE MAP COMMAND
  - SEND TEXT
  - RECEIVE TEXT



9.1: Functions of CICS Command for BMS

## Basic Functions

- The CICS commands for BMS perform the following three basic functions:
  - Map Sending Function
  - Map Receiving Function
  - Text Handling Function

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

### Functions of CICS Command for BMS:

- The CICS commands for BMS perform the following three basic functions:
  - **Map Sending Function:** Using the data in the symbolic map, BMS prepares the corresponding physical map, and sends to the terminal.
  - **Map Receiving Function:** Using the input from the terminal, BMS prepares data in the symbolic map through the corresponding physical map.
  - **Text Handling Function:** BMS prepares text without using a map and sends to the terminal.

9.1: Functions of CICS Command for BMS

## Basic Functions

- The following commands are available for the basic BMS input / output operations:
  - RECEIVE MAP: Formatted data transfer, to receive a map
  - SEND MAP: Formatted data transfer, to send a map
  - RECEIVE: Unformatted data transfer, to receive text
  - SEND TEXT: Unformatted data transfer, to send text

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

9.2: RECEIVE MAP Command

## Use of RECEIVE MAP Command

- RECEIVE MAP command is used to receive a map from a terminal.
  - At the completion of the command, the symbolic map of the specified map will contain the data from the terminal in the following fields per each field defined by the DFHMD macro:
    - Fieldname+L
    - Fieldname+F
    - Fieldname+I

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

### RECEIVEMAP Command:

- **Fieldname+L:** It is the length field, which contains actual number of characters typed in the screen
- **Fieldname+F:** It is the flag byte field, which is normally X'00'. It will be X'80' if the screen field has been modified but cleared.
- **Fieldname+I:** It is the actual input data field.

9.2: RECEIVE MAP Command

## Format

- The RECEIVE MAP command when executed places all modified data items in the symbolic work area.

```
EXEC CICS RECEIVE MAP(map name)
    MAPSET(mapset name)
    INTO(data-area)
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

**RECEIVE MAP Command:**

- Map:** It specifies the name of the Map.
- Mapset:** It specifies the name of the mapset containing the map.
- INTO:** It specifies the symbolic map. If nothing else is specified, then the INTO option is assumed, and BMS automatically finds the symbolic map area to place the data from the terminal.

9.2: RECEIVE MAP Command

## Common Exceptional Conditions

- Common Exceptional Conditions thrown by the RECEIVE MAP Command:
  - MAPFAIL:
    - If the data to be mapped has the length of zero
    - If user presses the ENTER key without typing data
    - If map is received by pressing CLEAR or any of the PA keys

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

9.2: RECEIVE MAP Command

## Data Validity Checking

- After the completion of the RECEIVE MAP command, the data from the terminal is placed in the fields of the symbolic map.
- However, it cannot be assumed that all data in the symbolic map is valid, since BMS has nothing to do with the quality of data content in the fields which were entered by the terminal user.
- Hence data validity checking has to be performed right after the RECEIVE MAP command.



Copyright © Capgemini 2015. All Rights Reserved 8

9.2: RECEIVE MAP Command

## Procedure for Data Validity

- Following is a set of procedures for effective data validity checking after the completion of the RECEIVE MAP command:
  - Perform the data validity checking in detail as soon as the program receives data from the terminal.
  - Check the fieldname+L or fieldname+I
  - If an error is detected, prepare an error message. For this, each map should have error message fields.
  - Perform Data validation for all fields.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 9

**RECEIVE MAP Command:**

Following is a set of procedures for effective data validity checking after the completion of the RECEIVE MAP command:

1. Perform the data validity checking in detail as soon as the program receives data from the terminal.
2. Check the fieldname+L:
  - If it is zero, the user has entered no data. Therefore take the default action, or consider an error if it is the required field.
  - If it is a positive value, some data of that length has been entered. Therefore validate the field data in the fieldname+I.
3. Alternatively, directly check the fieldname+I:
  - If it is LOW-VALUES or SPACE, then no data has been entered or space key has been pressed. Therefore take the default action, or consider an error if it is the required field.
  - If it is not LOW-VALUE or SPACE, then some data has been entered. Therefore validate the field data in the fieldname+I.
  - Note that CICS considers space characters as data. Therefore space may be valid data, depending on the application specifications.
4. If an error is detected, prepare an error message. For this, each map should have the error message fields.
5. Do not stop validity checking at the first error detection. Go to the next field for another validation.
6. Repeat this until all data fields have been verified.
7. At completion of the data validation, if there are errors, send error message(s) to the terminal for the user to correct them and reenter.

9.3: SEND MAP Command

## Use of SEND MAP Command

- SEND MAP command is used to send a map to a terminal.
- Before issuing this command, the application program must prepare the data in the symbolic map of the map to be sent, which has following fields per each field defined by the DFHMDF macro:
  - Fieldname+L
  - Fieldname+A
  - Fieldname+O

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 10

**SEND MAP Command:**

**fieldname+L:** It denotes the length field, to which the application program does not have to prepare the data except for the dynamic cursor positioning.

**fieldname+A:** It denotes the attribute character field, to which the application program does not have to prepare the data except for the dynamic attribute character assignment.

**fieldname+O:** It denotes the actual output data field, to which the application program must place the data.

9.3: SEND MAP Command

## Format

- Given below is the format for SEND MAP command:

```
EXEC CICS SEND
    MAP(MAP-NAME)
    MAPSET(MAPSET-NAME)
    [FROM(DATA-AREA)]
    [CURSOR/CURSOR(DATA-VALUE)]
    [ERASE/ERASEAUP]
    [FREEKB]
    [ALARM]
    [FRSET]
    [DATAONLY/MAPONLY]
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 11

### SEND MAP Command:

**Map:** It specifies the name of the map. This value is supplied as a literal string in quotes.

**Mapset:** It specifies the name of the mapset which contains the map. Both map and mapset values are supplied as a literal string in quotes

**From:** This option names the symbolic map definition in the program's working-storage section. Even if not specified, this option is assumed, and BMS automatically finds the symbolic map area to take the data to the terminal.

**CURSOR:** It makes the application program able to position the cursor dynamically on any part of the screen.

**ERASE:** If this is specified, the current screen will be erased before the specified map appears on the screen. If this is not specified, then the specified map will be overwritten onto the current screen.

**ERASEUP:** It is used to erase all unprotected fields. The protected fields or attribute fields remain as they are in the current screen. The ERASEUP causes only data-entry fields to be erased.

**FREEKB:** It is used to free the keyboard.

**FRSET:** It is used to reset MDT to zero (that is, not modified) for all unprotected fields of the screen.

**DATAONLY:** It specifies that only application program data in the symbolic map needs to be sent to the terminal.

**MAPONLY:** It specifies that only the default data from the physical map needs to be sent to the terminal.

**SEND MAP Command:****Format for SEND MAP Command:**

- If you specify the **MAPONLY** option on a **SEND MAP** command, then only the data on the physical map is sent to the screen. Hence only screen headings will be displayed. If **DATAONLY** option is specified, then only the data from symbolic map is sent to the screen. Hence only the actual data values are sent to the screen, the headings are not.
- If both the options are omitted, then the data in the symbolic map is combined with the data in the physical map and both are sent to the screen.
- The first **SEND MAP** command executed in a terminal session usually sends both the headings and initial data to the screen. As a result, the first **SEND MAP** command is coded with neither the **DATAONLY** or **MAPONLY** option. Subsequent **SEND MAP** commands generally use the **DATAONLY** options so the headings are not transmitted again.
- Using the **DATA ONLY** option improves transmission time. The **ERASE** option causes CICS to erase the contents of the screen before any data is displayed. If **ERASE** is not specified, then the screen is not erased so any characters that are not overlaid by new data remain on the screen. **ERASE** is usually specified for the first **SEND MAP** that is issued by the program. On subsequent **SEND MAP** commands, the **ERASE** option is not coded, and particularly **DATONLY** option is coded. **ERASEUP** option specifies that only unprotected fields are erased, while **ERASE** completely clears the screen. **ERASEUP** causes only data entry fields to be erased. Whether you use **EARSEUP** option depends on whether all data-entry fields have to be erased between transmissions. In most applications, the **SEND MAP** command is coded with **DATAONLY** and **ERASEUP** options.

## 9.3: SEND MAP Command

**Example**

WORKING-STORAGE SECTION.

-----  
COPY MAPA.  
-----PROCEDURE DIVISION.  
-----EXEC CICS HANDLE CONDITION  
INVMPSZ(BIG-MAP)  
END-EXEC.  
-----EXEC CICS SEND MAP MAP('MAPA')  
MAPSET('MAPA')  
FROM(MAPA1)  
END-EXEC.  
-----BIG-MAP.  
ERROR PROCESSING.....

Copyright © Capgemini 2015. All Rights Reserved 13

9.4: RECEIVE Command (Unformatted Data Transfer)

## Use of RECEIVE Command

- Function:
- This command is used to receive an unformatted message from the terminal. In this data transfer, maps are not involved.
- Format:

```
EXEC CICS RECEIVE INTO(data-area)
  LENGTH(data-value)
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 14

## 9.4: RECEIVE Command (Unformatted Data Transfer)

**Example**

```
WORKING-STORAGE SECTION.  
01 MSG-LENGTH      PIC S9(4) COMP.  
01 INPUT-MSG.  
    02 TRANS-ID      PIC X(4).  
    02 FILLER        PIC X(1).  
    02 MESSAGE        PIC X(5).  
----
```

```
PROCEDURE DIVISION.  
MOVE 10 TO MSG-LENGTH  
EXEC CICS HANDLE CONDITION  
    LENGTHERR (LENGTH-ERROR-RT)  
END-EXEC  
----  
EXEC CICS RECEIVE INTO (INPUT-MSG)  
    LENGTH (MSG-LENGTH)  
END-EXEC  
----  
LENGTH-ERROR-RT.
```



Copyright © Capgemini 2015. All Rights Reserved. 15

9.5: SEND TEXT Command (Unformatted Data Transfer)

## Use of SEND TEXT Command

- Function:
  - SEND TEXT command is used to send unformatted message.
  - This message will always get displayed at row1, column1 position on the screen.
- Format:

```
EXEC CICS SEND TEXT FROM(data-area)
  LENGTH(data-value)
  [ERASE]
  [FREEKB]
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16

### SEND TEXT Command (Unformatted Data Transfer):

- In many cases, you want to send a short message to the terminal without having to create a BMS mapset. The SEND TEXT command allows you to do just that.
  - **From:** It contains the name of the field containing data to be transmitted to the terminal.
  - **Length:** It specifies the number of characters to be sent.
  - **Erase:** It specifies that the screen should be erased before any data is sent on the screen.
  - **Freekb:** It specifies that the keyboard should be unlocked after the data is sent else the operator has to press the RESET key.

9.5: SEND TEXT Command (Unformatted Data Transfer)

## Example

IDENTIFICATION DIVISION.

-----

WORKING-STORAGE SECTION.  
01 WS-MESSAGE PIC X(30) Value Spaces

PROCEDURE DIVISION.

-----  
MOVE 'END OF SESSION' TO WS-MESSAGE.  
EXEC CICS SEND TEXT  
    FROM (WS-MESSAGE)  
    LENGTH (30)  
    ERASE  
    FREEKB  
END-EXEC.  
-----



Copyright © Capgemini 2015. All Rights Reserved. 17

## Summary

- In this lesson, you have learnt:
  - Map Receiving Commands
  - Map Sending Commands
  - Commands for sending and receiving text



## Review Questions

- Question 1: The transmission time is improved by using the \_\_\_ option of the SEND MAP command.
  
- Question 2: \_\_\_ command is used to send unformatted message.
  
- Question 3: The Erase option is used in the SEND MAP command to erase the contents of protected field.
  - True / False



# **Customer Information Control System**

Lesson 10: Interval Control  
Operations

## Lesson Objectives

- In this lesson, you will learn about:
  - ASKTIME command
  - FORMATTIME command



10.1: Introduction

## Concept of Interval Control Operations

- The CICS Interval Control Program provides application program with time-controlled functions, such as the time-oriented task synchronization, providing current date and time, and automatic initiation of the time-ordered tasks.
- Two commands are prominently used:
  - ASKTIME Command
  - FORMATTIME Command

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

10.2: ASKTIME Command

## Use of ASKTIME Command

- Function:
  - This command is used to request the current date and time.
  - It also updates the EIBDATE and EIBTIME fields.
- Format:

```
EXEC CICS ASKTIME  
[ABSTIME(data-area)]  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

The EIBDATE and EIBTIME fields have the values at the task initiation time. Up on completion of this command, these fields will be updated to have the current date and time.

ABSTIME option is required if you want to store current date and time for further processing. Without this option ASKTIME will just update EIBDATE and EIBTIME field. The data-area is defined in Working-Storage Section with picture clause of S9(15) COMP. "data-area" after the execution gets the value in milliseconds.

10.3: FORMATTIME Command (Function)

## Use of FORMATTIME Command

- The FORMATTIME command is used to receive the information of date and time in various formats.



Copyright © Capgemini 2015. All Rights Reserved 5

10.3: FORMATTIME Command (Function)

## Format

Given below is the format for FORMATTIME command:

```
EXEC CICS FORMATTIME ABSTIME(data-area)
[YYDDD(data-area)]
[YYMMDD(data-area)]
[YYDDMM(data-area)]
[MMDDYY(data-area)]
[DDMMYY(data-area)]
[DATESEP(data-value)]
[DAYOFWEEK(data-area)]
[DAYOFMONTH(data-area)]
[MONTHOFYEAR(data-area)]
[YEAR(data-area)]
[TIME(data-area)[TIMESEP(data-value)]]]
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

**FORMATTIME Command (Function):****Format for FORMATTIME Command:**

- “data-area” for date and time are defined as X(08).
- Time separator by default is “:” and date separator by default is “/”(for this specify TIMESEP and DATESEP options without argument). If these two options are omitted, no separator is provided.

10.3: FORMATTIME Command (Function)

## Demo

- Demo on Display Current Date and Time



 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

## Summary

- In this lesson, you have learnt:
  - A method to extract date
  - A method to format date



## Review Questions

- Question 1: ASKTIME command is used to update the current date and time.
  - True / False
- Question 2: The default time separator is \_\_\_\_.
- Question 3: The default date separator is \_\_\_\_.



# **Customer Information Control System**

Lesson 11: Program Control  
Operations

## Lesson Objectives

- In this lesson, you will learn about:
  - Application Program Levels
  - Return, Link, and XCTL commands
  - Passing data through DFHCOMMAREA



11.1: Program Control Operations

## Introduction to PCP

- The CICS Program Control Program (PCP) governs a flow of control among CICS application programs and CICS itself.
- Following commands are available for the program controls services:
  - RETURN: To return to the next higher-level program or CICS
  - LINK: To pass control to another program at lower level
  - XCTL: To pass control to another program at the same level
  - LOAD: To load a program
  - RELEASE: To release a loaded program



Copyright © Capgemini 2015. All Rights Reserved. 3

**Note:**

The name of a CICS application program must be registered in the Processing Program Table (PPT) .

11.2: Application Program Levels

## Hierarchy in Program Levels

- The application programs under CICS run at various logical levels:
  - The first CICS application program to receive control from CICS within a task runs at the highest logical level 1.
  - A LINKed program runs at the next lower logical level from the LINKing program, while a XCTL'ed program runs at the same logical level as the XCTL'ing program.
  - The RETURN command always passes control back to the program at one logical level higher.



Copyright © Capgemini 2015. All Rights Reserved 4

11.3: RETURN Command  
**The Format**

- Given below is the format for RETURN command:

```
EXEC CICS RETURN  
  [TRANSID(name)  
   [COMMAREA(data-area)  
    [LENGTH(data-value)]]]  
  END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 5

### RETURN Command:

The above slide shows the format for RETURN command:

- TRANSID:** It specifies the one-to-four-character name of the transaction to be invoked when the user presses the attention key. The trans-id must be defined in the Program Control Table (PCT).
- COMMAREA:** It specifies a data area that is passed to the next execution of a pseudo-conversational program. The next program execution accesses the communication area via its DFHCOMMAREA field.
- LENGTH:** It specifies a binary halfword (PIC S9(4) COMP) or numeric literal that indicates the length of data area specified in the COMMAREA option.

11.3: RETURN Command

## Use of RETURN Command

- RETURN command returns the control to CICS.
  - If the RETURN command is coded without any options, then the control simply returns to CICS and the terminal session ends.
  - If the TRANSID option is coded, then CICS invokes the trans-id you specify the next time the operator presses an AID key.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

### RETURN Command:

- The COMMAREA and LENGTH options are used to pass a communication area to the next program.
  - COMMAREA specifies the name of the field in the working storage containing the data to be passed.
  - LENGTH specifies the number of bytes data to be passed.
- When the next program execution begins, the data passed via the communication area is available in the DFHCOMMAREA in its linkage section.

11.4: LINK Command

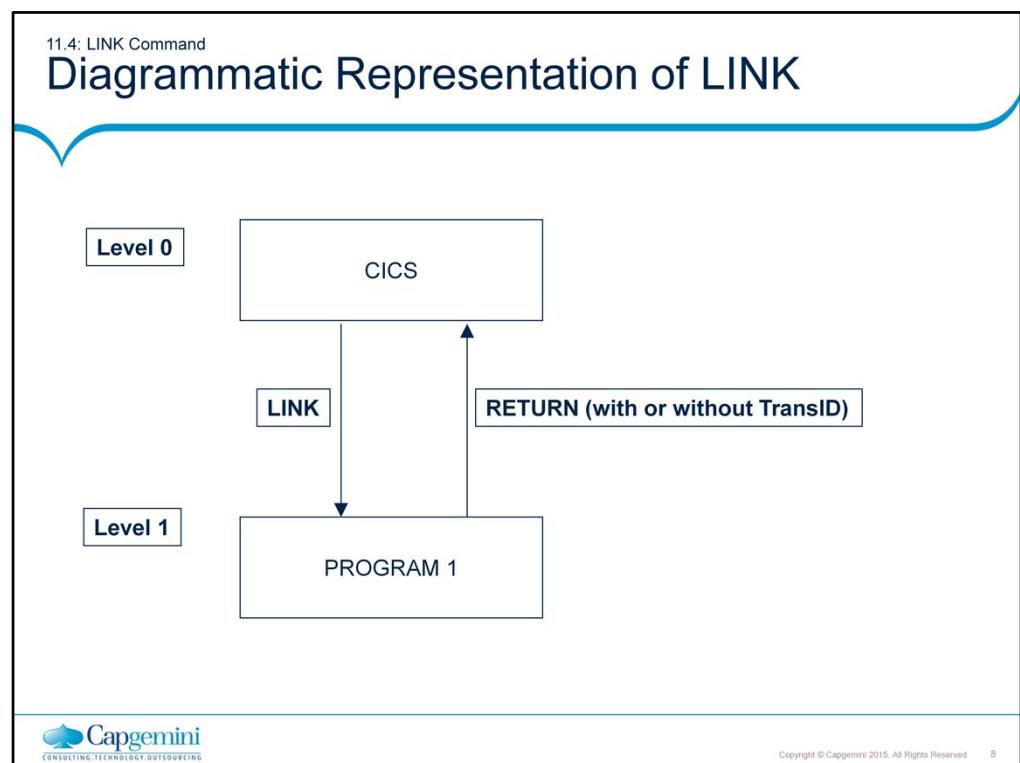
## Use of LINK Command

- LINK command is used to pass control from an application program at one logical level to another application at the next lower logical level.
- The calling program expects control to be returned to it.
- The control is returned at the next sentence when the called program completes execution.



Copyright © Capgemini 2015. All Rights Reserved

7

**LINK Command:**

- In the above diagram, CICS at level-0, invokes program1 at level-1, by linking to it.
- When program1 terminates, it issues a RETURN command to pass control back to CICS
- The RETURN command may or may not contain the TRANSID option.
  - If it does, the transaction identifier it specifies is invoked by CICS the next time the operator presses an AID key at the terminal.
  - If the RETURN command does not include the TRANSID option, CICS ends the terminal session.
- Regardless of whether the RETURN command includes the TRANSID option, it causes control to be passed, up one logical level (in this case, back to CICS).

11.4: LINK Command

## Format for LINK Command

- The PROGRAM command specifies the name of the program to be invoked (Program name should appear in the PPT).

```
EXEC CICS LINK  
  PROGRAM(name)  
  [COMMAREA(data-area)]  
  [LENGTH(data-value)]  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 9

### LINK Command:

- This is similar to the RETURN command. The difference is that the TRANSID option on the RETURN command requires a four character transaction, but the PROGRAM operand on the LINK requires an eight-character program name.
- Also, you can omit the TRANSID option on the RETURN command, but program is required on the LINK command.

11.4: LINK Command

## Example for LINK Command

- Code in Program 'Menu':

```
EXEC CICS HANDLE CONDITION  
  PGMIDERR(4000-PGM-NOT-FND)  
  ERROR(4000-GENERAL-ERROR)  
END-EXEC.  
EXEC CICS LINK  
  PROGRAM('FILE-MAINT')  
  COMMAREA(WS-COMM-AREA)  
  LENGTH(20)  
END-EXEC.
```

- Code in Program 'FILE-MAINT':

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

Takes control back in 'Menu'  
Program



Copyright © Capgemini 2015. All Rights Reserved 10

11.4: LINK Command

## Example for LINK Command

- Only the program at level-1 can issue a RETURN command with the TRANSID option.
- In case the program at level-2 or below issues a RETURN with options, the task terminates abnormally.

```
graph TD; CICS[CICS] -- "LINK" --> P1[Program 1]; P1 -- "LINK" --> P2[Program 2]; P2 -- "RETURN (without TransID)" --> CICS; P1 -- "RETURN (with or without TransID)" --> CICS;
```

**Capgemini**  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 11

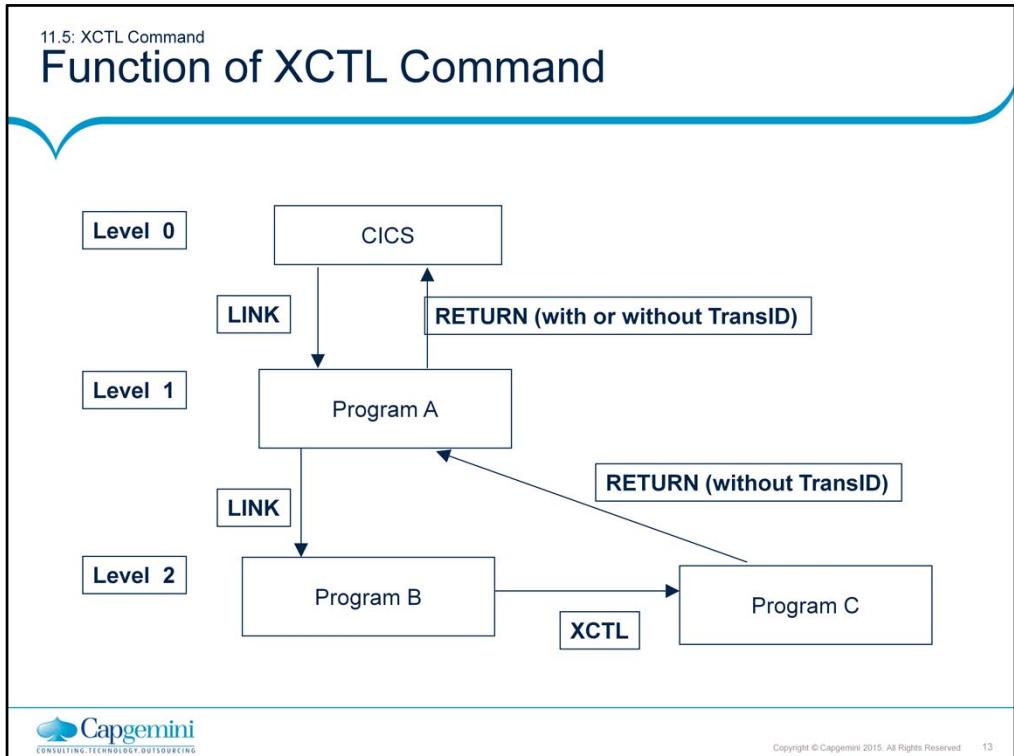
11.5: XCTL Command

## Use of XCTL

- The XCTL command is used to pass control from one application program to another application program at the same logical level.
- The calling program does not expect control to be returned.
- Data can be passed to the program through a special communication area called COMMAREA.
- Since this command requires less overhead, performance is relatively better in comparison to the LINK command.



Copyright © Capgemini 2015. All Rights Reserved 12



### XCTL Command:

- In the above diagram, CICS invokes Program A at level 1.
- Next Program A issues a LINK command to Program B at level 2.
- Then, Program B issues an XCTL command to invoke Program C at the same level.
- When Program C issues a RETURN command (without the TRANSID option), control is returned to the next higher level.
- So execution continues with the statement in Program A after the LINK command that invoked Program B.

11.5: XCTL Command

## Format for XCTL Command

- Topic Details - Line 2

```
EXEC CICS XCTL
  PROGRAM(name)
  [COMMAREA(data-area)]
  [LENGTH(data-value)]
END-EXEC
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 14

### XCTL Command:

The above slide shows the format for the XCTL command:

- **PROGRAM:** It specifies the one to eight Character name of the program to be invoked. This name must be defined in the Processing Program Table (PPT).
- **COMMAREA:** It specifies the data area that is passed to the invoked program as a communication area. The invoked program accesses the communication area via its DFHCOMMAREA field.
- **LENGTH:** It specifies a binary halfword (PIC S9(4) COMP) or numeric literal that indicates the length of data area specified in the COMMAREA option.

11.5: XCTL Command

## Example for XCTL Command

- Code in Program 'Menu':

```
EXEC CICS HANDLE CONDITION  
  PGMIDERR(4000-PGM-NOT-FND)  
  ERROR(4000-GENERAL-ERROR)  
END-EXEC.  
EXEC CICS XCTL  
  PROGRAM('FILE-MAINT')  
  COMMAREA(WS-COMM-AREA)  
  LENGTH(20)  
END-EXEC.
```

- Code in Program 'FILE-MAINT':

```
EXEC CICS  
  RETURN Takes control back to CICS  
END-EXEC
```



Copyright © Capgemini 2015. All Rights Reserved 15

11.6: XCTL versus LINK  
**A Comparison**

■ XCTL:

- Calling program is released from the main memory.
- Control is transferred to the same logical level.
- Return can be to CICS or to the application program.
- COMMAREA option can be used to pass data to the invoked program.

■ LINK:

- Calling program is not released from the main memory.
- Control is transferred to the next logical level.
- Return is always to the application program.
- COMMAREA option is not required in RETURN because the invoked program does pass pointer to the communication area of the calling program.



Copyright © Capgemini 2015. All Rights Reserved 16

11.7: Exceptional Conditions

## Some Cases

- Following are some of the exceptional conditions occurring during PCP:
  - NOTAUTH: A resource security check has failed
  - PGMDERR: The program specified is not found in PPT



Copyright © Capgemini 2015. All Rights Reserved. 17

## 11.8: Data Passing through COMMAREA

## The Procedure

- Data can be passed to a called program using the COMMAREA option of the LINK or XCTL commands in a calling program.
- In case of LINK, the called program may alter the data content of COMMAREA and the changes will be available to the calling program after the RETURN command is issued in the called program.
- This implies that the called program does not have to specify the COMMAREA option in the RETURN command.



Copyright © Capgemini 2015. All Rights Reserved 18

## 11.8: Data Passing through COMMAREA

## The Procedure

- If the COMMAREA is used in the calling program, the area must be defined in the Working Storage Section of the program, whereas in the called program, the area must be defined as the first area in the Linkage Section, using the reserved name DFHCOMMAREA.
- The length of the COMMAREA must be specified in the LENGTH parameter of the LINK or XCTL command in the calling program.



Copyright © Capgemini 2015. All Rights Reserved 19

11.8: Data Passing through COMMAREA

## The Procedure

- The LENGTH parameter must be defined as a halfword binary field (S9(04) COMP).
- The maximum length which can be specified is 65,536 bytes.



Copyright © Capgemini 2015. All Rights Reserved 20

## Summary

- In this lesson, you have learnt:
  - The methods to use Link, XCTL Commands
  - Difference between Link and XCTL Commands



## Review Questions

- Question 1: Only the program at level-2 can issue a RETURN command.
  - True / False
  
- Question 2: The \_\_\_ command is used to pass control from one application program to another application program at the same logical level.
  
- Question 3: \_\_\_ exception is raised when the program specified by the link command is not found in PPT.



# **Customer Information Control System**

Lesson 12: File Control  
Operations (Random Access)

## Lesson Objectives

- In this lesson, you will learn about:
  - How to write records to a dataset
  - How to read records from a dataset
  - How to rewrite records to a dataset
  - How to delete records from a dataset



12.1: Supported Access Methods

## Types of Methods

- CICS File Control supports the VSAM and BDAM data access method under CICS.
- There are three types of VSAM files, all of which are supported by CICS.
- These are as follows:
  - Key-Sequenced Dataset (KSDS)
  - Entry-Sequenced Dataset (ESDS)
  - Relative-Record Dataset (RRDS)

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

**Supported Access Methods:**

- The CICS File Control Program (FCP) provides application programs with services to read, update, add, and delete records in a file.
- In addition, it makes application programs independent of the structure of the database, while it manages exclusive control over the records in order to maintain the data integrity during record updates.

12.2: Commands for File Handling

## Available Commands

- Following are some of the commands used for file handling:
  - READ: To read a record directly
  - WRITE: To newly write a record
  - REWRITE: To update an existing record
  - DELETE: To delete a record
  - UNLOCK: To release exclusive control acquired for update



Copyright © Capgemini 2015. All Rights Reserved. 4

12.3: Special Services of File Control

## Some Special Services

➤ Special services of File Control Program are as follows:

- Data Independence
- Exclusive control during updates
- File Open/Close

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

**Special Services of File Control:**

- **Data Independence:** Data independence is a concept of a program being independent of the structure of database or the data access methods. The CICS File Control provides data independence to application program, so that the application programmer does not have to be concerned with such data-dependent COBOL parameters or JCL as:
  - INPUT-OUTPUT SECTION
  - SELECT Statement
  - FD Statement
  - OPEN/CLOSEThe system programmer (or application programmer) defines the File Control Table (FCT) to specify the characteristics of files to be used under CICS, while application programmer codes application program using CICS commands. In this way, it is almost transparent to the application programmer which data access method is being used.
- **Exclusive Control During Updates:** If a task is updating a record, the other tasks must be excluded from updating the record, otherwise data content will be incorrectly updated. This control is called exclusive control over the resources during updates. This is important because in the CICS environment, many tasks might be concurrently accessing the same file (possibly the same record). CICS locks the entire control interval where that record under subject is residing.
- **File Open / Close:** When an application program accesses a file, the file must be open under CICS. For this, FCT defines an initial file open/close status. If the file is closed, you must open the file using the Master Terminal Transaction (CEMT) before you initiate an application program, which uses this file.

12.4: READ Command Using Full Key

## Use of READ Command

- The READ command with the INTO option using the full key of a record is used to read a specific record by the full key.
- The data content of the record will be moved into the specified field defined in the working-storage section of the program.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

12.4: READ Command Using Full Key

## Format for READ Command Using Full Key

Let us see the format for READ command:

```
EXEC CICS READ  
  DATASET(name)  
  INTO(data-area)  
  RIDFLD(data-area)  
  [GTEQ]  
  [RRN]  
  [UPDATE]  
  [LENGTH(data-value)]  
  [KEYLENGTH(data-value)]  
  [GENERIC]  
 END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

### Format for READ Command using Full Key:

The above slide shows the format for READ command:

- **DATASET:** It names the file which has to be read. The file name must be specified in the FCT.
- **INTO:** It names the field in the working storage section to which the record content needs to be placed.
- **RIDFLD:** It specifies the key field which identifies the record to be read.
- **LENGTH:** It indicates the maximum length of the record to be read. This is optional. If it is not specified, at the completion of the command, CICS will place the actual length of the record into the LENGTH field.
- **RRN:** It must be coded if the file is a relative record VSAM file (RRDS).
- **UPDATE:** It specifies that you intend to update the record with a REWRITE or DELETE command.

12.4: READ Command Using Full Key

## Format for READ Command Using Full Key

- Let us see an example of READ Command using full key:

### WORKING-STORAGE SECTION

```
01 LEN      PIC S9(04) COMP VALUE 25.  
01 RECORD-KEY    PIC X(05) VALUE 'A0001'.  
01 FILE-AREA.  
 02 REC-KEY     PIC X(05).  
 02 REC-DESC    PIC X(20).
```



Copyright © Capgemini 2015. All Rights Reserved 8

12.4: READ Command Using Full Key

## Format for READ Command Using Full Key

```
PROCEDURE DIVISION.  
2500-READ SECTION.  
  EXEC CICS HANDLE CONDITION  
    LENGERR (2500-LENGERR)  
    NOTFND (2500-NOTFND)  
    ERROR (2500-OTHER)  
  END-EXEC.  
  EXEC CICS READ  
    INTO (FILE-AREA)  
    DATASET ('MASTER')  
    RIDFLD (RECORD-KEY)  
    LENGTH (LEN)  
  END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 9

12.4: READ Command Using Full Key

## Format for READ Command Using Full Key

- Common Exceptional conditions:
  - DUPKEY
  - NOTFND
  - LENGERR
  - NOTOPEN

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 10

### Exceptions for READ Command Using Full Key:

The following exceptions may arise for READ command using full key:

- **DUPKEY:** Duplicate record is found in the specified key (In case of Alternate Record Key). The first record in the file, which has that key, will be read. If you want to read other records of the same key, you have to use the Browse operation.
- **NOTFND:** The record with the specified key is not found.
- **LENGERR:** The specified length is shorter than the actual record length. The record will be truncated at the length specified and moved into the INTO field. The actual length will be placed in the LENGTH field.
- **NOTOPEN:** The file specified is not open.

12.5: READ Command Using GENERIC Option

## Use of READ Command with GENERIC Option

- The READ command with the GENERIC option is used to read a nonspecific record based on the generic key that is specified.
- This is useful when you do not know the complete information of the key.



Copyright © Capgemini 2015. All Rights Reserved 11

12.5: READ Command Using GENERIC Option

## Example of READ Command with GENERIC

Let us see an example of READ command with GENERIC option:

```
MOVE 35 TO WK-LEN.  
MOVE 'NY' TO REC-A-KEY.  
EXEC CICS READ  
  DATASET('FILE2')  
  INTO(FILE-OAREA)  
  RIDFLD(REC-A-KEY)  
  KEYLENGTH(2)  
  GENERIC  
  LENGTH(WK-LEN)  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 12

### READ Command Using GENERIC Option:

- Suppose that file FILE2 has records in the following order:  
BC001  
DC001  
DC001  
NY000\*  
NY001  
NY002  
PH001  
PH002
- Then, the record (NY000) will be read, because this is the first record of the generic key "NY".

12.5: READ Command Using GENERIC Option

## Exceptions of READ Command with GENERIC

- Following exceptional conditions can occur while using READ command with GENERIC option:
  - NOTFND: No record with the key specified is found.
  - LENGERR: The specified length is shorter than the actual record length. The record will be truncated at the length specified and moved into the INTO field. The actual length will be placed in the LENGTH field.
  - INVREQ: The key length specified is greater than the actual key length of the record.



Copyright © Capgemini 2015. All Rights Reserved 13

12.6: READ Command Using GTEQ Option

## Use of READ Command with GTEQ Option

- The READ command with the GTEQ option is used to read a nonspecific record whose key is equal to or greater than the full key data specified.
- This is useful when the full key is known, but you are not sure that the key exists in the file.



Copyright © Capgemini 2015. All Rights Reserved 14

12.6: READ Command Using GTEQ Option

## Example of READ Command with GTEQ

- Let us see an example of using READ command with GTEQ option:

```
MOVE 35 TO WK-LEN  
MOVE 'NY003' TO REC-KEY ↪ =must be a full key  
EXEC CICS READ  
  DATASET('FILE2')  
  INTO(FILE-IOAREA)  
  RIDFLD(REC-A-KEY)  
  GTEQ  
  LENGTH(WK-LEN)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 15

12.7: WRITE Command

## Use of WRITE Command

- The WRITE command is used to write a record directly into a file based on the specified key.

```
EXEC CICS WRITE  
  DATASET(name)  
  FROM(data-area)  
  RIDFLD(data-area)  
  [RRN]  
  [LENGTH(data-value)]  
  END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16

## Format for WRITE Command

- The WRITE command format involves the following attributes:
  - Dataset: It indicates the filename from the File control table.
  - From: It indicates record to be written.
  - RIDFLD: It indicates the field identifying the record. For VSAM keyed files (KSDS) the value is the key of the record. For relative record files (RRDS), the value is the relative record number.
  - RRN: RRN must be coded if the file is a relative record VSAM file (RRDS).



Copyright © Capgemini 2015. All Rights Reserved 17

12.7: WRITE Command

## Example for WRITE Command

- Let us see an example for WRITE command:

```
MOVE 35 TO WK-LEN.  
MOVE 'NY004' TO REC-A-KEY.  
Move symbolic map fields to the record area.....  
EXEC CICS WRITE  
    DATASET('FILE2')  
    FROM(FILE-IOAREA)  
    RIDFLD(REC-A-KEY)  
    LENGTH(WK-LEN)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 18

12.7: WRITE Command

## Exceptions for WRITE Command

- Following exceptions can occur while using WRITE command:
  - DUPREC: It indicates that the duplicate record is found.
  - NOSPACE: It indicates that no disk space is available for the record addition.
  - LENGERR: It indicates that the length specified is greater than the maximum length specified in the VSAM cluster.



Copyright © Capgemini 2015. All Rights Reserved 19

12.8: READ/UPDATE and REWRITE Commands

## Use of READ/UPDATE and REWRITE

- A combination of the READ command with the UPDATE option and the REWRITE command is used to update a record.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 20

12.8: READ/UPDATE and REWRITE Commands

## Example of READ/UPDATE and REWRITE

```
MOVE 35 TO WK-LEN. MOVE 'NY001' TO REC-A-KEY.  
EXEC CICS READ  
    DATASET('FILE2')  
    INTO(FILE-IOAREA)  
    RIDFLD(REC-A-KEY)  
    UPDATE  
    LENGTH(WK-LEN)  
    ← =read for updating  
END-EXEC.  
Set of MOVE Statements.....  
EXEC CICS  
    REWRITE  
    DATASET('FILE2')  
    FROM(FILE-IOAREA)  
    LENGTH(WK-LEN)  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 21

### READ/UPDATE and REWRITE Commands:

#### **Example:**

- On the completion of the READ/UPDATE command, record NY001 will be read and reserved for the subsequent update. That is, exclusive control over the record is maintained for this task.
- On the completion of the REWRITE command, the same record NY001 will be rewritten and the record will be released from exclusive control.

12.8: READ/UPDATE and REWRITE Commands

## Exceptions for READ/UPDATE Commands

- The following exception can occur while using READ/UPDATE commands:
  - INVREQ: The REWRITE command is issued without a prior READ command with the UPDATE option.



Copyright © Capgemini 2015. All Rights Reserved. 22

12.9: UNLOCK Command

## Use of UNLOCK Command

- The READ command with the UPDATE option normally maintains exclusive control over the record read until the following situations occur:
  - The record is updated by the REWRITE command.
  - The transaction is normally or abnormally completed.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 23

12.9: UNLOCK Command

## Use of UNLOCK Command

- Occasionally, it is found that after reading a record through the READ command with the UPDATE option, the update is no longer required or the REWRITE command itself fails the execution.
- In this case, the application program should release exclusive control from the record, so that other tasks can access the same record.
- The UNLOCK command is used to release the exclusive control from the record.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 24

**Note:**

Since any records held by the UPDATE option are released when your task is terminated anyway, the UNLOCK command is generally not used.

12.9: UNLOCK Command

## Example of UNLOCK Command

- Let us see an example of using UNLOCK command:

```
EXEC CICS UNLOCK  
    DATASET('FILE2')  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 25

12.10: DELETE Command

## Use of DELETE Command

- The DELETE command is used to delete one record or a group of records from a file.
- DELETE command can be used with two approaches:
  - DELETE after READ/UPDATE approach
  - Direct DELETE approach

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 26

12.10: DELETE Command

## The DELETE after READ/UPDATE Approach

- In the DELETE after READ/UPDATE approach, the DELETE command is used without the record key information, after the READ command with the UPDATE option is completed.
- The record which was read by the READ/UPDATE command will be deleted from the file.



Copyright © Capgemini 2015. All Rights Reserved. 27

12.10: DELETE Command

## Example of DELETE after READ/UPDATE

- Let us see an example of DELETE after READ/UPDATE:

```
MOVE 35 TO WK-LEN.  
MOVE 'NY001' TO REC-A-KEY.  
EXEC CICS READ  
      DATASET('FILE2')  
      INTO(FILE-IOAREA)  
      RIDFLD(REC-A-KEY)  
      UPDATE  
      LENGTH(WK-LEN)  
  
END-EXEC.  
.....  
EXEC CICS DELETE  
      DATASET('FILE2')  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 28

12.10: DELETE Command

## Exceptions with DELETE after READ/UPDATE

- The following exceptional condition can occur while using DELETE after READ/UPDATE approach:
  - INVREQ:
    - The DELETE command without the RIDFLD option is issued without a prior READ/UPDATE command.



Copyright © Capgemini 2015. All Rights Reserved 29

12.10: DELETE Command

## The Direct DELETE Approach

- In the Direct DELETE approach, the DELETE command is issued with the full key information in the RIDFLD field.
- The record specified will be directly deleted from the file.
- Format / Example:

```
MOVE 'NY001' TO REC-KEY.  
EXEC CICS DELETE  
      DATASET ('FILE2')  
      RIDFLD(REC-A-KEY)      ← = required  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 30

**DELETE Command:****The Direct DELETE Approach:**

- The format of the DELETE command in this approach is shown in the example shown in the above slide.
- DATASET must name the file from which a record needs to be deleted. In addition, the full key data must be provided in the RIDFLD field for the record to be deleted.

12.10: DELETE Command

## Exceptions for Direct DELETE

- The following exceptional conditions can occur while using the Direct DELETE approach:
  - DUPKEY
    - There is more than one record with same key. (In the case of Alt. Index).
  - NOTFND
    - The record specified is not found.



Copyright © Capgemini 2015. All Rights Reserved. 31

## Summary

- In this lesson, you have learnt:
  - READ command can be used to read a record from a dataset using full key or partial key.
  - WRITE command is used to write a record to a dataset.
  - DELETE command is used to delete a record from a dataset.



## Review Questions

- Question 1: The \_\_\_ command is used to release the exclusive control from the record.
- Question 2: A combination of the READ command with the \_\_\_ option and the REWRITE command is used to update a record.
- Question 3: \_\_\_ exception is raised when the REWRITE command is issued without a prior READ command with the UPDATE option.



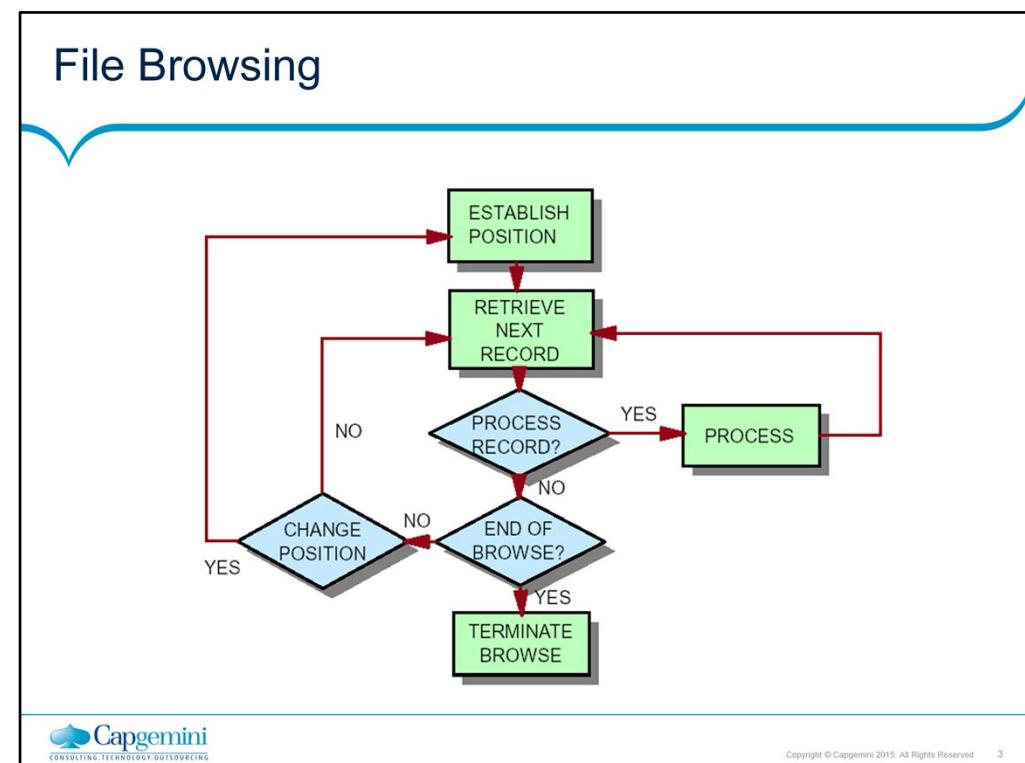
# **Customer Information Control System**

Lesson 13: File Control  
Operations (Sequential  
Access)

## Lesson Objectives

- In this lesson, you will learn:
  - The method to use STARTBR command to position at a record
  - The method to use READNEXT and READPREV commands to read a file sequentially forward and backward
  - The method to use the RESETBR command and ENDBR command





13.1: Introduction

## Sequential Access

- Under CICS any of the VSAM datasets can be accessed randomly as well as sequentially.
- Available Commands:
  - STARTBR: To establish the position of the BROWSE operation
  - READNEXT: To read a next record (Forward)
  - READPREV: To read a previous record (backward)
  - RESETBR: To reestablish another position for a new browse
  - ENDBR: To complete a browse operation

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

13.2: STARTBR Command

## Use of STARTBR Command

- The STARTBR command is used to establish a browse starting position for a file.

```
EXEC CICS STARTBR  
    DATASET(data-name| literal)  
    RIDFLD(data-name)  
    [GTEQ/EQUAL]  
    END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

**STARTBR Command:**

The format for the STARTBR command includes the following attributes:

- Dataset:** It indicates the file name from the File control Table.
- RIDFLD:** It indicates the field identifying the record at which the Browse operation will start.
- GTEQ:** It indicates that the browse operation will start at the first record, whose key is greater than or EQUAL to the value in RIDFLD.
- EQUAL:** It indicates that the browse operation will start at the record, whose key is equal to the value in the RIDFLD. If there is no such record, then the NOTFND condition is raised.
- RRN:** It needs to be coded if the file is a VSAM relative-record file (RRDS).
- RBA:** It needs to be coded if the file is a VSAM entry sequence file (ESDS).

13.2: STARTBR Command

## Example of STARTBR Command

Let us see an example of STARTBR command:

```
WORKING-STORAGE SECTION.  
 01 WK-LEN          PIC S9(4)  
  COMP.  
  01 FILE-IOAREA.  
    05 REC-A-KEY.  
      10 REC-A-KEY-CITY  PIC XX.  
      10 REC-A-KEY-SEQ   PIC 999.  
    05 REC-A-DETAIL    PIC X(30).  
  
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

### STARTING BROWSE FROM THE BEGINNING OF DATASET

In KSDS, by moving LOW-VALUES to RIDFLD of STARTBR before issuing STARTBR

If key is numeric, move ZERO not LOW-VALUES

In a ESDS, move ZEROS to RIDFLD.

In a RRDS, move 1 to RIDFLD.

### STARTING BROWSE FROM END OF DATASET

In KSDS, by moving HIGH-VALUES to RIDFLD of STARTBR before issuing STARTBR

HIGH-VALUES can not be moved to numeric RIDFLD in KSDS

In a ESDS / RRDS, move HIGH-VALUES to RIDFLD.

The STARTBR command positions the pointer to the record specified.

13.2: STARTBR Command

## Example of STARTBR Command

- Let us see one more example of STARTBR Command:

```
. PROCEDURE DIVISION.  
:  
MOVE 'NY000' TO REC-A-KEY.  
EXEC CICS STARTBR  
    DATASET ('FILE2')  
    RIDFLD(REC-A-KEY)  
    GTEQ  
END-EXEC
```

← = default



Copyright © Capgemini 2015. All Rights Reserved. 7

### STARTBR Command:

#### **Example:**

##### Execution Results:

- Suppose the file has the records in the following order:

B0001  
DC001  
DC002  
NY001  
NY002  
NY003  
PH001  
PH002

←=Then, browse starting position is set here (NY001).

13.2: STARTBR Command

## Exceptions for STARTBR Command

- Following exceptional conditions can occur while using STARTBR command:
  - DSIDERR: The file specified is not found in FCT.
  - NOTFND: The specified record is not found.



Copyright © Capgemini 2015. All Rights Reserved 8

13.3: Start Browse at the Beginning of File

## Concept of Start Browse at Beginning of File

- Move LOW-VALUES to the RIDFLD before you can issue the STARTBR command.
- The processing then begins at the first record whose key is greater than or EQUAL to hexadecimal zeros.
- If a KSDS has a numeric key, move ZERO to the key field to start a browse at the first record.



Copyright © Capgemini 2015. All Rights Reserved 9

13.4: Start Browse at a Specific Record

## Concept of Start Browse at a Specific Record

- The record to be browsed must exist in the file:

```
EXEC CICS STARTBR  
  DATASET ('CUSTMAS')  
  RIDFLD(CM-CUSTOMER-NUMBER)  
  EQUAL  
END-EXEC.
```

- If CM-CUSTOMER-NUMBER contains 10000, then processing will start with the same record.
- If there is no such record, NOTFND condition is raised.



Copyright © Capgemini 2015. All Rights Reserved 10

13.5: Start Browse at the End of File

## Concept of Start Browse at the End of File

- Move HIGH-VALUES to the RIDFLD before you can issue the STARTBR command.
- Issuing the STARTBR command when the RIDFLD contains HIGH-VALUE is a special case – it does not cause the NOTFND condition to be raised.
- Instead, it establishes the position in the file at the last record.



Copyright © Capgemini 2015. All Rights Reserved 11

13.5: Start Browse at the End of File

## Concept of Start Browse at the End of File

- You cannot move all 9s to a numeric field RIDFLD field to start processing at the last record.
- This is because the NOTFND condition will be raised if there is not a record with that key in the file.
- Since the compiler will not let you move HIGH-VALUE to a numeric field, you will have to define RIDFLD as alphanumeric to be able to move HIGH-VALUE to it.



Copyright © Capgemini 2015. All Rights Reserved 12

13.6: READNEXT Command

## Use of READNEXT Command

- The READNEXT command is used to read a record of a file sequentially forward.
- The STARTBR must have been successfully completed prior to issuing the READNEXT command.
- The data-name specified in the RIDFLD parameter of the READNEXT command must be the same as specified in the STARTBR command.



Copyright © Capgemini 2015. All Rights Reserved 13

13.6: READNEXT Command

## Format of READNEXT Command

Let us see the format of READNEXT Command:

```
EXEC CICS READNEXT  
  DATASET(data-name| literal)  
  INTO(data-name)  
  RIDFLD(data-name)  
  [LENGTH(data-value)]  
  [RBA/RRN]  
  END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 14

### READNEXT Command:

#### **Format for READNEXT Command:**

The above slide shows the format for READNEXT command. Following are the attributes used in the format:

- **Dataset:** It indicates the file name from the File control Table.
- **INTO:** It indicates the area that will contain the record being read.
- **RIDFLD:** It indicates the field identifying the record at which the Browse operation will start. Need to specify the same data name as specified in the STARTBR command. After completion of the READNEXT command, this field is updated to indicate the key, RRN or RBA of the record read.
- **RRN:** It needs to be coded if the file is a VSAM relative-record file (RRDS).
- **RBA:** It needs to be coded if the file is a VSAM entry sequence file (ESDS).

13.6: READNEXT Command

## Example of READNEXT Command

- Let us see an example of READNEXT command:

```
WORKING-STORAGE SECTION.  
01 WK-LEN  
    COMP.  
01 FILE-IOAREA.  
    05 REC-A-KEY.  
        10 REC-A-KEY-CITY  
        10 REC-A-KEY-SEQ  
    05 REC-A-DETAIL  
PROCEDURE DIVISION.  
:  
MOVE 'NY000' TO REC-A-KEY.
```

PIC S9(4)

PIC XX.

PIC 999.

PIC X(30).



Copyright © Capgemini 2015. All Rights Reserved 15

13.6: READNEXT Command

## Example of READNEXT Command

```
EXEC CICS STARTBR
  DATASET ('FILE2')
  RIDFLD(REC-A-KEY)
  GTEQ           ← = default
END-EXEC.
MOVE 35 TO WK-LEN.
EXEC CICS READNEXT DATASET('FILE2')
  INTO(FILE-IOAREA)
  RIDFLD(REC-A-KEY)           ← = required
  LENGTH(WK-LEN)
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved. 16

13.6: READNEXT Command

## Exceptions for READNEXT Command

- The following exceptional conditions can occur while using READNEXT Command:
  - DUPKEY: The key of the record is a duplicate of the next record's key.
  - ENDFILE: The end of the file is detected.
  - LENGERR: The actual record length is greater than the length specified.



Copyright © Capgemini 2015. All Rights Reserved 17

13.7: READPREV Command

## Use of READPREV Command

- READPREV command is used to read a record of a file in a backward manner.
- The STARTBR command must have been successfully completed prior to issuing the READPREV command.



Copyright © Capgemini 2015. All Rights Reserved 18

13.7: READPREV Command

## Format for READPREV Command

- Given below is the format for READPREV command:

```
EXEC CICS READPREV  
  DATASET(data-name| literal)  
  INTO(data-name)  
  RIDFLD(data-name)  
  [RBA/RRN]  
  END-EXEC
```



Copyright © Capgemini 2015. All Rights Reserved. 19

### READPREV Command:

The following attributes are used in the READPREV command:

- Dataset:** It indicates the file name from the File control Table.
- INTO:** It indicates the area that will contain the record being read.
- RIDFLD:** It indicates the field identifying the record at which the Browse operation will start. You need to specify the same data name as specified in the STARTBR command. After completion of the READPREV command, this field is updated to indicate the key, RRN or RBA of the record that is read.
- RRN:** It needs to be coded if the file is a VSAM relative - record file (RRDS).
- RBA:** It needs to be coded if the file is a VSAM entry sequence file (ESDS).

13.7: READPREV Command

## Exceptions with READPREV Command

### ■ NOTFND:

- It indicates that the record positioned by the STARTBR command or the RESETBR command is not found.
- The STARTBR or RESETBR command prior to the READPREV command must specify an existing record as the start key.
- Otherwise, at the READPREV time, the NOTFND condition will occur, in which case the browse operation must be terminated by the ENDBR command or reset by the RESETBR command.



Copyright © Capgemini 2015. All Rights Reserved 20

13.7: READPREV Command

## Peculiarities with READPREV Command

- Peculiarity 1:
  - If you issue a READPREV command following a READNEXT command, then the same record is retrieved twice.
  - For example:
    - Suppose, a file contains three records with keys 1000,1001,1002.
    - Now, if a READNEXT command is issued, it retrieves record 1001. If READPREV is issued, the same record (1001) is retrieved.
  - Similarly, if you issue a READNEXT command following a READPREV command, the same record is retrieved.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 21

13.7: READPREV Command

## Peculiarities with READPREV Command

### ■ Peculiarity 2:

- If the STARTBR command establishes the position for the file at the last record because the RIDFLD contains HIGH-VALUE, then it is safe to issue a READPREV command.
- However, if the RIDFLD for the STARTBR command doesn't contain HIGH-VALUE, it should not be followed with a READPREV command.
- READPREV command raises a NOTFND exception if the STARTBR refers to a record that is not in the file, even if the GTEQ option on the STARTBR command positions the file to the next record in the sequence.
- It is recommended not to code READPREV immediately after the STARTBR command unless the STARTBR command's RIDFLD field contains HIGH-VALUES.



Copyright © Capgemini 2015. All Rights Reserved 22

13.7: READPREV Command

## Peculiarities with READPREV Command

### ■ Peculiarity 3:

- In order to achieve the effect of a STARTBR command with a RIDFLD that doesn't contain HIGH-VALUE followed by a READPREV command, you must issue four commands:
  - **StartBR:** It positions the file to the record you specify.
  - **ReadNext:** It retrieves the record.
  - **ReadPrev:** It changes the direction of Browse.
  - **ReadPrev:** It retrieves the previous record.



Copyright © Capgemini 2015. All Rights Reserved 23

13.8: READNEXT/READPREV – NOTE  
**A Recommendation**

- It is recommended to use READNEXT after STARTBR to avoid NOTFND exceptional condition.
- Even if the specified record is not existing, READNEXT will read next record in sequence.



Copyright © Capgemini 2015. All Rights Reserved 24

13.9: Changing Direction of Browse

## Procedure for Changing Direction of Browse

- After the STARTBR command, the direction of browse can be changed from forward to backward by simply switching the READNEXT command to the READPREV command, or vice versa.
- The first READPREV (or READNEXT) command after the direction change will read the same record as the last READNEXT (or READPREV) command.



Copyright © Capgemini 2015. All Rights Reserved 25

13.9: Changing Direction of Browse

## Example of Changing Direction of Browse

STARTBR → READNEXT → READNEXT → READNEXT → READPREV → READPREV

(rec1)                    (rec2)                    (rec3)                    (rec3)                    (rec2)

↓

Change of Direction

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 26

13.10: ENDBR Command

## Use of ENDBR Command

- At the physical end of file or the logical end of file, the browser operation is terminated using the ENDBR command.
- Format / Example:

```
EXEC CICS ENDBR  
      DATASET('FILE2')  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 27

13.11: RESETBR Command

## Use of RESETBR Command

- Function:
  - RESETBR command is used to reestablish another starting point within the same browse operation against the same file.
  - Functionality-wise it is same as STARTBR.
  - This command makes it possible to reposition the dataset for a new browse operation (without issuing ENDBR command).

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 28

13.11: RESETBR Command

## Example of RESETBR Command

- Let us see an example of RESETBR command:

```
WORKING-STORAGE SECTION.  
01 WK-LEN          PIC S9(4) COMP.  
01 FILE-IOAREA.  
    05 REC-A-KEY.  
        10 REC-A-KEY-CITY   PIC XX.  
        10 REC-A-KEY-SEQ    PIC 999.  
    05 REC-A-DETAIL      PIC X(30).  
PROCEDURE DIVISION.  
    MOVE 'NY000' TO REC-A-KEY.  
    EXEC CICS STARTBR  
        DATASET ('FILE2')  
        RIDFLD(REC-A-KEY)  
        GTEQ           ← = default  
    END-EXEC
```



Copyright © Capgemini 2015. All Rights Reserved 29

13.11: RESETBR Command

## Example of RESETBR Command

```
MOVE 35 TO WK-LEN.  
EXEC CICS READNEXT DATASET('FILE2')  
    INTO(FILE-IOAREA)  
    RIDFLD(REC-A-KEY)      ← = required  
    LENGTH(WK-LEN)  
END-EXEC.  
**Go for READNEXT until key matches....  
**Reset the key  
    EXEC CICS RESETBR DATASET('FILE2')  
        RIDFLD(REC-A-KEY)  
        EQUAL  
    END-EXEC.  
:  
:
```



Copyright © Capgemini 2015. All Rights Reserved. 30

13.12: Multiple Browse Operations

## Use of Multiple Browse Operations

- The multiple browse operations are used to perform several concurrent browse operations against the same file.
- One browse operation needs to be identified by the REQID parameter in the browse command.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 31

### Multiple Browse Operations:

```
*OPERATION (1)
MOVE 'NY000' TO REC-A-KEY1.
EXEC CICS STARTBR
    DATASET('FILE2')
    REQID(1)
    RIDFLD(REC-A-KEY1) GTEQ
END-EXEC.
*
*OPERATION(2)
MOVE 'DC002' TO REC-A-KEY2.
EXEC CICS STARTBR
    DATASET('FILE2')
    REQID(2)
    RIDFLD(REC-A-KEY2) GTEQ
END-EXEC.
*
*OPERATION (3)
MOVE 'B0000' TO REC-A-KEY3.
EXEC CICS STARTBR
    DATASET('FILE2')
    REQID(3)
    RIDFLD(REC-A-KEY3) GTEQ
END-EXEC.
```

13.12: Multiple Browse Operations

## Use of Multiple Browse Operations

- For multiple browse operations, all browse commands (STARTBR, READNEXT, READPREV, RESETBR, ENDBR) must have the REQID parameter to identify to which browse operation group the command belongs.
- Once REQID is established, within the same REQID group, the browse operation can be performed in any way as you wish, independently from the other REQID group.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 32

### Multiple Browse Operations:

```
**OPERATION (1)
MOVE 35 TO WK-LEN1
EXEC CICS READNEXT DATASET('FILE2')
    REDQID(1)
    INTO(FILE-IOAREA1)
    RIDFLD(REC-A-KEY1) LENGTH(WK-LEN1)
END-EXEC.
*
* OPERATION (2)
MOVE 35 TO WK-LEN2
EXEC CICS READNEXT DATASET('FILE2')
    REDQID(2)
    INTO(FILE-IOAREA2)
    RIDFLD(REC-A-KEY2) LENGTH(WK-LEN2)
END-EXEC.
*
* OPERATION (3)
MOVE 35 TO WK-LEN3
EXEC CICS READNEXT DATASET('FILE2')
    REDQID(3)
    INTO(FILE-IOAREA3)
    RIDFLD(REC-A-KEY3) LENGTH(WK-LEN3)
END-EXEC.
```

## Summary

- In this lesson, you have learnt:
  - The method to browse from the beginning of file, end of file, or a specific record
  - The method to read records using READNEXT and READPREV command
  - The method to end a browse operation



## Review Question

- Question 1: \_\_\_ command is used to reestablish another starting point within the same browse operation against the same file.
  
- Question 2: For multiple browse operations, all browse commands (STARTBR, READNEXT, READPREV, RESETBR, ENDBR) must have the \_\_\_ parameter to identify to which browse operation group the command belongs.



# **Customer Information Control System**

Lesson 14: Temporary Storage  
Control

## Lesson Objectives

- In this lesson, you will learn:
  - The method to read and write records from TSQ
  - The method to delete records from TSQ



14.1: Temporary Storage Control

## Concept of Temporary Storage Control

- Temporary storage is a place provided by CICS that can be used by programs to temporarily store data.
  - Temporary storage is divided into temporary storage queues.
  - Each TS Queue contains one or more records called items that contain data stored by the application program.
  - Usually TS Queues contain one record, and the records are not necessarily related in the way that the data elements in the file record are related.
  - The CICS Temporary Storage Control Program (TSP) provides the application program with an ability to store and retrieve the data in a Temporary Storage Queue (TSQ).
  - Typically, TSQ is used for passing data among transactions.
  - A TSQ is a queue of stored records (data).

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

**Temporary Storage Control:**

- Many CICS programs have been able to process data outside their working storage sections.  
**Example:** Pseudo-conversational programs often need to save data between executions during a single terminal session.
- The storage area in which you place data like that is sometimes called as a **scratch pad**.
  - Communication area is one such example of scratch pad.
  - Temporary storage control provides a more sophisticated scratchpad facility than the communication area.

14.1: Temporary Storage Control

## Concept of Temporary Storage Control



Copyright © Capgemini 2015. All Rights Reserved. 4

Typical uses of TSQ :

- To maintain data integrity between pseudo-conversational executions
- To prevent double updating of a record
- In file browse operations
- Other uses may include:
  - Review mode in multiple screens
  - Printing reports.

Typical use of TDQ:

- In Automatic Task Initiation (ATI) for printing applications

14.2: Characteristics of TSQ

## Important Characteristics

- Let us discuss some important characteristics of TSQs:
  - TSQ is created and deleted dynamically by an application program without specifying in the CICS control tables, as long as data recovery is not intended.
  - The application program can use TSQ as a scratch pad memory facility for any purpose.
  - A TSQ is identified by the queue id (1 to 8 bytes), and a record within a TSQ is identified by the relative position number called item number.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

14.2: Characteristics of TSQ

## Important Characteristics

- The records in TSQ once written, remain accessible until the entire TSQ is explicitly deleted.
- The records in TSQ can be read sequentially or directly.
- The records in a TSQ can also be updated.
- TSQ may be written in the main storage or the auxiliary storage in the direct access device.
- A TSQ can be accessed by any transactions in the same CICS region.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

**Characteristics of TSQ:**

Usually, a pseudo conversational program writes one record to a TS Queue when the operator invokes the program for the first time. Thus on subsequent execution of the program during the same terminal session, the program rewrites the existing record. It does not add additional records.

14.3: TSQ in MAIN Storage

## Concept of TSQ in MAIN Storage

- TSQ can be written in main storage.
  - In this case, accessing TSQ is a fast and convenient operation.
  - TSQ in this mode are not recoverable.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

14.4: TSQ in Auxiliary Storage

## Concept of TSQ in Auxiliary Storage

- TSQ can also be written in auxiliary storage in a VSAM file (DFHTEMP) established by the system programmer.
  - It is always available to the application program, which implies that no file open/close is required.
  - TSQ in this mode is recoverable.
- In case of large quantity of data, auxiliary storage is preferred.
- If you wish to recover TSQ in this mode, you must specify so in the Temporary Storage Table (TST).

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 8

**Note:**

TSQ created in an auxiliary storage is always available to the application program, which implies that no file open/close is required.

14.5: Naming Convention for TSQ Queue Id

## Format for TSQ Queue Id

- Following is an example of QID naming convention:
  - Format: dttann
    - where:
      - d: Division id (for example: A for Accounts, B for Budget)
      - ttt: Terminal id
      - a: Application code (A, b, C,.....)
      - nn: Queue number(1, 2, .....
  - Terminal Id in QID: For a terminal dependent task (for example, pseudo-conversational task), the terminal id should be included in QID in order to ensure the uniqueness of TSQ to the task.



Copyright © Capgemini 2015. All Rights Reserved 9

**Note:**

In order to avoid confusion and to maintain data security, a strict naming convention for QID will be required in the installation.

14.6: TSQ Access

## Modes for TSQ Access

- TSQ can be accessed by any of the following transactions:
  - Same transaction:
    - From same terminal
    - From different terminal
  - Different transaction:
    - From same terminal
    - From different terminal

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 10

14.7: Item and Item Numbers

## Concept of Item and Item Numbers

- Within each queue, each record is assigned an item number.
  - The item numbers are automatically supplied by CICS.
  - The first record written to the queue is item 1, the second is item 2, and so on.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 11

14.7: Item and Item Numbers

## Concept of Item and Item Numbers

- The item number needs to be specified to retrieve or update a record.
- An application program can retrieve the records either sequentially or randomly.
  - For sequential retrieval, records are retrieved in Item number sequence.
  - For random retrieval, the Item number of the record to be retrieved has to be specified.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 12

14.8: Commands used in Temporary Storage Control

## Available Commands

- Following commands are available in Temporary Storage Control:
  - WRITEQ TS: To write or rewrite a record in a TSQ
  - READQ TS: To read a record in a TSQ
  - DELETEQ TS: To delete a TSQ



Copyright © Capgemini 2015. All Rights Reserved 13

14.9: WRITE Queue

## The Format

- Let us see the format for WRITE Queue:

```
EXEC CICS WRITEQ TS
    QUEUE(data-name|literal)
    FROM(data-name)
    LENGTH(data-name|literal)
    [ITEM(data-name)]
    [REWRITE]
    [{MAIN/AUXILIARY}]
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 14

### Format for WRITE Queue:

The above slide shows the format for WRITE queue. The following attributes are used in this format:

- **QUEUE:** It indicates one to eight character name of the temporary storage queue to which the data is written.
- **FROM:** It indicates the data area that contains the record to be written.
- **LENGTH:** It indicates the length of the from area. It has to be numeric. If it is a data-name it has to be defined as PIC S9(04) COMP.
- **ITEM:** It indicates the item number of the record to be updated. It must be a binary half-word which is defined as PIC S9(04) COMP.
- **MAIN:** It indicates the temporary queue will reside in the main storage.
- **AUXILIARY:** It indicates the temporary queue will reside on the disk in the temporary storage file (DFHTEMP).

14.9: WRITE Queue

## Example of WRITEQ TS Command

- The WRITE Queue command is used to write a record (item) in a TSQ:

```
EXEC CICS WRITEQ TS  
QUEUE('AttttM01')  
FROM(TSQ-DATA)  
LENGTH(TSQ-LEN)  
ITEM(TSQ-ITEM)  
MAIN  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 15

14.9: WRITE Queue

## Execution Results for WRITEQ TS Command

- If TSQ with this QID does not exist, then a TSQ will be created with QID=AttttM01, where tttt is the terminal id.
- CICS will write a record in the TSQ identified by the QID.
- CICS will place the actual relative record number (that is, item number) of the TSQ into TSQ –ITEM.
- You may have to remember this for later use if more than one record is in the TSQ.
- If TSQ with this QID already exists, then CICS will simply write a record in the existing TSQ at the next to the last existing record, and place the relative number of the record into TSQ-ITEM.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16

14.9: WRITE Queue

## Exception for WRITEQ TS Command

- The following exception can occur while using WRITEQ TS command:
  - NOSPACE: Sufficient space is not available. The default action is to suspend the task until space becomes available.



Copyright © Capgemini 2015. All Rights Reserved. 17

14.10: WRITEQ TS Command with REWRITE Option

## Use of WRITEQ TS Command with REWRITE

- The WRITEQ TS command with REWRITE option is used to rewrite a record of a TSQ that has been read.
- The READQ command does not hold exclusive control over TSQ.
- Therefore, if a TSQ is shared by other transactions, it is the application program's responsibility to establish exclusive control over the TSQ during update.
- In this case, you must use:
  - ENQ command before the READQ command, and
  - DEQ command after the WRITEQ command with the REWRITE option



Copyright © Capgemini 2015. All Rights Reserved 18

14.10: WRITEQ TS Command with REWRITE Option

## Example of WRITEQ TS with REWRITE

```
EXEC CICS READQ TS
  QUEUE(TSQ-QID)
  INTO(TSQ-DATA)
  LENGTH(TSQ-LEN)
  ITEM(TSQ-ITEM)
END-EXEC.
:
*Process data in TSQ-DATA
:
EXEC CICS WRITEQ TS
  QUEUE(TSQ-QID)
  FROM(TSQ-DATA)
  LENGTH(TSQ-LEN)
  ITEM(TSQ-ITEM)
  REWRITE
  MAIN
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 19

14.11: READ Queue

## The Format

- Following is the format for READ Queue:

```
EXEC CICS READQ TS
    QUEUE(data-name|literal)
    FROM(data-name)
    LENGTH(data-name)
    [{ITEM(data-name|literal)/NEXT}]
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 20

### READ Queue:

The above slide shows the format for READ queue:

- **QUEUE:** It indicates one to eight character name of the temporary storage queue to which the data is written.
- **INTO:** It indicates the data area that will contain the record.
- **LENGTH:** It indicates the length of the INTO area. It has to be defined as PIC S9(04) COMP.
- **ITEM:** It indicates the item number of the record to be read. It must be a binary half-word which is defined as PIC S9(04) COMP.
- **NEXT:** It indicates that the next record in the sequence should be read.

14.11: READ Queue

## Direct Read for READQ TS Command

- Direct read for READQ TS command is used to read a particular record (item) of a particular TSQ.
- Format/Example:

```
EXEC CICS READQ TS  
  QUEUE(TSQ-QID)  
  INTO(TSQ-DATA)  
  LENGTH(TSQ-LEN)  
  ITEM(TSQ-ITEM)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved. 21

14.11: READ Queue

## Direct Read for READQ TS Command

- Commonly used options during direct read for READQ TS Command are as follows:
  - NUMITEMS: If you wish to know the number of items in the TSQ, specify NUMITEMS (data-area).
    - The data-area is defined as PIC S9(4) COMP.
    - At the completion of the command, CICS will place the actual number of records in the TSQ that is specified.
  - NEXT: If the NEXT option is specified, then the task will read the next sequential record in the TSQ that is specified.
    - This option is mutually exclusive to ITEM option.



Copyright © Capgemini 2015. All Rights Reserved 22

14.11: READ Queue

## Exceptions for Direct Read for READQ TS

- Following exceptions can occur during direct read for READQ TS command:
  - QIDERR: It is raised if the Queue does not exist.
  - ITEMERR: It is raised if the item number is not present within the queue.
    - ITEMERR is never raised if you are updating record number 1.
    - This is because if the item does not exist, the queue does not exist. Hence the QIDERR is raised.



Copyright © Capgemini 2015. All Rights Reserved 23

14.11: READ Queue

## Sequential Read for READQ TS Command

- The READQ TS command can be also used for sequential reading of all records in the queue.
- This reading should be performed based on the programming techniques, instead of NEXT option of the READQ TS command.



Copyright © Capgemini 2015. All Rights Reserved 24

14.11: READ Queue

## Sequential Read for READQ TS Command

```
*ESTABLISH HANDLE CONDITION.  
  EXEC CICS HANDLE CONDITION  
    ITEMERR(TSQ-EOF)  
    ERROR(SQ-ERROR)  
  END-EXEC.  
LOOP.  
  ADD 1 TO TSQ-ITEM.  
  MOVE 200 TO TSQ-LEN.  
*READ A QUEUE.  
  EXEC CICS READQ TS QUEUE(TSQ-QID)  
    INTO(TSQ-DATA)  
    LENGTH(TSQ-LENGTH)  
    ITEM(TSQ-ITEM)  
  END-EXEC.  
:  
  (PROCESS A RECORD)  
:  
  GO TO LOOP.  
TSQ-EOF.  
  (EOF PROCESSING)  
:
```



Copyright © Capgemini 2015. All Rights Reserved. 25

14.11: READ Queue

## Sequential Read for READQ TS Command

■ Note:

- The sequential read of TSQ should be done by providing ITEM.
- This is a better approach than specifying the NEXT option in the READQ TS command.
- If NEXT option is specified, and if other tasks have read several records of the same TSQ after the last read by this task, then this task will read a skipped record.
- Therefore NEXT option should be used very carefully.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 26

14.12: DELETEQ TS Command

## Use of DELETEQ TS Command

- Function:
  - The DELETEQ TS command is used to entirely delete a TSQ.
  - A record of a queue cannot be deleted.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 27

14.12: DELETEQ TS Command

## Format for DELETEQ TS Command

- Format/Example:

```
EXEC CICS DELETEQ TS  
QUEUE(TSQ-ID)  
END-EXEC.
```

- Exceptional Condition:

- QIDERR: The queue id specified is not found.



## Summary

- In this lesson, you have learnt:
  - The method to add records in a TSQ
  - The method to modify records in a TSQ
  - The method to delete records from TSQ



## Review Questions

- Question 1: By default a TSQ is written in main storage.
  - True / False
  
- Question 2: For random retrieval, records are retrieved in Item number sequence.
  - True / False
  
- Question 3: TSQ can also be written in auxiliary storage in a VSAM file \_\_\_\_.



# **Customer Information Control System**

Lesson 15: Transient Data  
Control

## Lesson Objectives

- In this lesson, you will learn about:
  - What is a TDQ?
  - Intra-partition and Extra-partition TDQ
  - Command to write, read, and delete from TDQ
  - Automatic Task Initiation
  - Indirect Destination



15.1: Introduction

## Introduction to Transient Data Control

- The Transient Data Control module provides a convenient way to use simple sequential files.
- Each file is called Transient Data Queue or just TD Queue.
- Records in a TD Queue may be read or written but not updated or deleted.
  - The only way to delete a record is to delete an entire queue.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

**Introduction:**

- The transient data is sometimes called **Transient Data Queue (TDQ)**, while at other times it is called **Transient Data Destination**.
  - The word “Queue” is used because the records in the Transient data are put together in a sequential mode.
  - The word “Destination” is used because this sequential data is directed to other transactions.

Both terms are used interchangeably and they essentially mean the same.

15.1: Introduction

## Introduction to Transient Data Control

- The Transient Data Control provides a way to do simple sequential processing.
- It allows to store data sequentially and retrieve it later in the same sequence in which it was stored.
- TDQ is used for programs that produce output on 3270 printers.
- The advantage of using a TDQ for printing is that there is no need to worry about the complex formatting requirements of 3270 printers.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

15.1: Introduction

## Introduction to Transient Data Control

- The CICS Transient Data Control Program (TDP) allows a CICS transaction to deal with sequential data called Transient data files.
- They are also termed as Transient Data Queues (TDQs) or Transient Data Destinations.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 5

15.1: Introduction

## Introduction to Transient Data Control

- TDQs are identified by 1 to 4 characters queue-id.
- All TDQs must be registered in the Destination Control Table (DCT).
  - The primary function of the Destination Control Table (DCT) is to register control information of all TDQs.
  - The CICS Destination Control program (DCP) uses this table for identifying all TDQs and performing input/output operations against them.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 6

15.2: Types of TDQ

## Different Types of TDQs

- There are two types of TDQs:
  - Intra-partition TDQ
  - Extra-partition TDQ

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

15.2: Types of TDQ

## Concept of Intra-partition TDQ

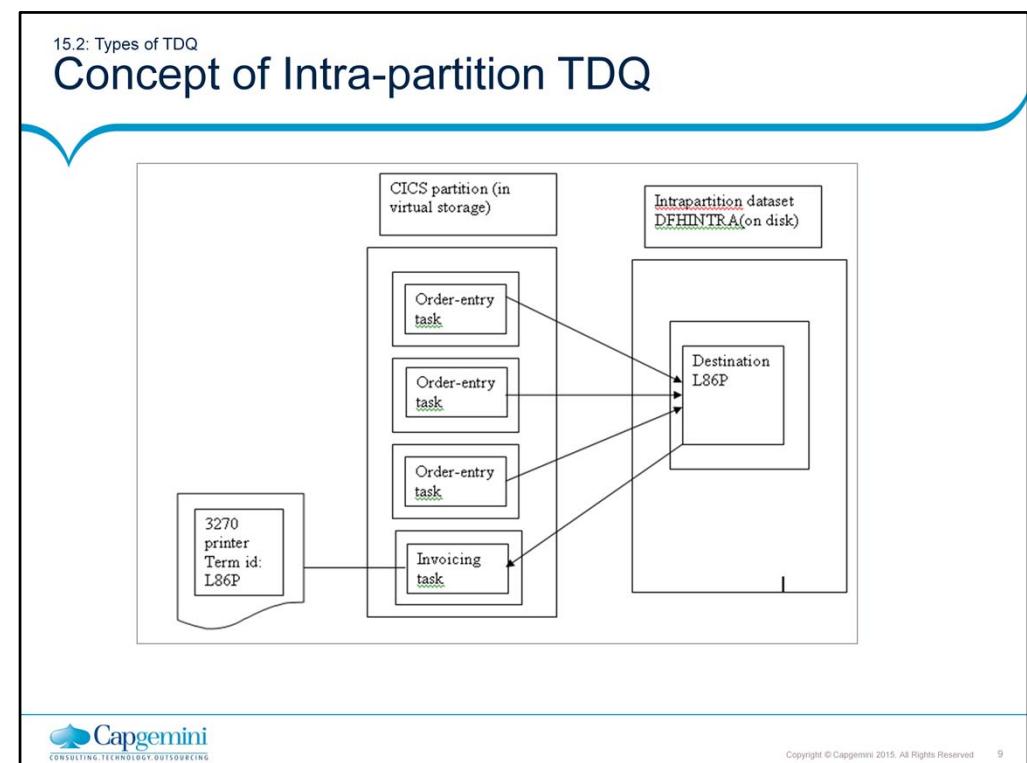
- An Intra-partition TDQ is a group of sequential records which are produced and processed by the same and/or different transactions within a CICS region.
- These TDQs are stored in only one physical file in a CICS region, which is prepared by the system programmer.
- Once a record is read from a queue, the record will be logically removed from the queue.
- An Intra-partition Queue can be assigned only to a disk device.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 8

**Intra-partition TDQ:**

- CICS stores the **Intra-partition TDQ** in a VSAM file called **DFHINTRA**.
- Regardless of how many intra-partition transient data queues are in use at one time, they are all stored in DFHINTRA.

**Note:**

An order-entry program writes records to an intra-partition destination for subsequent access by an invoicing program.

15.2: Types of TDQ

## Concept of Intra-partition TDQ

- The Intra-partition TDQ is used for various applications such as:
  - Interface among CICS transactions
  - Automatic Transaction Initiation (ATI)

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 10

15.2: Types of TDQ

## Concept of Extra-partition TDQ

- An Extra-partition TDQ is a group of sequential records, which interface between the transactions of the CICS region and the system outside of the CICS region.
- The Extra-partition TDQ is typically used for the following two applications:
  - Interface to batch jobs
  - Interface from batch jobs

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 11

The Extrapartition TDQ is typically used for the following two applications:

- Interface to batch (or TSO, or PC) jobs.
- CICS Appl. Prog. → TDQ → File → Batch Prog.
- Interface from batch (or TSQ, or PC) jobs.
- Batch prog. → File → TDQ → CICS Appl. Prog.

15.2: Types of TDQ

## Concept of Extra-partition TDQ

- Extra-partition destinations are used to collect data that is entered online but processed later by a batch program.
  - Example: An extra-partition may write order records to an extra-partition destination on a disk or tape.
- On a nightly basis, orders may be processed by a batch program.
- Each Extra-partition TDQ is a separate physical file, and it may be on the disk, tape, printer, or plotter.



Copyright © Capgemini 2015. All Rights Reserved 12

15.3: WRITEQ TD

## The Format

- Following is the format for WRITEQ TD:

```
EXEC CICS  
    WRITEQ TD QUEUE(data-name|literal)  
        FROM(data-name)  
        LENGTH(data-name|literal)  
    END-EXEC
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 13

### WRITEQ TD:

The slide shows the format for WRITEQ TD. Following attributes are used in the format:

- **Queue:** It indicates one to eight character name of the transient data queue to which the data is written.
- **FROM:** It indicates the data area that contains the record to be written.
- **LENGTH:** It indicates the length of the FROM area. It must be numeric. If a data name is used it must be PIC S9(4) COMP.

## 15.3: WRITEQ TD Example of WRITEQ TD

- Let us see an example for WRITEQ TD:

```
EXEC CICS  
  WRITEQ TD QUEUE('L86P')  
    FROM(PRINT-AREA)  
    LENGTH(133)  
  END-EXEC
```

- Here the 133 bytes of data contained in the field named PRINT-AREA are written to a TDQ named L86P.



Copyright © Capgemini 2015. All Rights Reserved 14

15.3: WRITEQ TD

## Exceptions for WRITEQ TD Command

- Following exceptions can occur while using WRITEQ TD:
  - QIDERR: The destination id specified cannot be found in DCT.
  - LENGERR: The length specified is greater than the maximum record length specified in DCT.
  - NOSPACE: No space is available in the TDQ.



Copyright © Capgemini 2015. All Rights Reserved 15

15.4. READQ

## The Format

- Following is the format for READQ:

```
EXEC CICS
    READEQ TD QUEUE(data-name|literal)
        INTO(data-name)
        LENGTH(data-name|literal)
    END-EXEC
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 16

### READQ:

The above slide shows the format for READQ. The following attributes are used in the format:

- Queue:** It indicates one to eight character name of the transient data queue to which the data is written.
- INTO:** It indicates the data area which will contain the record to be written.
- LENGTH:** It indicates the length of the INTO area. It must be numeric. If a data name is used, it must be PIC S9(4) COMP. The initial value of the length field indicates the length of largest record that the program will accept. After the READQ TD command completes, the length field is updated to indicate the actual length of the record that was read. If the program reads a record that is longer than the maximum length specified, then the LENGERR condition is raised. When the LENGERR condition occurs, as much of the input data as will fit, is placed in the INTO field. The rest of the data is discarded. Hence the program can process some of the data, provided LENGERR condition is given with a HANDLE CONDITION.

15.4. READQ

## Example for READQ TD Command

- READQ TD command is used to read a particular record of a particular TDQ.
- Format / Example:

```
EXEC CICS READQ TD  
QUEUE('MSG$')  
INTO(MSG-AREA)  
LENGTH(MSG-LEN)  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 17

15.4. READQ

## Exceptions for READQ TD Command

- The following exceptions can occur while using READQ TD:
  - QIDERR: The specified Destination Id is not found.
  - LENGERR: The length specified is shorter than the actual record length. The record will be truncated at the length specified, while that actual length will be placed in the LENGTH field.
  - QZERO: The queue is empty. When you issue a READQ TD command for a Queue that has no records, the QZERO command is raised.



Copyright © Capgemini 2015. All Rights Reserved 18

15.5: DELETEQ TD Command

## Concept of DELETEQ TD Command

- DELETEQ TD command is used to delete an Intra-partition TDQ entirely.
- Format/Example:

```
EXEC CICS DELETEQ TD  
QUEUE('MSG$')  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 19

15.6: Intra-partition TDQ

## The Format

- Following is the format for Intra-partition TDQ:

```
DFHDCT TYPE=INTRA,  
        DESTID=name,  
        [TRANSID=name,],  
        [TRIGLEV=number,],  
        [REUSE=YES|NO]
```
- TRANSID=name: Applicable only for Intra-partition.
- TRIGLEV=number: Applicable only for Intra-partition.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 20

### Intra-partition TDQ:

In the format shown in the above slide:

- **TYPE=INTRA** indicates that TDQ is the Intra-partition TDQ.
- **DESTID** defines the destination id (1 to 4 characters).
- **TRANSID** names the transaction id of the transaction to be initiated.
- **TRIGLEV** indicates the number of the records in TDQ, which triggers the transaction to be initiated.
- **REUSE** option defines space availability after a TDQ record has been read. This is useful for better disk space management. Once a record of Intra-partition TDQ is read by a transaction, the record is logically removed, but it still occupies the space.
  - If REUSE=YES is specified, then this space for the logically deleted record will be used for other TDQ records. However, in this case, records are not recoverable.
  - If REUSE=NO is specified, then the logically deleted records are recoverable.

15.7: Automatic Task Initiation (ATI)

## Concept of ATI

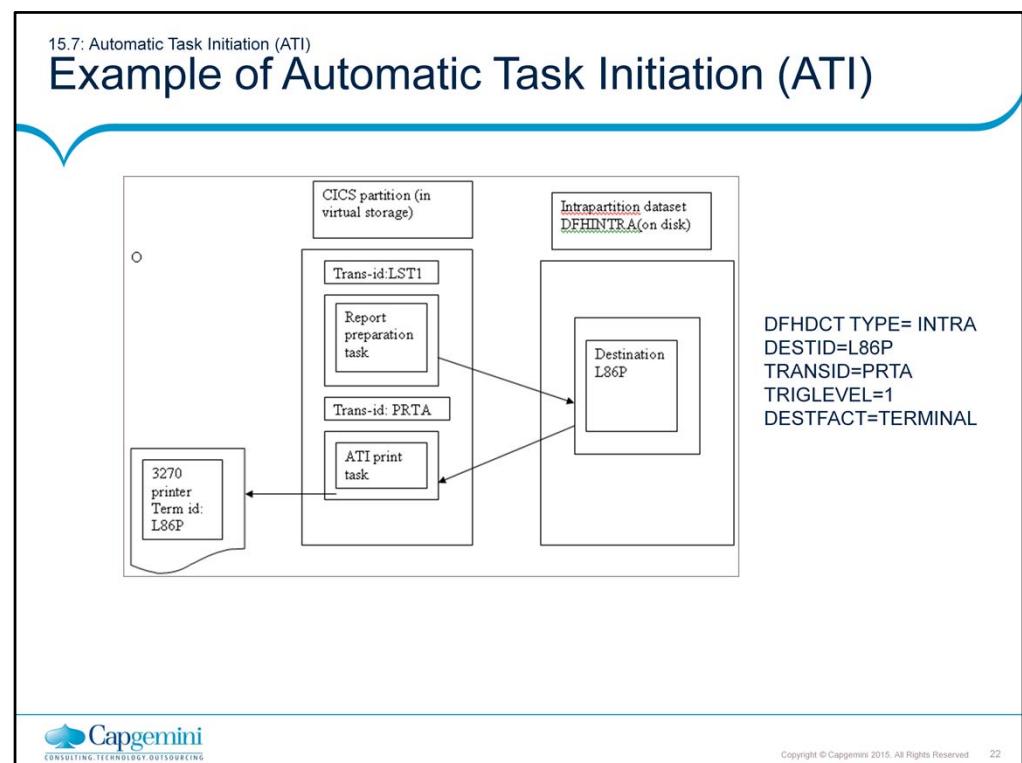
- Automatic Task Initiation (ATI) is a facility through which a CICS transaction can be initiated automatically.
- The number of the records in an Intra-partition TDQ triggers the transaction initiation.
- The transaction id (task) must be defined in the DCT entry of the Intrapartition TDQ, with nonzero trigger level.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 21

**Automatic Task Initiation (ATI):**

- ATI provides a convenient way to start a task automatically.
- To use ATI transaction identifier, a transaction identifier and trigger level has to be attached to a TDQ by making appropriate entries in the DCT. When the number of records in the queue reaches the trigger level, the specified transaction is automatically started.



15.7: Automatic Task Initiation (ATI)

## Example of Automatic Task Initiation (ATI)

- Since the entry specifies a trigger level of 1, a special print program identified by trans-id PRTA is automatically started when a record is written to the destination.
- The print task reads records from the destination, formats the print data for 3270 printer, and sends the formatted data to its attached terminal: a 3270 printer whose terminal id is L86P.

```
DFHDCT TYPE = INTRA,  
DESTID=L86P,  
TRANSID=PRTA,  
TRIGLEVEL=1,  
DESTFAC=TERMINAL
```

- Here a user-initiated reporting program prepares a report by writing records to an intra-partition destination named L86P.



Copyright © Capgemini 2015. All Rights Reserved 23

### Automatic Task Initiation (ATI):

In this case, the queue's destination id is the same as the terminal-id. Hence we can associate the ATI task for the queue with terminal L86P by coding DESFAC=TERMINAL in the queue's DCT entry.

15.7: Automatic Task Initiation (ATI)

## Example of Automatic Task Initiation (ATI)

- In this case, the Queue's destination ID is the same as terminal-id.
- DESTFAC=TERMINAL associates TDQ of DESTID L86P with printer whose TERMINALID is L86P.
- The output will be sent to a printer whose TERMINALID is L86P.



Copyright © Capgemini 2015. All Rights Reserved 24

15.8: Indirect Destination

## Concept of Indirect Destination

- An Indirect Destination lets a single transient data queue be identified by more than one destination-id.
- The DCT entry for an indirect destination simply specifies the name of a destination defined elsewhere in the DCT.
- One common reason for using an indirect destination is to shelter application programs from actual destination IDs.
- By using an indirect destination, you can change a destination ID without having to change and recompile every application program that refers to the destination.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved 25

15.8: Indirect Destination

## Concept of Indirect Destination

- In case the Printer name has to be changed, then only the INNDEST will have to be changed and no change will be required to be made in the program

DFHDCT TYPE = INTRA,  
DESTID=L86P,  
TRANSID=PRTA,  
TRIGLEVEL=1,  
DESTFAC=TERMINAL

DFHDCT TYPE =  
INDIRECT,  
DESTID=PRT1,  
INDDEST=L86P

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 26

## Summary

- In this lesson, you have learnt:
  - The difference between Intra-partition TDQ and Extra-partition TDQ
  - Concept of Automatic Task Initiation
  - Concept of Indirect Destination



Summary



Copyright © Capgemini 2015. All Rights Reserved. 27

## Review Questions

- Question 1: An Extra-partition TDQ can be used outside the CICS region.
  - True / False
  
- Question 2: CICS stores the Intra-partition TDQ in a VSAM file called \_\_\_\_.
  
- Question 3: When READQ TD command is issued for a Queue that has no records, the QZERO exception is raised.
  - True / False



# **Customer Information Control System**

Lesson 16: Tests and  
Debugging

## Lesson Objectives

- In this lesson, you will learn about:
  - ABEND EXIT
  - HANDLE ABEND Command
  - ABEND Command
  - DUMP Command



17.1: Commands for Tests and Debugging

## List of Commands

- Following is the list of commands used for testing and debugging CICS programs:
  - HANDLE ABEND Command
  - ABEND Command
  - DUMP Command

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 3

**Commands for Tests and Debugging:**

- For an effective debugging and/or better management of abnormal termination of a program, CICS provides useful control functions over these areas.
  - The Abnormal Termination Recovery function (ABEND Control) manages an abnormal termination (ABEND) of a task.
  - The Dump Control Program (DCP) manages dumping the main storage areas.
  - The CICS Trace Control Program makes use of the Trace Table for debugging aid purposes.

17.2: ABEND EXIT

## Use of ABEND EXIT

- An ABEND EXIT is a piece of code that supplements the standard abend processing that CICS provides.
- It can be a paragraph or section within the program, but is usually a separate program.
- If an ABEND EXIT is active, then it is automatically invoked when an abend occurs.
- To activate an ABEND EXIT, you issue an HANDLE ABEND command.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 4

**ABEND EXIT:**

- An ABEND EXIT may try to correct the problem that caused the abend.
- Usually an ABEND EXIT just records the information about the task that is abending.
- When it is finished, ABEND EXIT ends by issuing an ABEND command, so CICS completes the abend processing and produces a transaction dump.

17.3: HANDLE ABEND Command

## Use of HANDLE ABEND Command

- HANDLE ABEND command is used to intercept an abnormal termination within a program, and to activate, cancel, or reactivate an exit for the ABEND processing.

- Format:

```
EXEC CICS HANDLE ABEND  
[PROGRAM(name) | LABEL(label) | CANCEL | RESET]  
END-EXEC.
```



Copyright © Capgemini 2015. All Rights Reserved 5

17.3: HANDLE ABEND Command

## Use of HANDLE ABEND Command

- PROGRAM or LABEL is used to activate an exit to a program or a paragraph respectively, for the ABEND processing.
- CANCEL is used to cancel the previously established HANDLE ABEND request.
- RESET is to reactivate the previously cancelled HANDLE ABEND request.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 6

17.3: HANDLE ABEND Command

## Use of HANDLE ABEND Command

- HANDLE ABEND is generally coded at the beginning of the program.
  - So, if an abend occurs anywhere in the program, then the ABEND EXIT is invoked.
- If you code LABEL option rather than the PROGRAM option on a HANDLE ABEND command, then the paragraph or section name you specify becomes the ABEND EXIT.
  - When an abend occurs, CICS branches to that paragraph or section.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 7

**HANDLE ABEND Command:**

- The difference between the LABEL and the PROGRAM options is that LABEL option establishes an ABEND EXIT as a paragraph or section in the current program. In contrast the PROGRAM option specifies a separate program as the ABEND EXIT.
- It is easier to provide a standard abend processing in a separate program.
- Hence it is better to code PROGRAM option rather than LABEL option.

17.3: HANDLE ABEND Command

## Use of HANDLE ABEND Command

- Let us see an example of using HANDLE ABEND command:

```
EXEC CICS  
    HANDLE ABEND PROGRAM('ABEND1')  
END-EXEC.
```

- The program named ABEND1 will be invoked automatically if an abend occurs.  
The program named 'ABEND1' should exist in the PPT.



Copyright © Capgemini 2015. All Rights Reserved 8

17.4: ABEND Command

## Use of ABEND Command

- ABEND command is used to intentionally terminate a task, thus causing an ABEND.

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 9

**ABEND Command:**

- The ABCODE option causes CICS to produce a **transaction dump**. The value you supply appears in the transaction dump and in the message that is written to the terminal operator. The transaction dump is the only way to determine what caused the program to abend. Hence it is always recommended to specify ABCODE option on the ABEND command.
- There may be advantage of terminating a program using the ABEND command rather than using a RETURN command.
  - First, the ABEND command gives a **transaction dump**.
  - Second, the ABEND command invokes the CICS **dynamic transaction back out** facility to reverse any changes your task made to protected resources.

When your program develops a serious error, you normally want both **transaction dump** and **dynamic transaction back out**.

17.4: ABEND Command

## Format for ABEND Command

- Following is the format for ABEND command:

```
EXEC CICS ABEND  
[ABCODE(name)]  
END-EXEC
```

- ABCODE is used to specify the user abend code (1 to 4 characters).



Copyright © Capgemini 2015. All Rights Reserved 10

17.4: ABEND Command

## Example of ABEND Command

- Following is an example of ABEND command:

```
EXEC CICS HANDLE CONDITION  
    ERROR(ERROR-RTN)  
END-EXEC.  
:  
:  
ERROR-RTN.  
EXEC CICS ABEND  
    ABCODE('1234')  
END-EXEC.
```

 Capgemini  
CONSULTING TECHNOLOGY OUTSOURCING

Copyright © Capgemini 2015. All Rights Reserved. 11

**Example of ABEND Command:****Execution Results:**

If the general error condition occurs, then control will be passed to ERROR-RTN through the HANDLE CONDITION command. At the completion of the ABEND command, the task will be forcefully terminated with the user ABEND code 1234.

## Summary

- In this lesson, you have learnt:
  - A method to intentionally terminate a task
  - A method to use the HANDLE ABEND command



## Review Question

- Question 1: HANDLE ABEND is generally coded at the end of the program.
  - True / False
  
- Question 2: The \_\_\_ command is used to activate an ABEND EXIT.



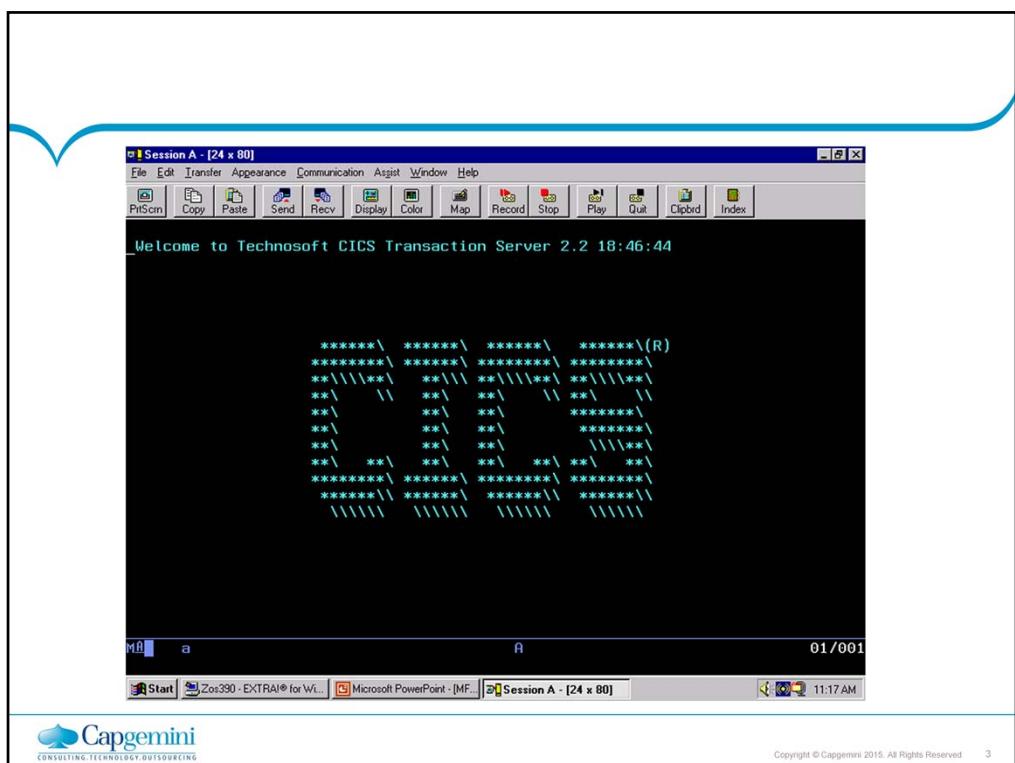
# **Customer Information Control System**

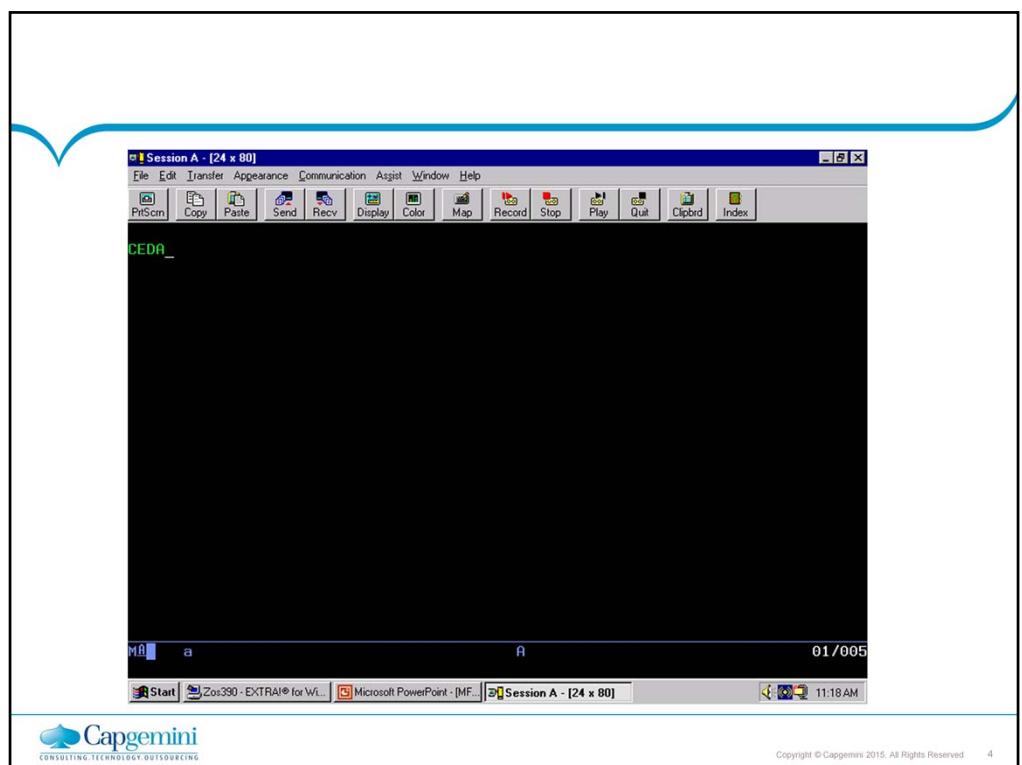
Lesson 17: CICS with DB2

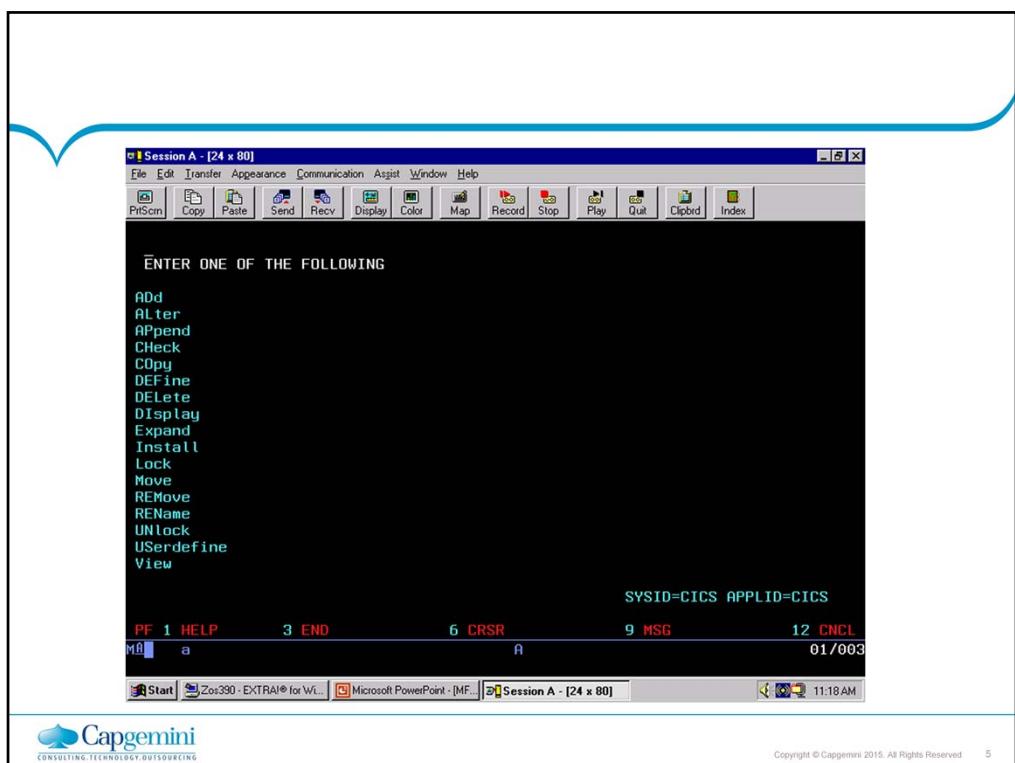
## Lesson Objectives

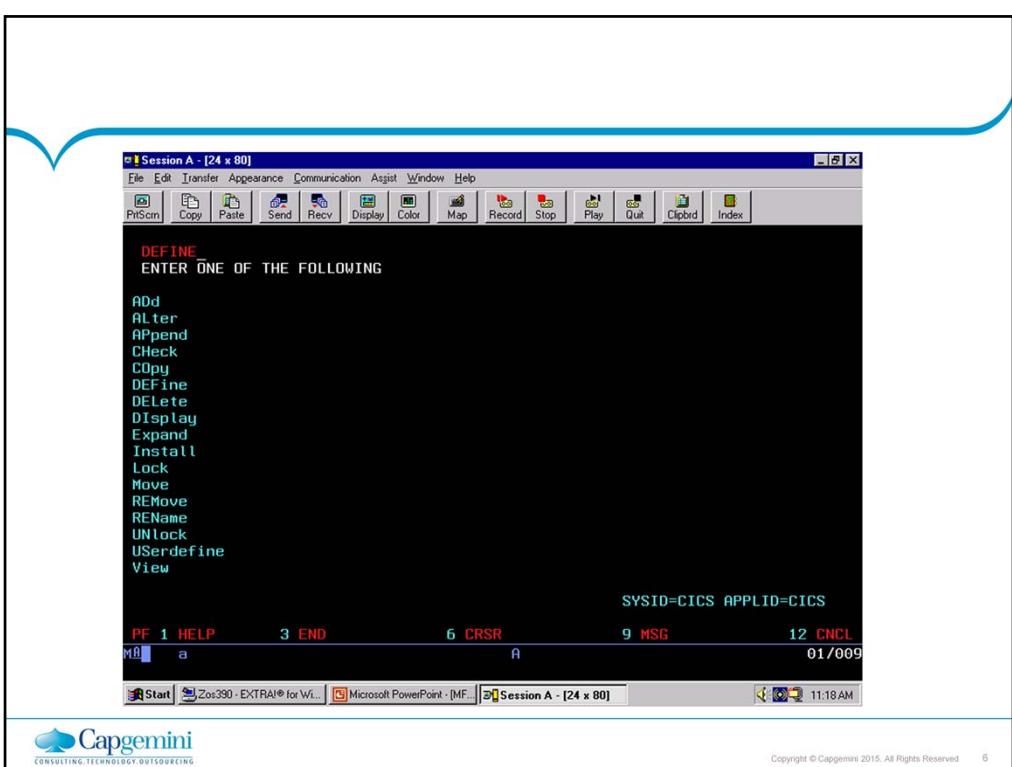
- In this lesson, you will learn about:
  - CICS-DB2 interface

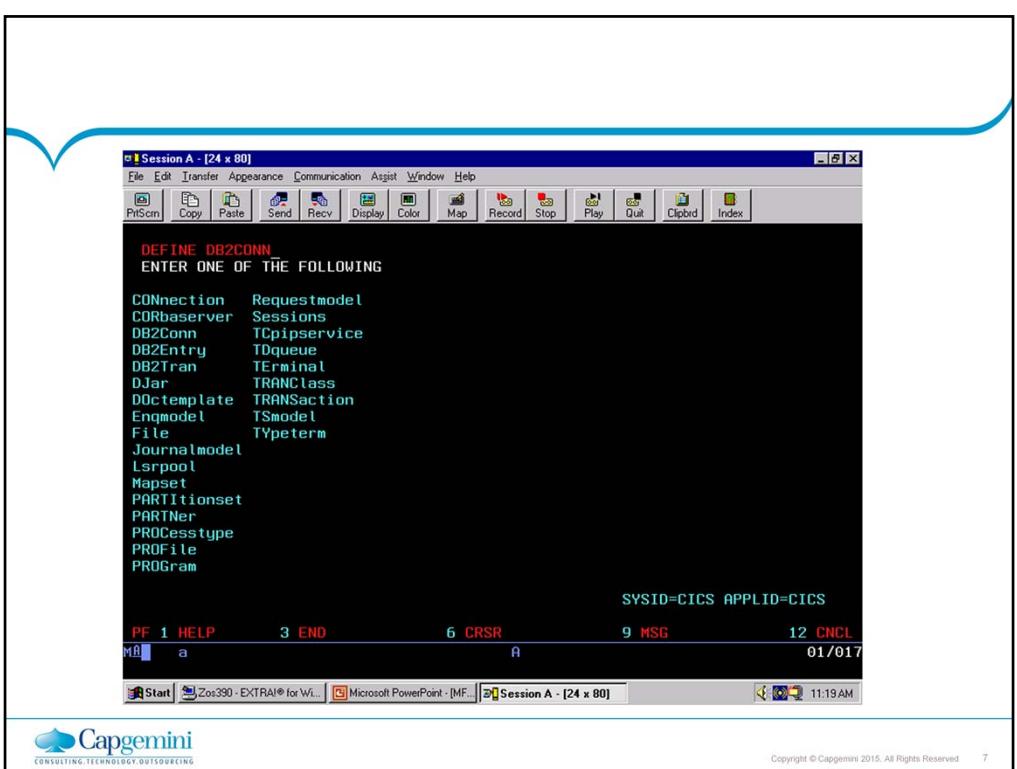






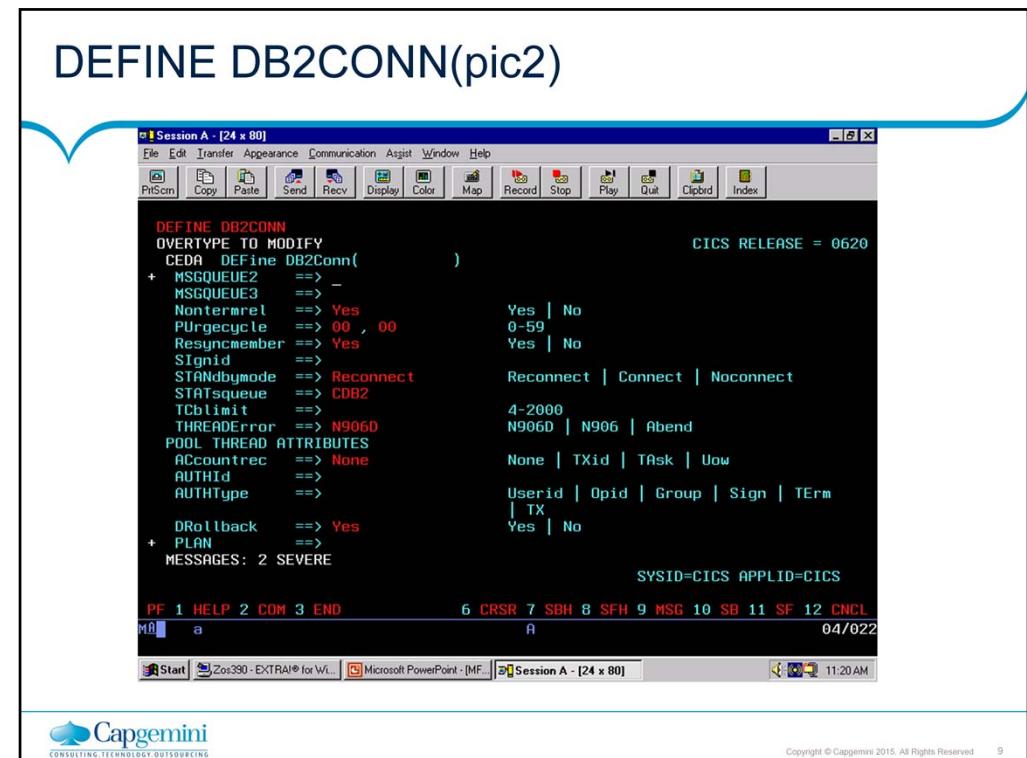






DEFINE DB2CONN(pic1)

```
Session A - [24 x 80]
File Edit Transfer Appearance Communication Assist Window Help
PrScrn Copy Paste Send Recv Display Color Map Record Stop Play Clipbrd Index
DEFINDB2CONN
OVERTYPE TO MODIFY
CEDA DEFine DB2Conn(          )
DB2Conn ==> -
Group ==> -
Description ==>
CONNECTION ATTRIBUTES
CONnecterror ==> Sqlcode      Sqlcode | Abend
DB2Groupid ==> -
DB2Id ==> -
MSGQUEUE1 ==> CDB2
MSGQUEUE2 ==> -
MSGQUEUE3 ==> -
Nontermrel ==> Yes           Yes | No
PURgecycle ==> 00 , 00        0-59
Resyncmember ==> Yes          Yes | No
Signid ==> -
STANdbymode ==> Reconnect   Reconnect | Connect | Noconnect
STATsqeue ==> CDB2
+ TCBlimit ==>                4-2000
MESSAGES: 2 SEVERE
SYSID=CICS APPLID=CICS
PF 1 HELP 2 COM 3 END       6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
MB a                         A
04/022
Start Zos390 - EXTRAI® for Wi... Microsoft PowerPoint - [MF... Session A - [24 x 80] 11:19 AM
Copyright © Capgemini 2015. All Rights Reserved. 8
```



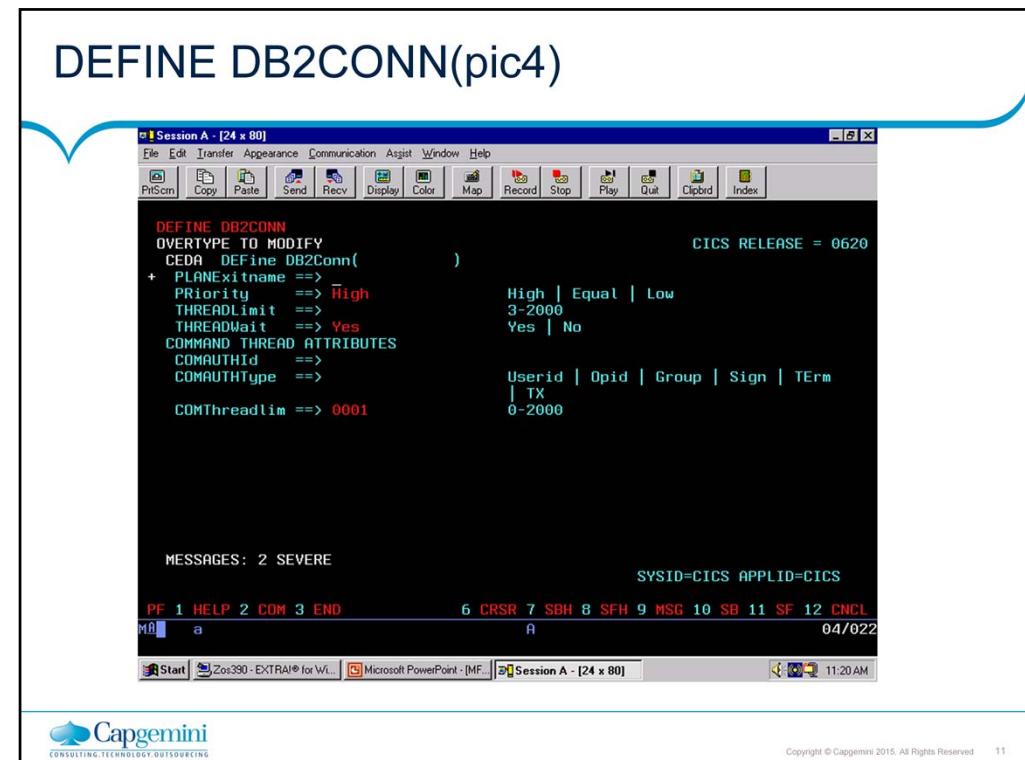
## DEFINE DB2CONN(pic3)

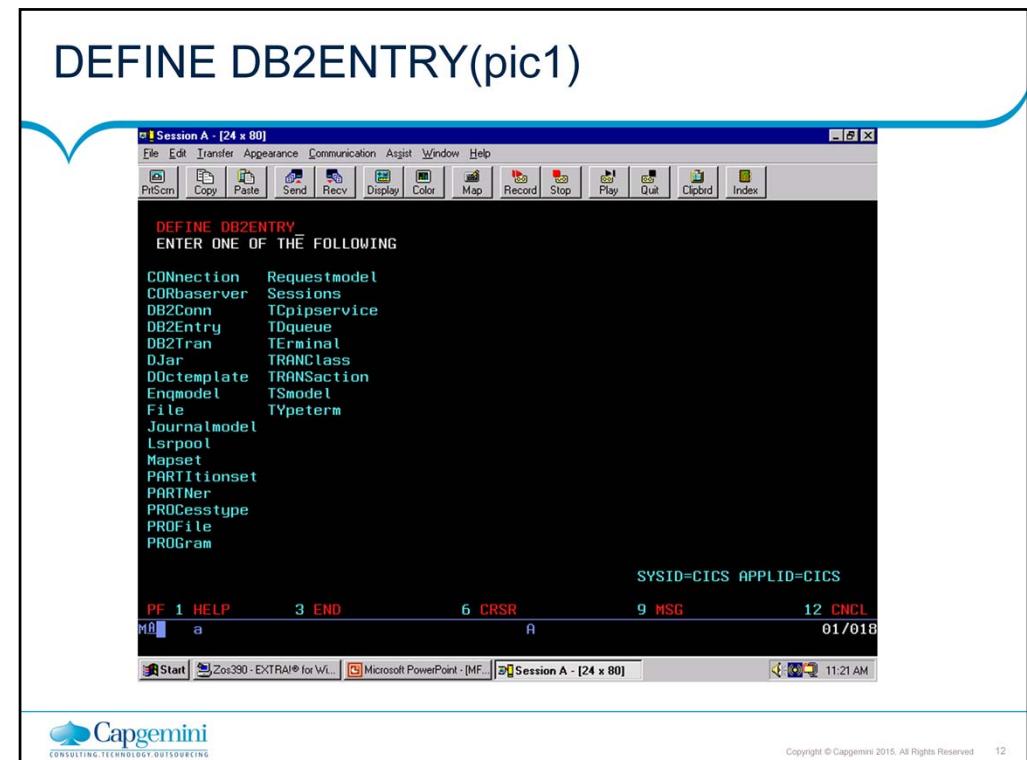
```
Session A - [24 x 80]
File Edit Transfer Appearance Communication Assist Window Help
[Icons] Copy Paste Send Recv Display Color Map Record Stop Play Clipbd Index

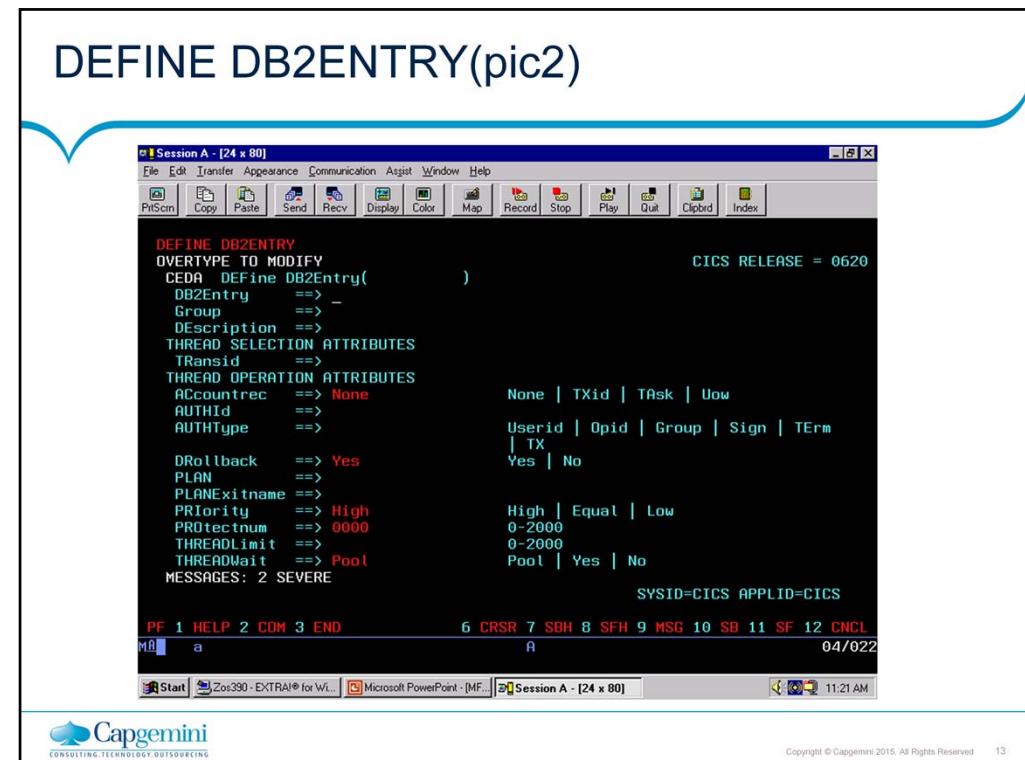
DEFINE DB2CONN
OVERTYPE TO MODIFY
CEDA DEFINE DB2Conn( ) CICS RELEASE = 0620
+ THREADError ==> N906D N906D | N906 | Abend
POOL THREAD ATTRIBUTES
ACcountrec ==> None None | TXid | TTask | Uow
AUTHId ==>
AUTHType ==>
DRollback ==> Yes Userid | Opid | Group | Sign | TTerm
PLAN ==> | TX
PLANExitname ==
Priority ==> High High | Equal | Low
THREADLimit ==> 3-2000
THREADWait ==> Yes Yes | No
COMMAND THREAD ATTRIBUTES
COMAUTHTId ==>
COMAUTHTType ==> Userid | Opid | Group | Sign | TTerm
COMThreadlim ==> 0001 | TX
0-2000
MESSAGES: 2 SEVERE
SYSID=CICS APPLID=CICS

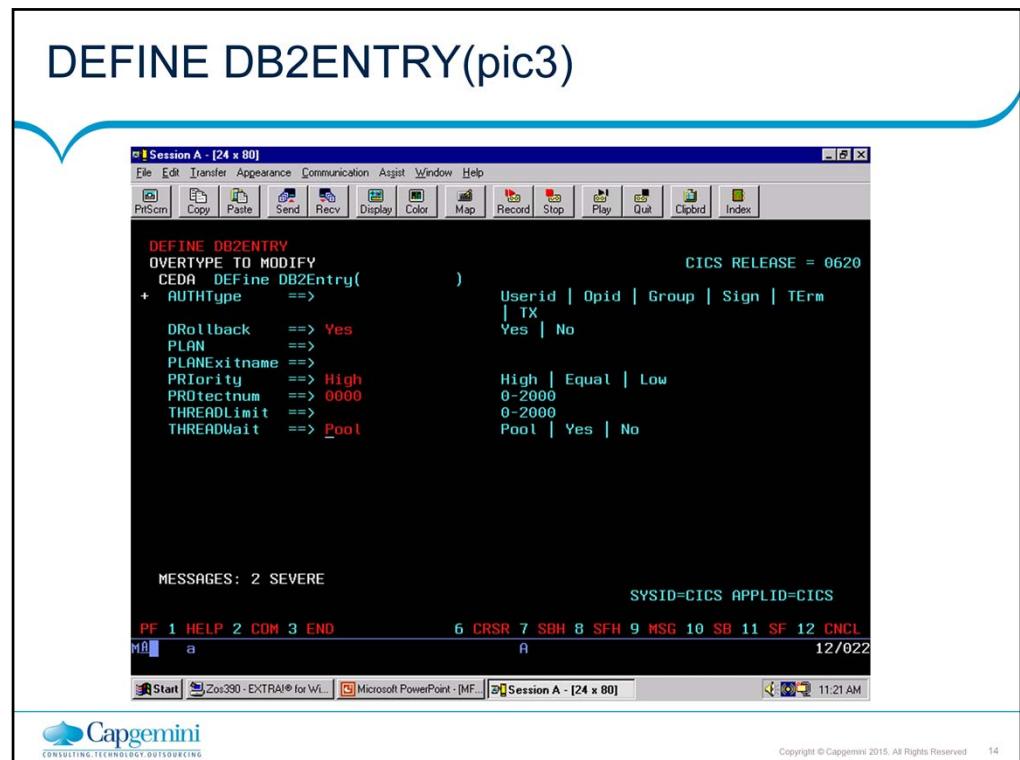
PF 1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
MB a                         A
04/022

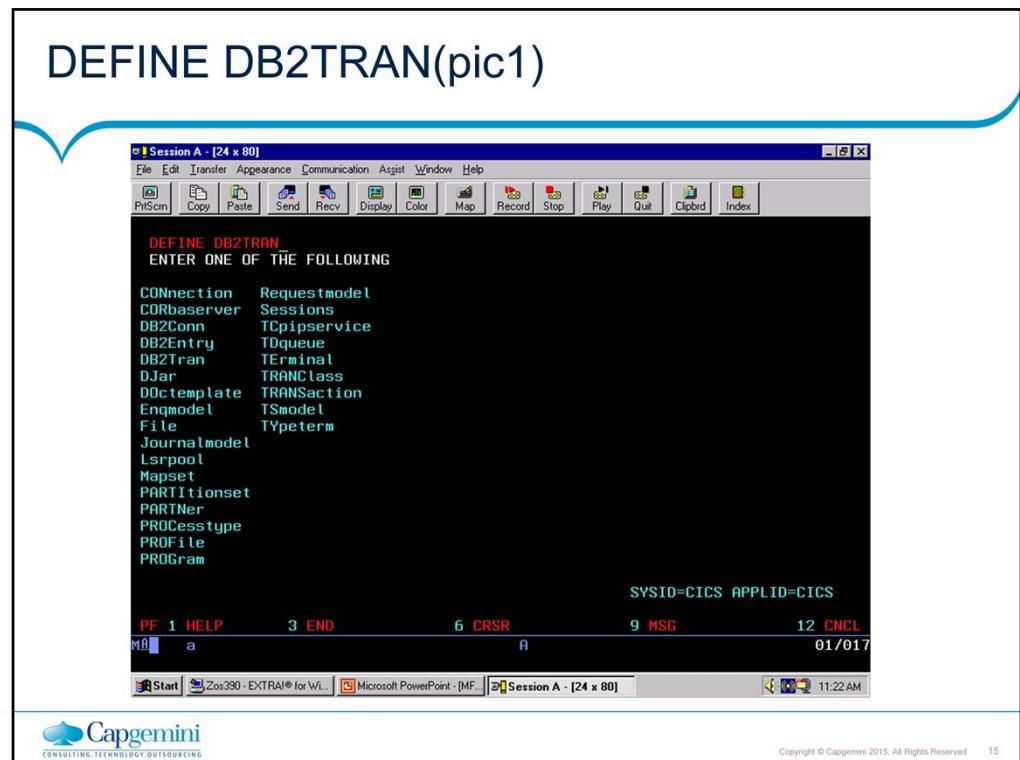
Start Zos390 - EXTRAI® for WL... Microsoft PowerPoint - MF Session A - [24 x 80]
Copyright © Capgemini 2015. All Rights Reserved. 10
11:20 AM
```

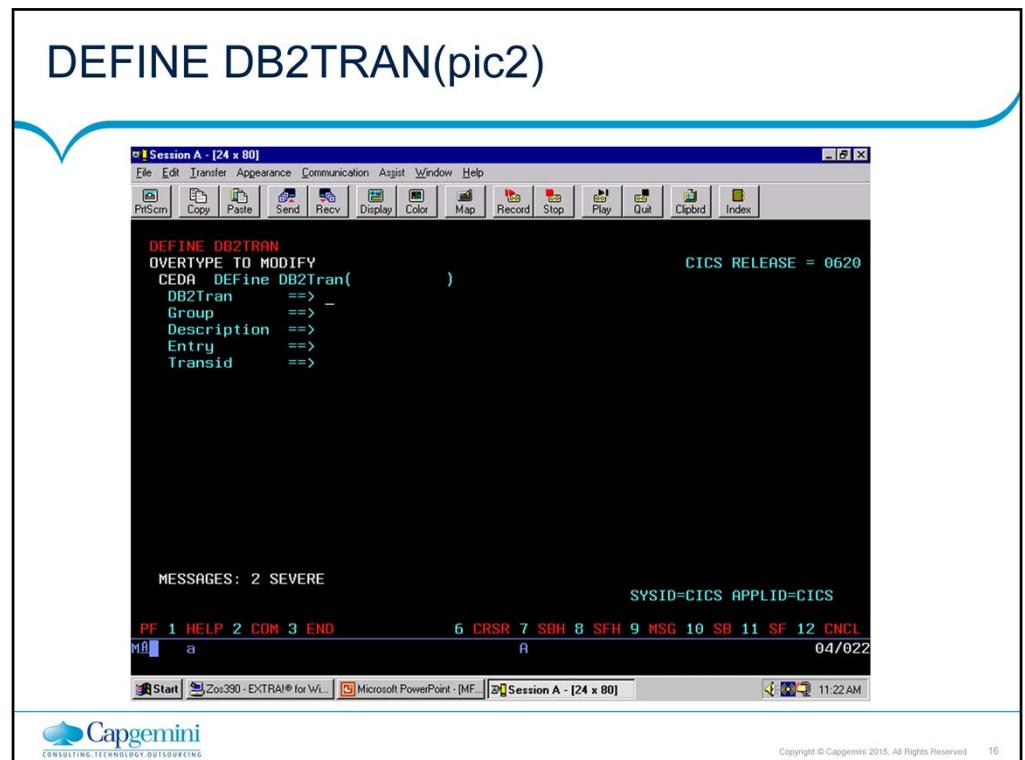


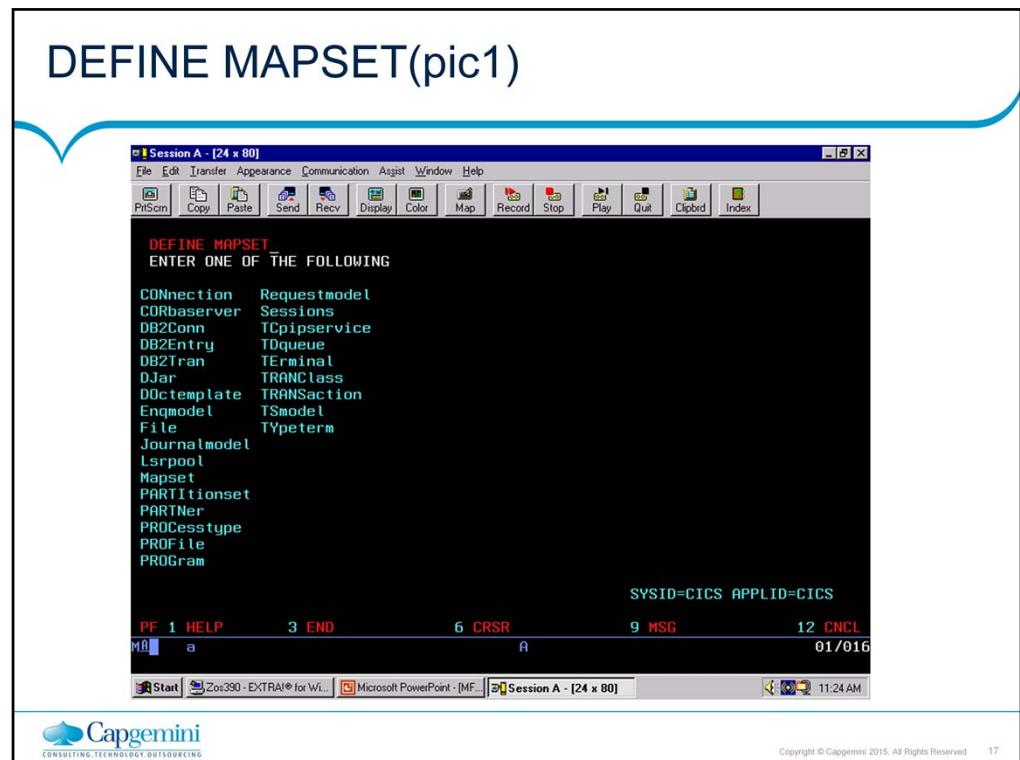


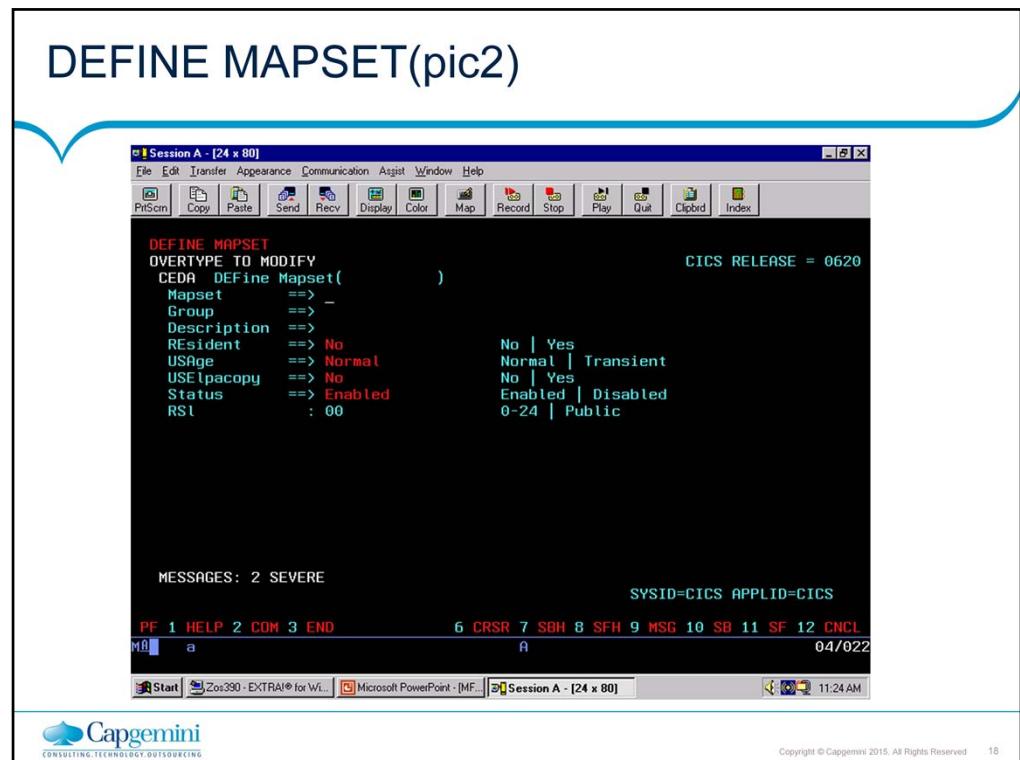


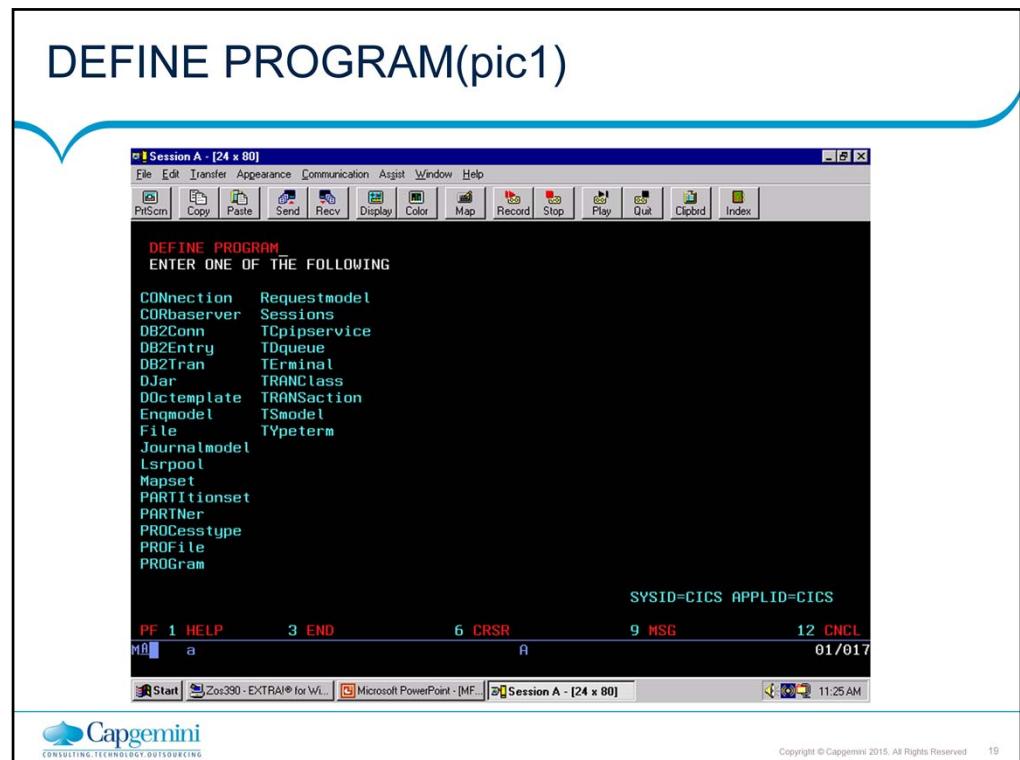


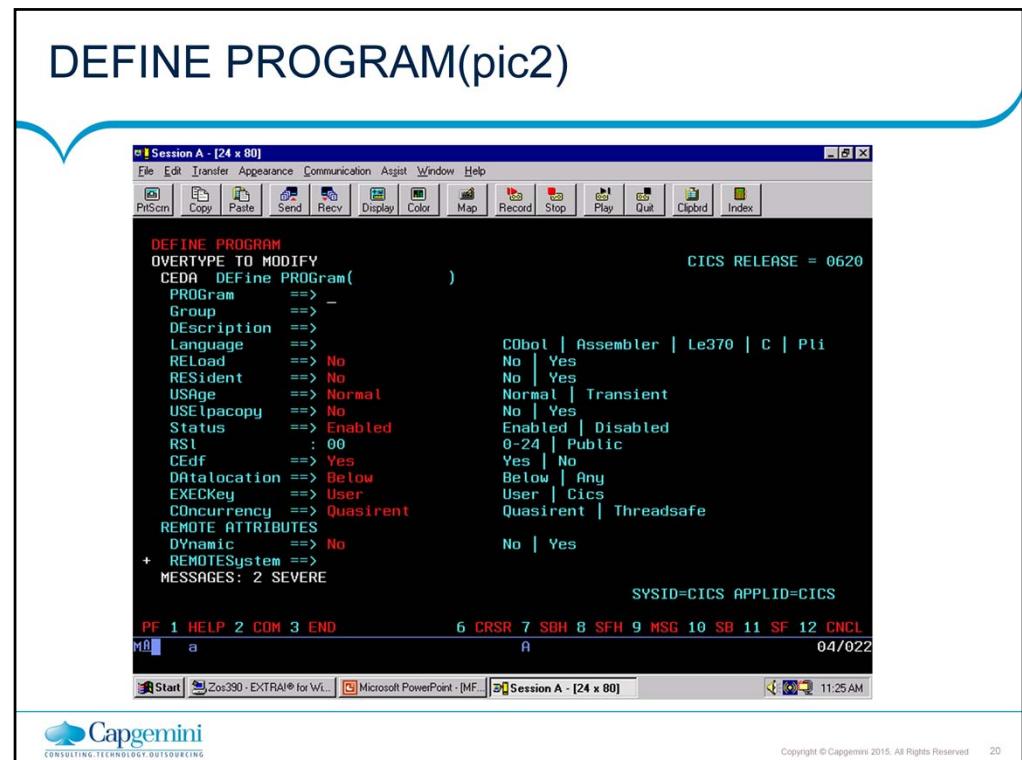


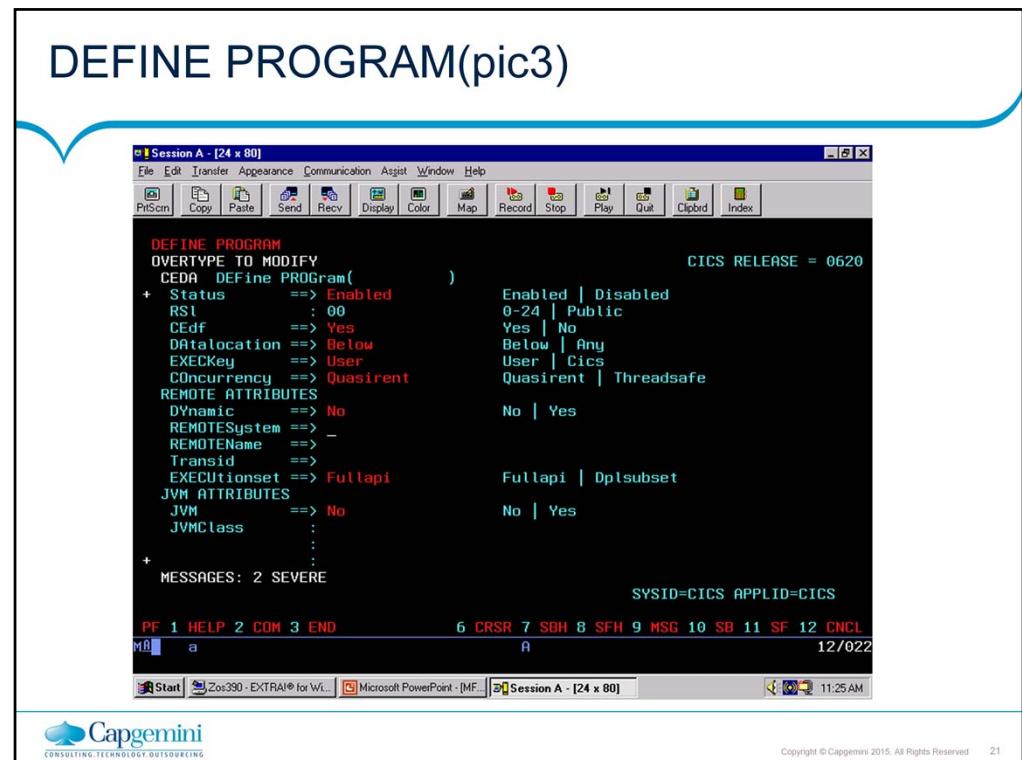


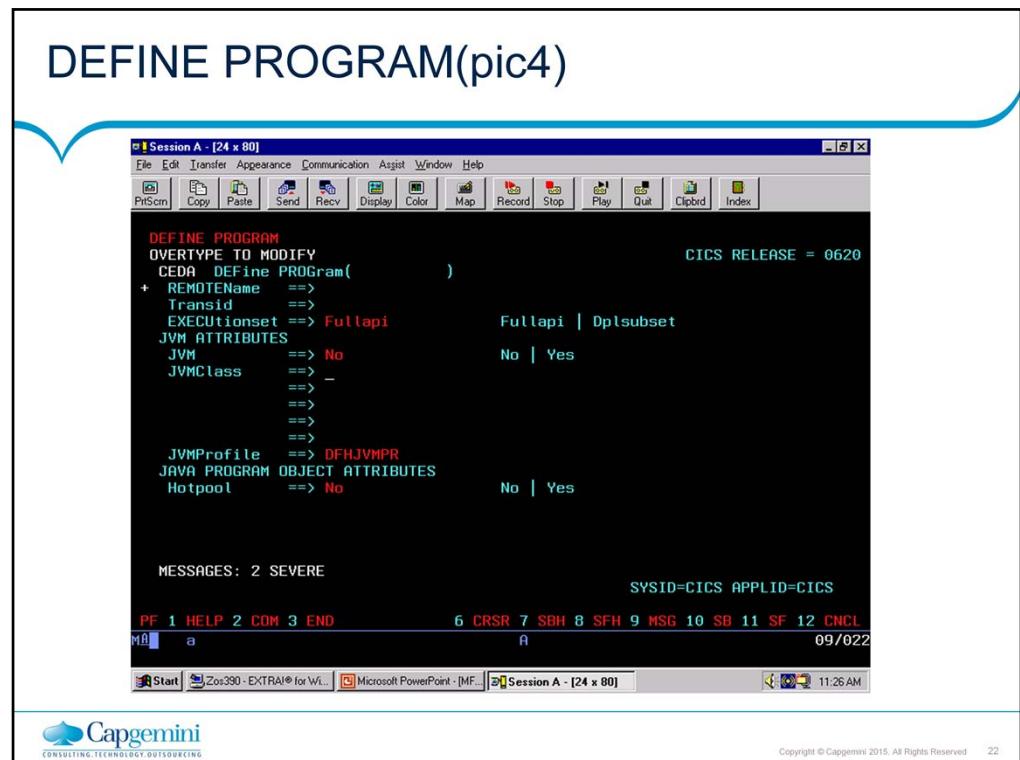


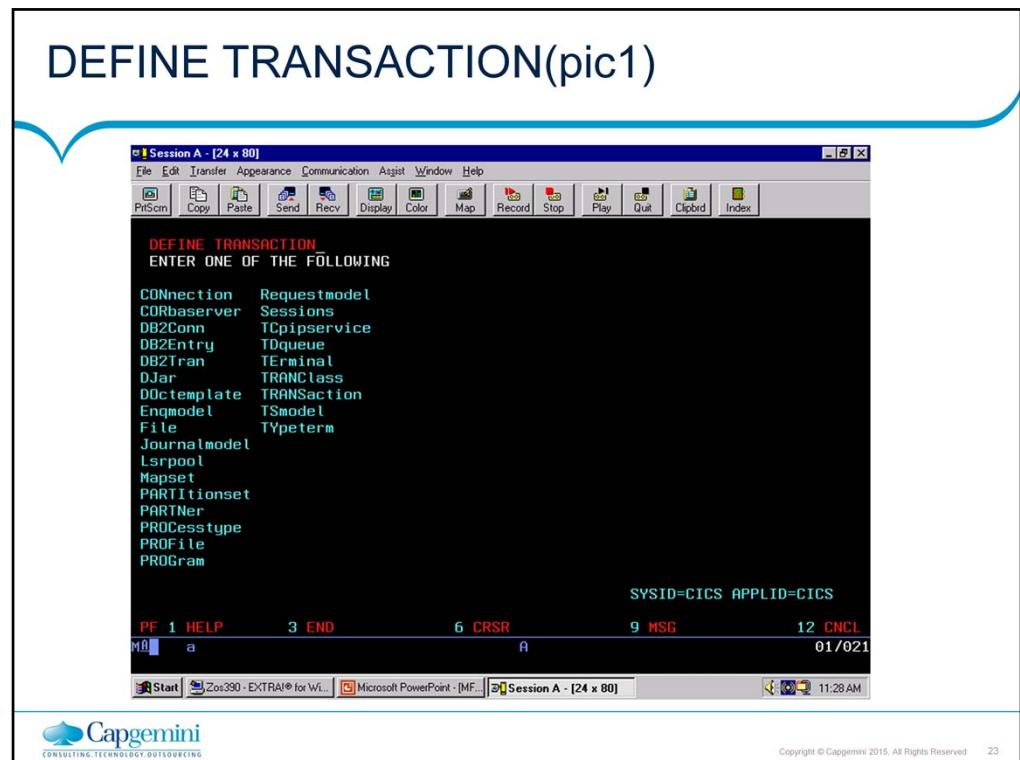


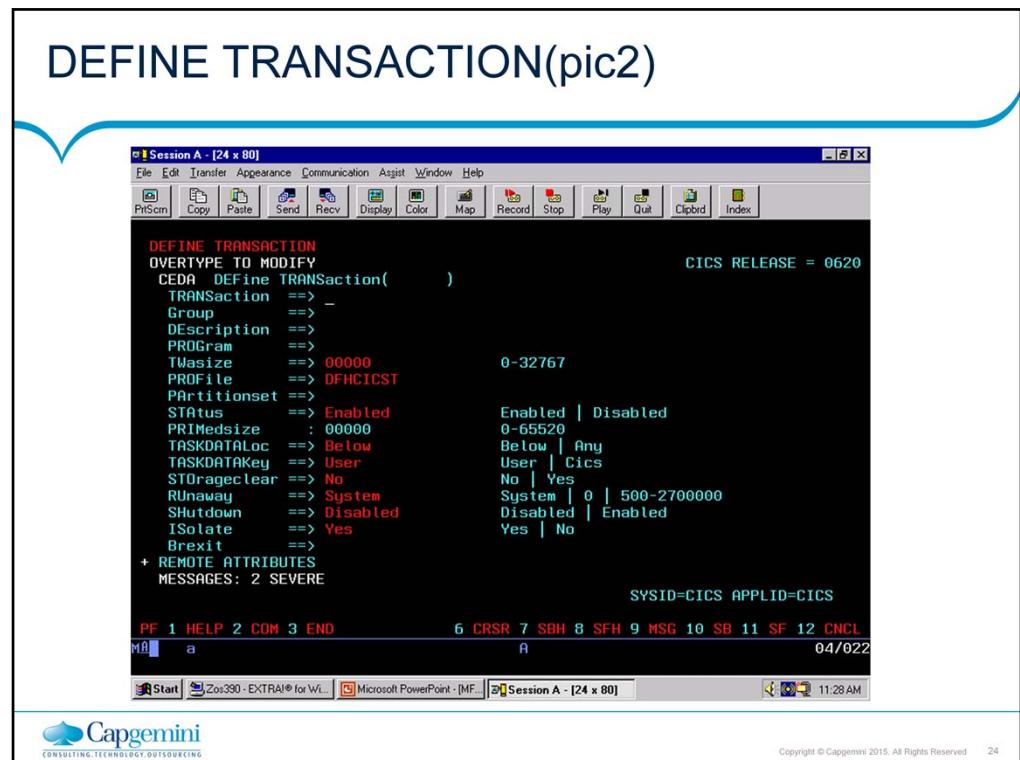


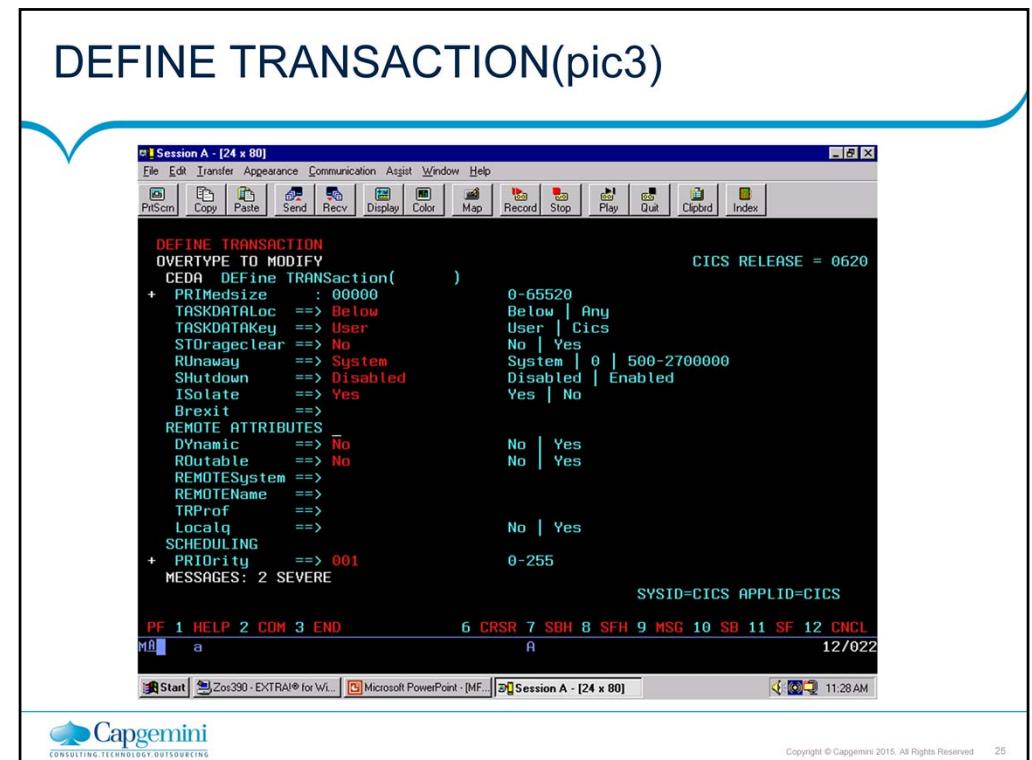


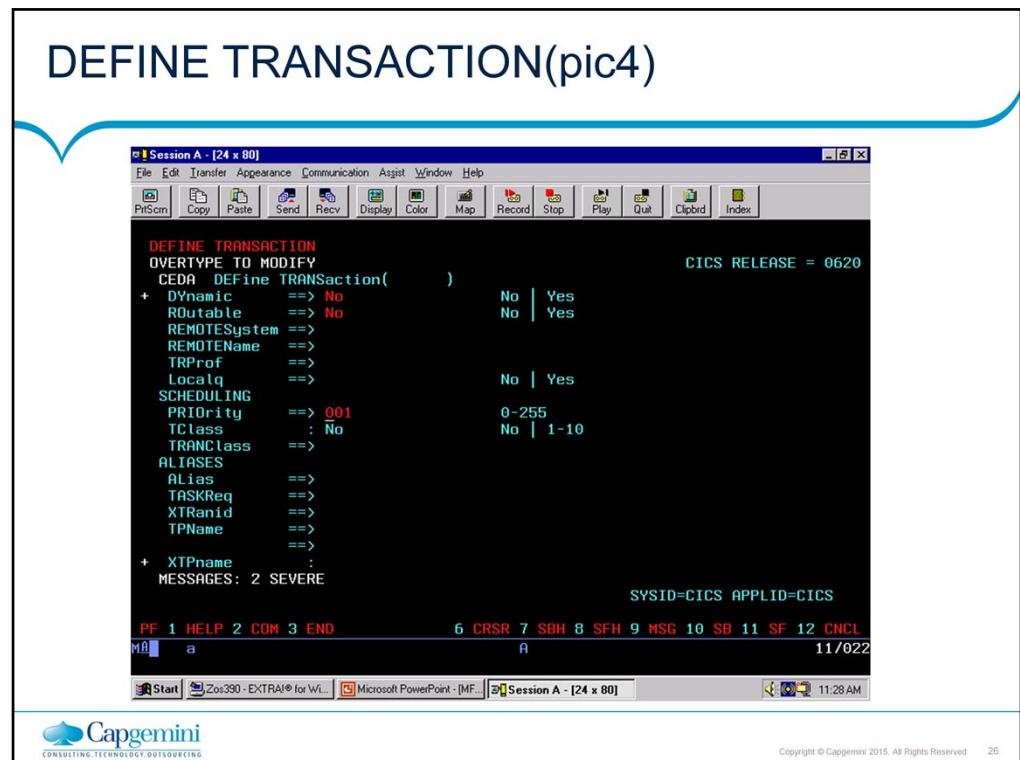


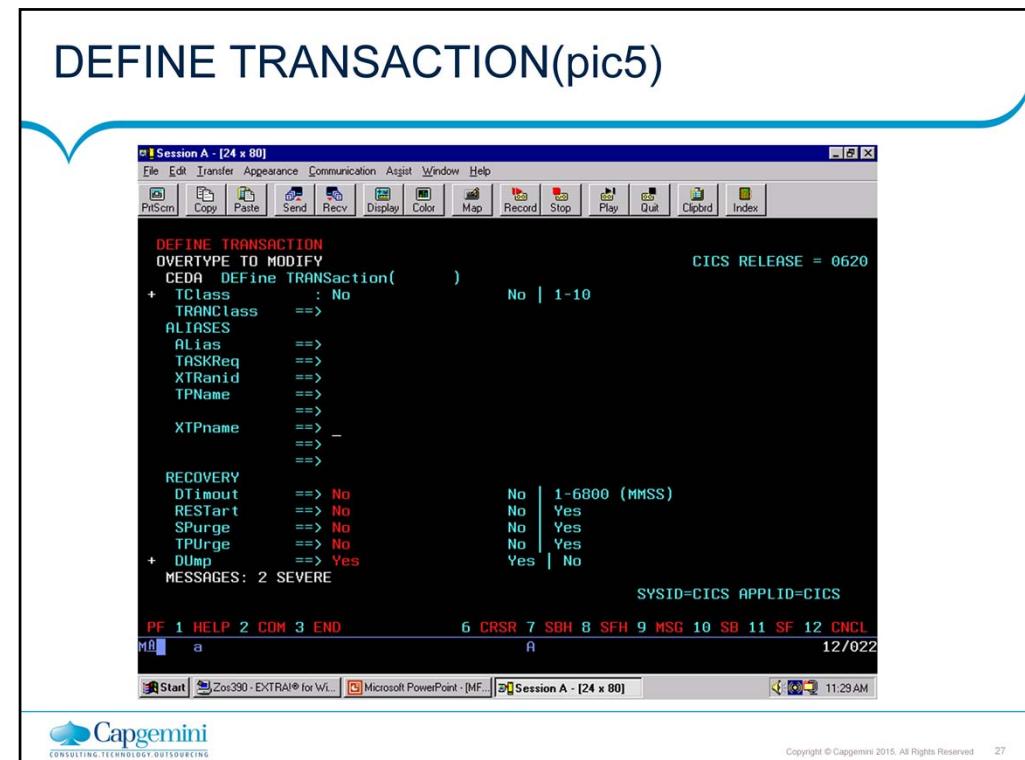


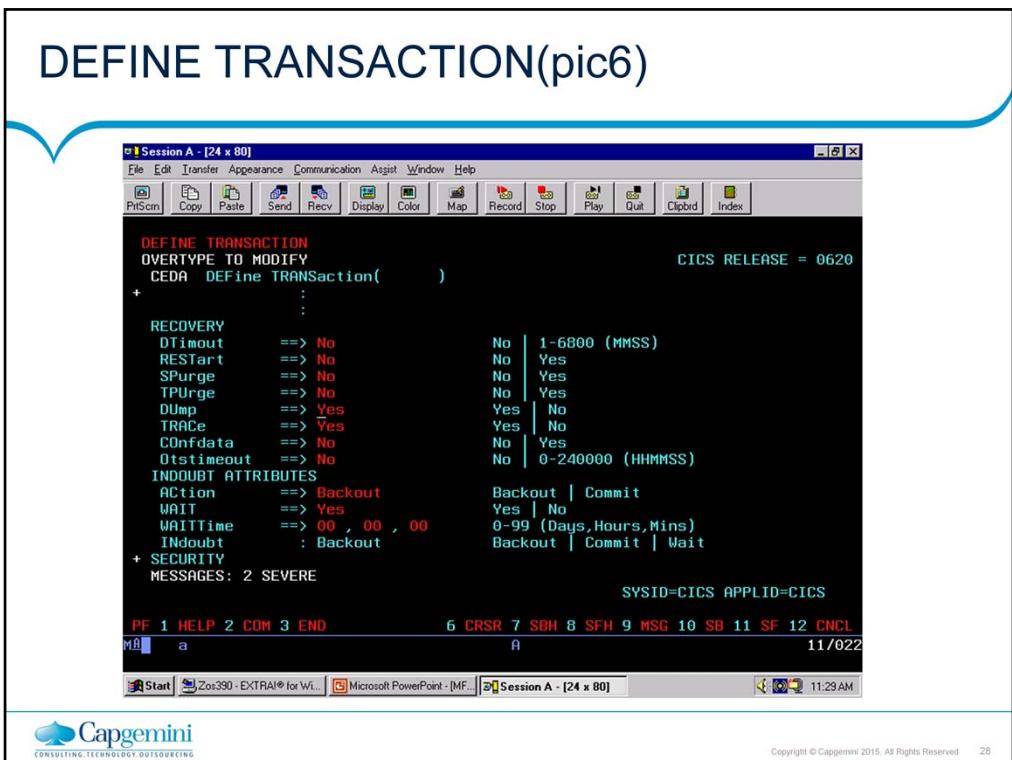


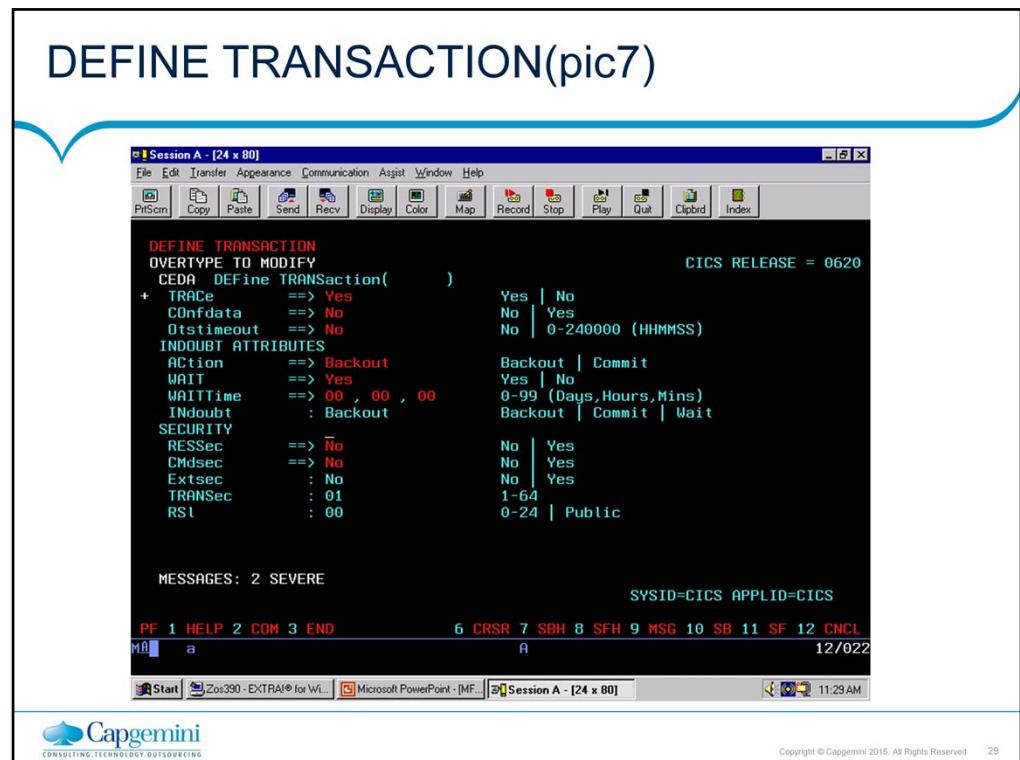












## Summary

- In this lesson, you have learnt about:
  - CICS-DB2 Interface



**CICS**

Lesson 18: CICS Web  
services

## Lesson Objectives

- In this lesson, you will learn the following topics:
  - Introduction to CICS Web services



## Introduction

- CICS/TS 3.1 continued this implementation started a few years ago.  
It now provides for:
  - Support for SOAP and Web Services
  - New resource definitions which will help in the implementation of web applications
  - Enhanced WEB API which will allow a CICS program to become a client on the web
- CICS/TS 3.2 added the following features:
  - Support for WSDL 2.0
  - Support for inter-region connectivity with TCPIP
  - Support for Dynamic Storage management above the Bar



Copyright © Capgemini 2015. All Rights Reserved. 3

CICS Integration enables re-use of CICS applications within broader SOA (Service Oriented Architecture) business scenarios, via standard APIs and protocols

CICS Transaction Server V3.1 provides capabilities to enable CICS-based applications to be integrated with a Service Oriented Architecture (SOA), enabling them to be exposed as Web Services. CICS has the ability to act as a Web Services service provider and service requestor which means it can be seen as a full participant in this SOA world.

By allowing CICS applications to be wrapped in this way and exposed as services, it easily enables new interoperability between these applications. This provides services to enable virtual enterprises to link heterogeneous systems as required. Examples include mergers, where the resulting enterprise must integrate disparate IT systems and business processes, or the combination of the travel industry and pervasive computing, when a travel application can be exposed as a service and made available for use by various devices in a SOA

Web Services provide standards-based interfaces to software functionality. Each Web Service describes how other systems, known as

Web Service consumers, can connect to it and exchange information with it. Therefore, the consumers need have no knowledge beforehand about a Service, other than where to find it and that it is based on the common Web Services standards. This approach enables software developers to focus on the business issues not the architecture.

To ensure it is relatively simple to transform an existing CICS application into a Web Service, there is a application development capability supplied with CICS TS V3 called CICS Web Services Assistant. This support is provided for COBOL, C/C++ and PL/I thus ensuring traditional program languages are able to participate and deliver immediate value to your existing application set. Given the existing investment customers have made in CICS business transactions, this ability to easily leverage them in new business processes is of huge value to the customer.

These capabilities should be seen as a major advance over the SOAP for CICS feature delivered on CICS TS V2 and we have taken the feedback from customers experience with that feature and incorporated them into the integrated web services capability in Versopn 3. With the provision of workload distribution and resource management facilities for this new workload, it ensures it receives the qualities of service expected for a CICS function.

To help with best practice, a new sample application is provided which illustrates how to code and implement a Web Service application. This ensures a customer business can receive immediate value from this ability.

### **V3.2**

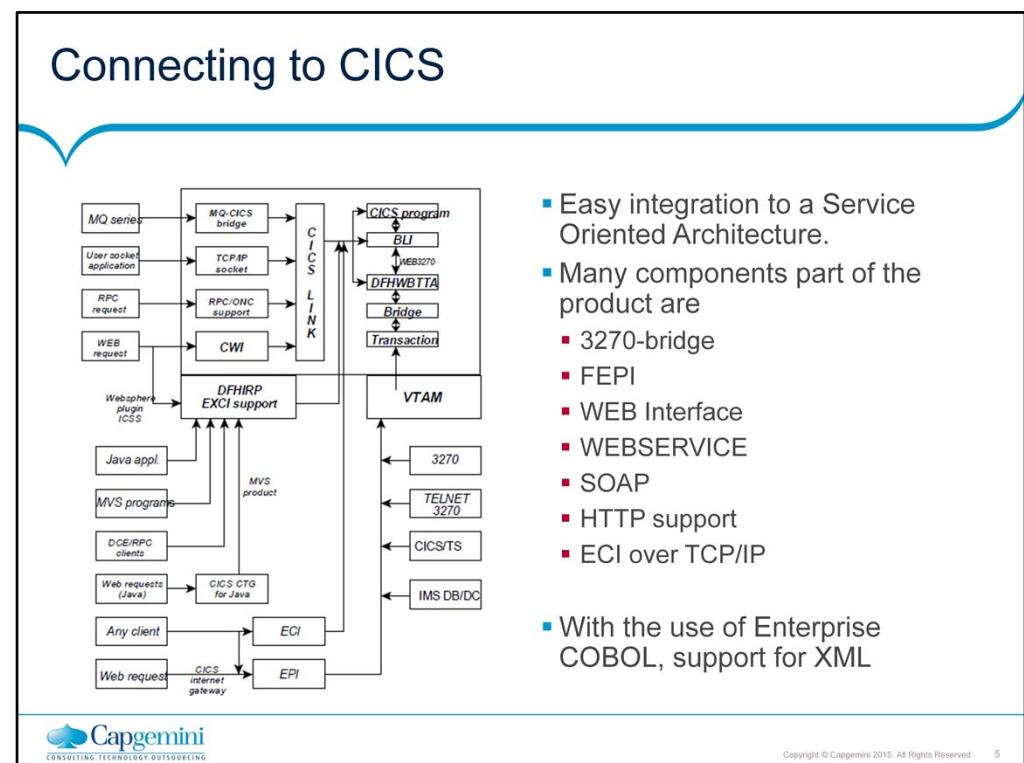
With CICS TS V3.2 we have enhanced the functionality for Web services by continuing its support for the latest Web Services standards and particularly by extending for WS-Security. This increases the maturity of the Web services implementation. In CICS TS V3.2 we are compliant with the latest specification for WSDL 2.0, WS-I BP and SSBP 1.0. CICS TS V3.2 builds on the framework provided by WS-Security in CICS TS V3.1. WS-Security provided the capability to protect the contents of a message through a mechanism for attaching security-related information targeted at a specific recipient. New in CICS TS V3.2 is support for WS-Trust which enables an authentication broker to negotiate trust between client applications and Web services, removing the need to build a chain of direct trust to all clients, with the corresponding system management burden in maintaining these relationships.

Optimisation of large data objects is provided by MTOM and XOP in CICS TS V3.2. MTOM and XOP define a method to significantly reduce the size of large data objects such as jpg, gif, wav, pdf, doc, blob within SOAP messages and so greatly reduces message parsing processing time and optimizes its transmission.

We have started to reduce dependency on SNA by moving towards making IP pervasive for CICS inter-connectivity starting with DPL. IP interconnectivity for DPL offers transactional syncpointing, security control and for a link with commarea or channels and containers.

As new capabilities are introduced for TCP/IP communications, CICS TS V3.2 also introduces management and control Facilities that allow work passing in and out of regions over a network to be monitored and managed. This allows you to Identify bottlenecks or blockages in the system, reconfigure the regions for improved throughput, and locate tasks that

have stalled. So that about completes CICS Application Connectivity, let now move to CICS Application Reuse, where we breath new life into programs and applications...



As identified in the above picture, CICS is now accessible from just about anything that runs on a computer. By linking to the Business Logic Interface, CICS can interact with:

- MQ-series
- Your own TCP/IP socket programs
- Remote Procedure Calls
- Web Url's pointing at CICS programs

EXCI can be used to support:

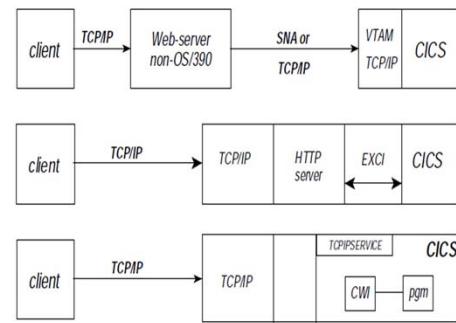
- Requests from OS/390 web servers
- MVS programs wanting access to CICS data
- DCE clients
- Java clients

Using CICS/CLIENTS ECI interface, any client program can gain access to CICS/BLI.

CICS for INTERNET GATEWAY, combined with the External Presentation Interface, can provide access to CICS server programs

## Connecting to CICS

- There are 4 types of connection:
  - Non-zOS servers that connect to CICS using a network
  - zOS servers that connect to CICS using EXCI.
  - Requests sent directly to CICS in HTTP format
  - CICS acting as an HTTP client and sending requests to other HTTP servers



## Weservice Overview

- Software system designed to support interoperable machine-to-machine interaction over a network.
- It has an interface described in a WSDL format (Web Service Definition Language).
- A Web service is described using a standard, formal XML notion, called its service description, that provides all of the details necessary to interact with the service, including message formats (that detail the operations), transport protocols, and location.
- It conforms to open-standards including:
  - SOAP 1.1 and 1.2
  - HTTP 1.1
  - WSDL 2.0
- CICS can be a requester or provider of services or both
- The supplied software supports 2 transport protocols:
  - MQ-series
  - HTTP



Copyright © Capgemini 2015. All Rights Reserved 7

### Weservice Overview

Web services is a technology for deploying, and providing access to, business functions over the World Wide Web using loosely coupled servers. Web services make it possible for applications to be integrated more rapidly, easily, and cheaply than ever before.

The CICS/TS V4 implementation is conformed to the following standards:

- ◆ HTTP 1.1
- ◆ SOAP 1.1 and 1.2
- ◆ WSDL 2.0

Programs in CICS/TS V4 can act as requester or server. CICS/TS V4 supports the following transport mechanism:

- ◆ MQ-series
- ◆ HTTP

## Wbservice Overview

CICS/TS V4 also provides a web services assistant which can help deploying applications.

- The assistant will support applications written in:
  - COBOL
  - PL/I
  - C
  - C++
- There are 2utilities:
  - DFHLS2WS
  - DFHWS2LS
- Few more utilities were introduced in CICS/TS 4. These utilities will help you convert data structures to schemas and convert schemas to data structure.
- These utilities can also be used to help creating webservices
  - DFHSC2LS
  - DFHLS2SC



Copyright © Capgemini 2015. All Rights Reserved 8

### Wbservice Overview

The support for Web services includes the CICS Web services assistant, a set of batch utilities which can help you to

- transform an existing CICS application into a Web service
- enable a CICS application to use a Web service provided by an external provider.

The assistant can create a WSDL document from a simple language structure, or a language structure from an existing WSDL document, and supports COBOL, C/C++, and PL/I. It also generates information used to enable automatic runtime conversion of the SOAP messages to containers and COMMAREAAs, and *vice versa*.

Program DFHLS2WS will transform a language structure into a WSDL document. Program DFHWS2LS will transform a WSDL document into a language structure.

Two new utilities were added in CICS/TS V4:

- DFHSC2LS is a utility that will convert an XML schema to language structure.
- DFHLS2SC is a utility that converts a language structure to XML schema
- Both utilities support COBOL, C/C++ AND PL/I
- CICS does not support WXS schema; only XSD schemas are supported

## Weservice Overview

- WEB SERVICE application can be deployed:
  - Using the WEB services assistant, this will bring down the effort of programming
    - Using a data structure, the CICS web services assistant can be used to generate the CICS resources that need to be deployed
    - The message will be transformed automatically into a SOAP message by CICS
  - Can write your own message handler to process the data you are receiving.
    - This will be the case when the message you are receiving is a non-SOAP messageNote:
- To use the web services assistant, SOAP must be the protocol used to transfer data between nodes4



Copyright © Capgemini 2015. All Rights Reserved 9

The assistant supports rapid deployment of CICS applications for use in service providers and service requesters, with the minimum of programming effort.

When you use the Web services assistant for CICS, you do not have to write your own code for parsing inbound messages, and constructing outbound messages; CICS maps data between the body of a SOAP message and the application program's data structure.

Resource definitions are, for the most part, generated and installed automatically. You do have to define PIPELINE resources, but you can, in many cases, use one of the pipeline configuration file that CICS provides. These are:

- basicsoap11provider.xml  
Pipeline configuration file for a service provider using the SOAP 1.1 message handler.
- basicsoap11requester.xml  
Pipeline configuration file for a service requester using the SOAP 1.1 message handler.

If you decide not to use the CICS Web services assistant, you will have to:

- Provide your own code for parsing inbound messages, and constructing outbound messages
- Provide your own pipeline configuration file

Define and install your own URIMAP and PIPELINE resources

## Wbservice deployment

- 2 files are required to deploy a webservice:
  - The Web Service Description Language file
    - This file is known as the Webservice file and has a file type of WSDL
  - The binding file
    - It has a file type of WSBIND
    - It used by the CICS data wrappers to format the data
      - When it is received by a webservice provider
      - Before it is given to a webservice provider
- 2 utilities are provided to help generating these files:
  - DFHLS2WS or DFHLS2SC is used when setting up a service provider
    - It will take as input a data structure and create the above 2 files
  - DFHWS2LS or DFHSC2LS is used when setting up a service requester
    - It will take a WSDL file as input and create the data structure



Copyright © Capgemini 2015. All Rights Reserved 10

### Wbservice Deployment

To use web services in CICS, we need to generate 2 files:

- The web service description file:
  - It contains information about the mechanic of the webservice; information such as port number, program to invoke, target system information, etc. can be found in there
  - This is a file that is stored in HFS (Unix System Services File System) and it has a file type of WSDL
- The binding file:
  - Used by the CICS data wrappers to format data when it is received by the provider and before it is given to the requester
  - This is a file that is stored in HFS (Unix System Services File System) and it has a file type of WSBIND

In a typical environment, the provider application will be built first or will already be in existence. DFHLS2WS should be used to create the WSDL and binding files for the provider program. DFHWS2LS should use an existing WSDL file as input; this utility will create a binding file as output as well as data structures which can be copied into the requesting program. The schema utilities, DFHSC2LS and DFHLS2SC only manipulate 1 file, the XML schema.

## Webservice Message handlers & Pipelines

- Message handlers are invoked at specific points in the pipeline
- Pipelines will typically have 2 phases:
  - A request phase
  - A response phase
- Pipelines could be setup as:
  - Service provider pipelines
  - Service requestor pipelines
- Message handlers can:
  - Process the message
  - Interrupt the flow
  - Ignore the message entirely



Copyright © Capgemini 2015. All Rights Reserved 11

### Webservice Message Handlers & Pipelines

A message handler is a CICS program used to process a Web service request during input, and to process the response during output. Message handlers use channels and containers to interact with one another, and with the system. Message handlers are defined as part of the PIPELINE configuration.

PIPELINEs can be compared to a list of pre-defined actions which must all be done in order to execute a given function. They are defined to CICS by using a PIPELINE resource definition. CICS uses this definition to get the name of the PIPELINE configuration file; this file provides the system with the following information:

- Transport protocol being used
- Name of message handlers
- Information about the target environment

When a message is received, CICS will start the pipeline by invoking the first message handler; then, all message handlers will be invoked, 1 after the other, until the last message handler has been invoked. This is the request phase. Once all message handlers have executed, the pipeline will be in response phase and invoke all the message handlers in the reverse order. During the request phase, a message handler may decide to interrupt the process and provide a response immediately; in this case, the process will switch to response mode some handlers may not execute.

## Weservice Message Handler

- Used to process a request during input and process a response during output
- Set of programs which will be invoked at various times during the execution of the pipeline
  - Interaction between programs is via CHANNEL CONTAINERS
  - Interaction between programs and system is also using CHANNEL CONTAINERS
- What should be done:
  - Examine or change content of XML request
  - Pass XML request or response to the next message handler in the pipeline
  - Used to parse message and provide a response automatically
  - Handle errors



Copyright © Capgemini 2015. All Rights Reserved 12

### Weservice Message Handler

The message handler interface lets you perform the following tasks in a message handler program:

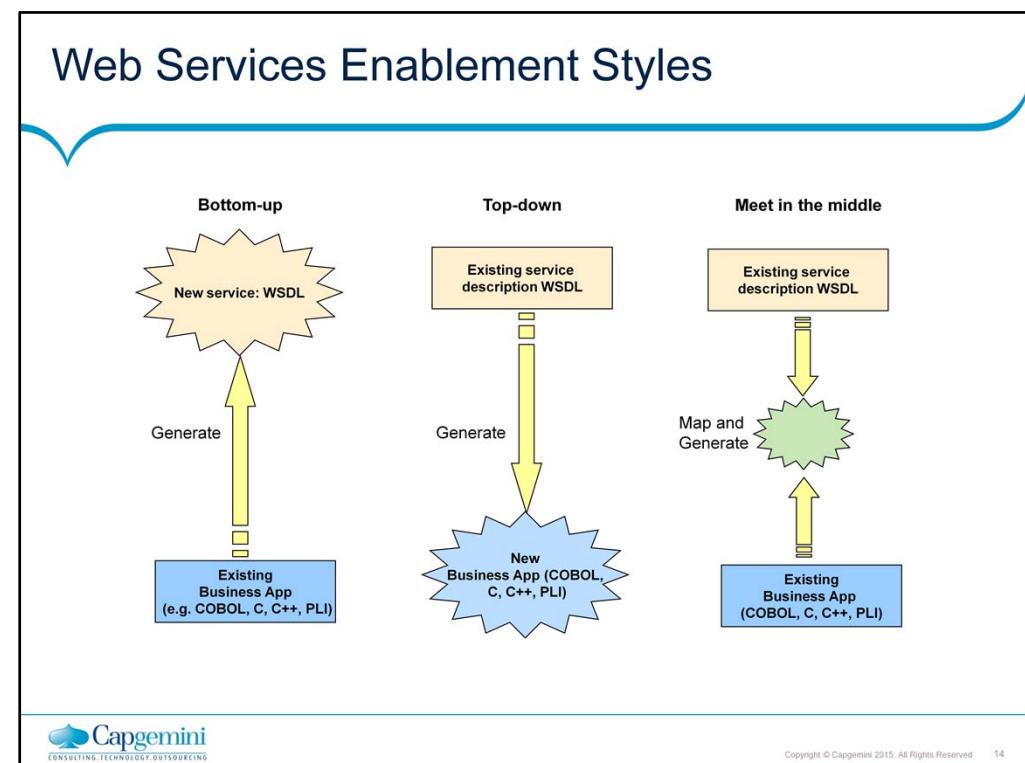
- Examine the contents of an XML request or response, without changing it
- Change the contents of an XML request or response
- In a non-terminal message handler, pass an XML request or response to the next message handler in the pipeline
- In a terminal message handler, call an application program, and generate a response
- In the request phase of the pipeline, force a transition to the response phase, by absorbing the request, and generating a response
- Handle errors

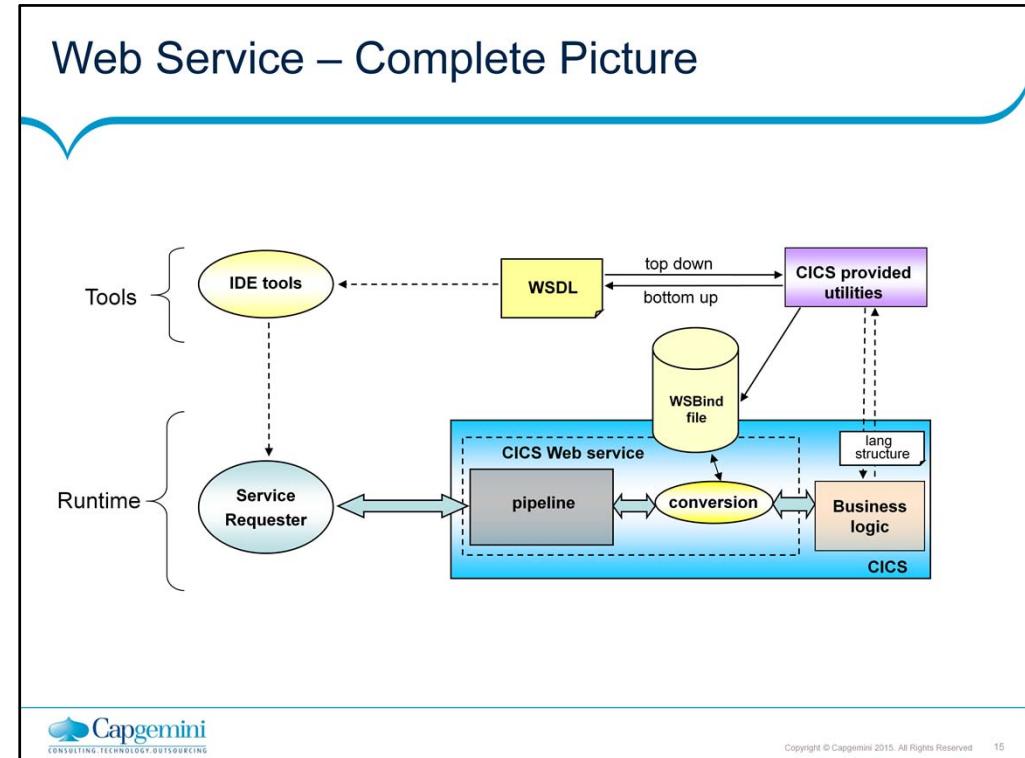
## Usage Scenarios

- CICS as a Service Provider using existing program (bottom up)
  - Existing application not changed
    - Existing language structure
- CICS as a Service Provider using new program (top down)
  - New application
    - Existing WSDL
- CICS as a Service Requester using a new program (top down)
  - New application
    - Existing WSDL



Copyright © Capgemini 2015. All Rights Reserved 13

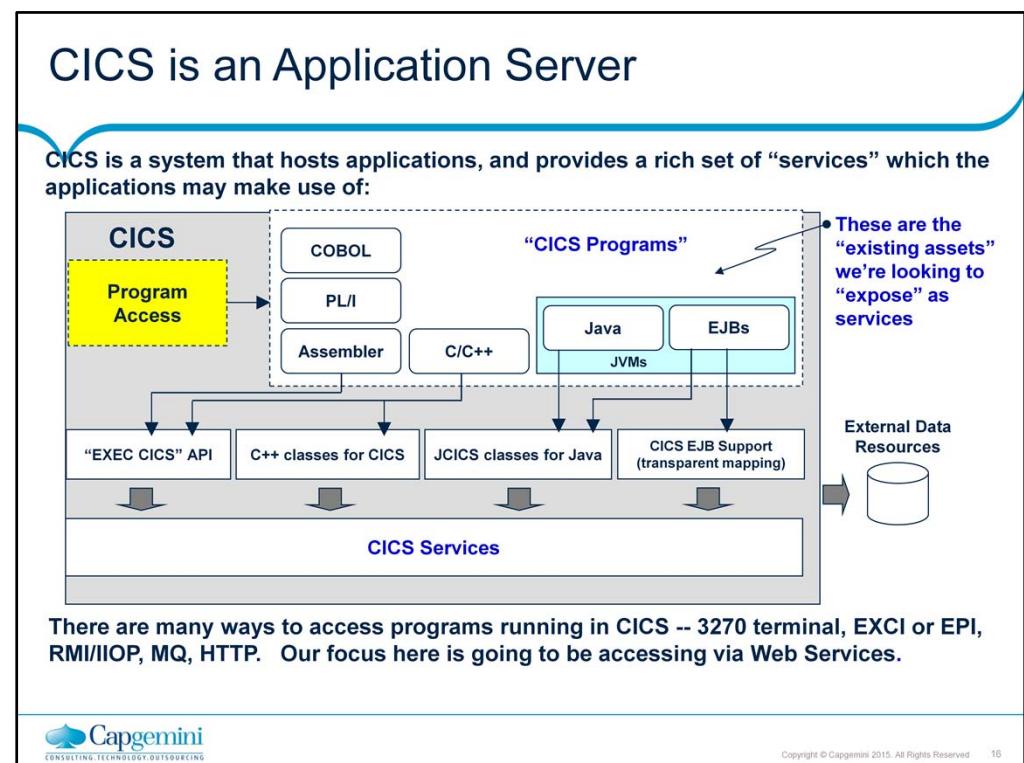




### Runtime Scenarios:

CICS can be the service provider: It is a traditional situation. CICS is the server in a client/server scenario. A client sends a request in to CICS.

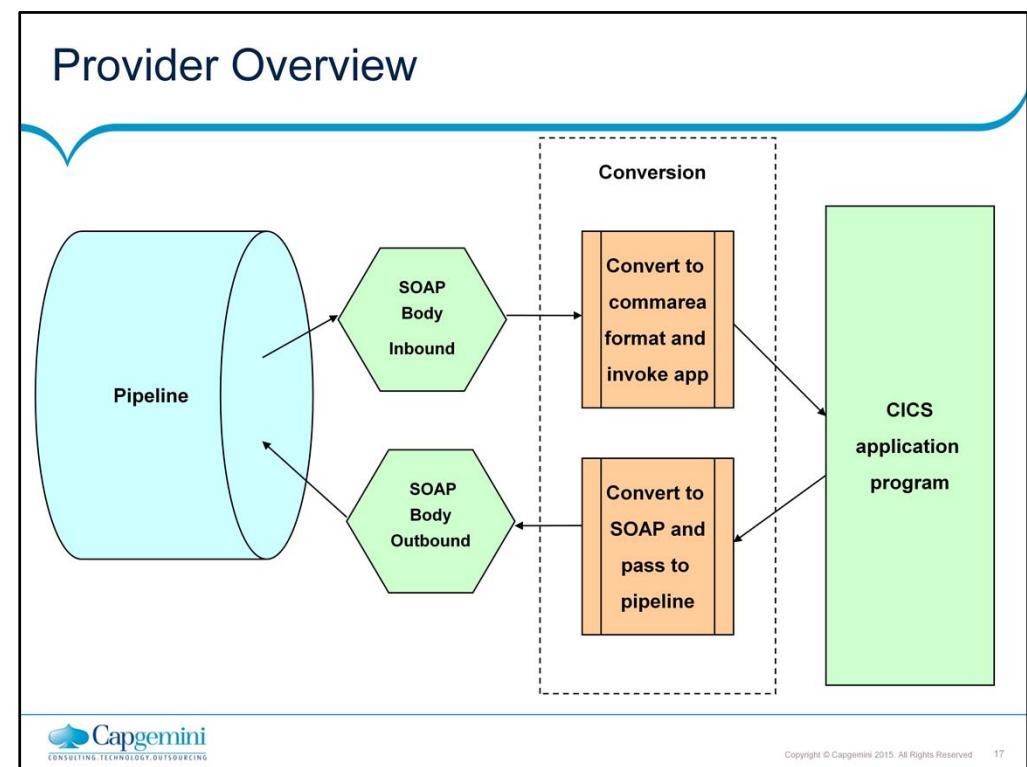
CICS can be the service requester: This is where CICS is the client in the client/server scenario. CICS is sending a request to execute a webservice to an external service provider.



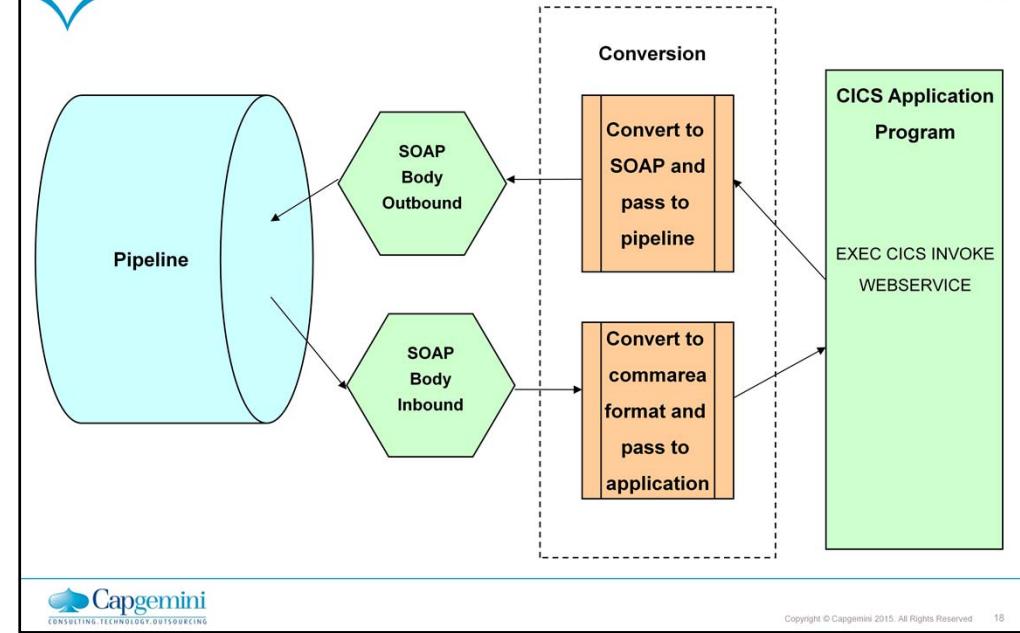
The first thing to put on the table is that CICS is in many ways an application server -- it hosts the running of applications within it, and it provides a rich set of “services” that applications can make use of rather than having to write their own. It provides a way to access backend data such as DB2 or IMS.

Programs written to run in CICS may be written in the traditional languages, COBOL, PL/I or Assembler; C or C++; as well as Java, in either “POJO” format (Plain Old Java Object) or EJB (Enterprise Java Bean). Each of these can gain access to the services provided by CICS, but though different means based on the programming language.

When it comes to accessing programs hosted by CICS, there’s a myriad of ways. Later on we’ll have a picture that shows the options available to you. In this workshop our interest is showing how to access Web Services ... and specifically, Web Services that are exposing an existing CICS application as a Web Service.



## Requester Overview



Copyright © Capgemini 2015. All Rights Reserved 18

## Summary

- In this lesson, you have learnt:
  - Introduction to webservices



# CICS

## Lab Book

## Document Revision History

Date	Revision No.	Author	Summary of Changes
23-Jan-2006	1.0		Content Creation
13-Nov-2009	2.0	Vandana Mistry	Content Upgradation
11-Dec-2009		CLS Team	Review
30-Jun-2011	3.0	Rajita Dhumal	Content Upgradation
8 <sup>th</sup> -Aug-2012	3.1	Rajita Dhumal	Revamped after Assignment Review
2-Aug-2016	3.2	Veena K	Revamp Post integration

## Table of Contents

<i>Document Revision History</i> .....	2
<i>Table of Contents</i> .....	3
<i>Getting Started</i> .....	5
<i>Overview</i> .....	5
<i>Setup Checklist for CICS</i> .....	5
<i>Instructions</i> .....	5
<i>Lab 1. Create Maps</i> .....	6
<i>1.1: Compile and Submit the Map</i> .....	7
<i>&lt;&lt;To Do&gt;&gt;</i> .....	7
<i>Assignments (Part I):</i> .....	7
<i>&lt;&lt;To Do&gt;&gt;</i> .....	10
<i>Lab 2. Translating and Compiling COBOL programs</i> .....	14
<i>2.1: Code the following COBOL program</i> .....	14
<i>Lab 3. Display Current Date and Time</i> .....	17
<i>3.1: Creating a map to display current date and time</i> .....	17
<i>Lab 4. File Handling</i> .....	18
<i>Assignments 1:</i> .....	18
<i>Assignments 2:</i> .....	19
<i>Lab 5. Analysis, Enhancement and Debugging Assignments</i> .....	21
<i>Lab 6. Analysis, Enhancement and Debugging Assignments</i> .....	23
<i>Functional components of the case study:</i> .....	23
<i>Lab 7. Browse records</i> .....	24
<i>5.1: Write a program for Customer Inquiry Module.</i> .....	24
<i>Lab 8. Temporary Storage Queue</i> .....	25
<i>6.1: Using TSQ in the MODIFY routine</i> .....	25
<i>Lab 9. Menu Handling</i> .....	26
<i>7.1: Creating a Menu driven application</i> .....	26
<i>Lab 10. Bank Application</i> .....	27
<i>1. Preface</i> .....	27
<i>2. Project Requirements</i> .....	27
<i>3. Design Details</i> .....	27
<i>3.1 File Layout</i> .....	27
<i>3.2 Screen Design</i> .....	28
<i>3.3 Report Layout</i> .....	29
<i>Appendices</i> .....	34

<i>Appendix A: CEBR, CECI, CEMT, CEDF .....</i>	34
<i>    Appendix A-1: CEBR: .....</i>	34
<i>    Appendix A-2: CECI .....</i>	38
<i>    Appendix A-3: CEMT .....</i>	42
<i>    Appendix A-4: CEDF .....</i>	51
<i>Create Maps using CA-Realia Simulator.....</i>	54
<i>Translating and compiling COBOL programs using Ca-Realia.....</i>	63
<i>File Handling using CA-Realia.....</i>	71
<i>Appendix B: Table of Figures .....</i>	74

## Getting Started

### Overview

This lab book is a guided tour for learning CICS. It comprises solved examples and 'To Do' assignments. Follow the steps provided in the solved examples and work out the 'To Do' assignments given.

### Setup Checklist for CICS

Here is what is expected on your machine in order for the lab to work.

#### Minimum System Requirements

- Network PCs with Mainframe Connectivity

#### Please ensure that the following is done:

- PASSPORT PC-TO-HOST (mainframe terminal simulator) is installed
- Connectivity to the Mainframe
- CA-Realia Software

### Instructions

- Create a directory by your name in drive <drive>. In this directory, create a subdirectory cics\_assgn. For each lab exercise create a directory as lab <lab number>.

## Lab 1. Create Maps

<b>Goals</b>	<ul style="list-style-type: none"> <li>• Learn to create, edit, and compile Maps.</li> </ul>
<b>Time</b>	90-120 minutes

### <<To Do>>

Create a Map with the following fields (constants):

**ABC LTD.**

1. File Maintenance
2. Inquiry Program

Enter your Choice:   

Ctrl+Enter = Process, F10 = EXIT

Message: \_\_\_\_\_

Operator Message: **Enter**-Process, **F3**-Exit, **F5**-Refresh

- a. Create a data entry field named **Choice** with the following attributes:

LENGTH =1

INITIAL=CHAR

ATTRB=UNPROT, IC, FSET

- b. Create a **Stopper** field for the **Choice** field, with the following attributes:

- c. Create the **Message** field (Display-only) with the following attributes:

- d. Define a **Stopper** field for the **Message** field.

- e. Define a **Dummy** field with the following attributes.

### 1.1: Compile and Submit the Map

<<To Do>>

#### Assignments (Part I):

Create the following Maps:

##### 1. Menu Screen:

Customer Information System	
Hindustan Pvt. Ltd.	Date: 99/99/99
<b>Menu</b>	
1.	Customer Maintenance
2.	Customer Inquiry
Enter Valid Choice: _____	
Message: _____	
Operator Message: <b>Enter</b> -Process, <b>F3</b> -Exit, <b>F5</b> -Refresh	

**Note:** The **Choice** field is a data entry field and the **Message** field is a display-only field. The remaining fields are constants.

##### 2. Key Screen:

### Customer Information System

Hindustan Pvt. Ltd.

Date: 99/99/99

Enter Customer code: \_\_\_\_\_

Message: \_\_\_\_\_

Operator Message: **Enter**-Process, **F3**-Menu, **F5**-Refresh

**Note:** The **Customer code** field is a data entry field and the **Message** field is a display-only field. The remaining fields are constants.

### 3. Detail Screen:

<b>Customer Information System</b>	
Hindustan Pvt. Ltd.	Date: 99/99/99
  <b>Modification/Inquiry/Data-entry</b>  	
Customer Code:	<hr/>
Name:	<hr/>
Address:	<hr/>
City:	<hr/>
Zip-Code:	<hr/>
  <b>Message:</b> <hr/>	
<b>Operator Message:</b> <hr/>	

**Note:** The Name, Address, City, and zip-code fields are data entry fields. The Customer code, Message, and the Operator Message fields are display-only fields. The remaining fields are constants.

Use following names for maps and mapsets:

Screen	Map Name	Mapset Name
Menu	menumap	menumap
Key	keymap	keymap
Detail	detmap	detmap

**Case Study 1: Voting Eligibility**

Here we are working with one case study-Voting Eligibility which accept birth details online and writing them into a Birth register with unique SSN thereby generating the VOTER ID.

Major tasks are as mentioned below:

- Accept the SSN of a person
- Check eligibility criteria to vote
- Generate voter-ID based on eligibility criteria.
- Modification of the birth details is also done.

Accept the person's details depending on the choice.

- 1 – BIRTH REGISTRATION
- 2 – DEATH REGISTRATION
- 3 – BIRTH MODIFICATION
- 4 – VOTER REGISTRY

If choice is 1, BIRTH DETAILS are entered.

If choice is 2, DEATH DETAILS are entered.

If choice is 3, the details of birth can be modified.

If choice is 4, the person can check whether he is eligible for applying the Voter ID and if some other choice is made other than these 4 options, an error message is displayed.

**Solve the given problems based on the above case study.**

**<<To Do>>**

**Problem 1:**

Refer to shared file named 'LOGIMAP' for BMS coding

- The participants have to analyze the program, remove the error and successfully execute the program.
- The participants have to document the error in the excel sheet along with steps taken to resolve the errors.

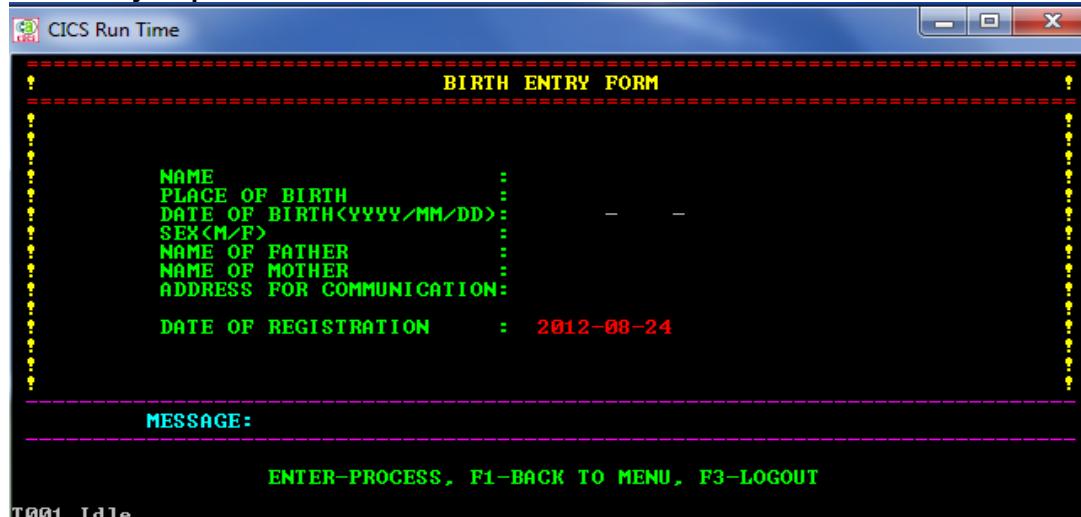
**Problem 2:**

Refer to shared file named 'MENMAP' for BMS coding

- The participants have to analyze the program, remove the error and successfully execute the program.
- BIRTH REGISTRATION choice is coded in the given program. Do enhance the program for remaining choices such as DEATH REGISTRATION, BIRTH MODIFICATION, VOTER REGISTRY, etc
- The participants have to document the error in the excel sheet along with steps taken to resolve the errors.

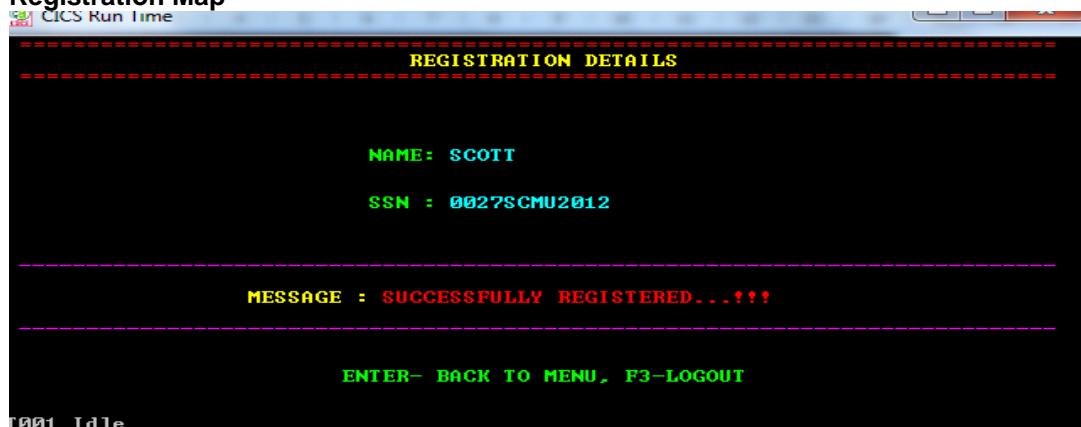
**Problem 3:**

Create the following maps:

**Birth Entry Map**

The terminal window title is "CICS Run Time". The screen displays a "BIRTH ENTRY FORM" with the following fields:  
NAME  
PLACE OF BIRTH  
DATE OF BIRTH<YYYY/MM/DD>  
SEX(M/F)  
NAME OF FATHER  
NAME OF MOTHER  
ADDRESS FOR COMMUNICATION:  
DATE OF REGISTRATION : 2012-08-24

At the bottom, there is a "MESSAGE:" field and a command prompt: "ENTER-PROCESS, F1-BACK TO MENU, F3-LOGOUT".

**Registration Map**

The terminal window title is "CICS Run Time". The screen displays "REGISTRATION DETAILS" with the following information:  
NAME: SCOTT  
SSN : 0027SCMU2012

At the bottom, there is a "MESSAGE : SUCCESSFULLY REGISTERED....!!!" field and a command prompt: "ENTER- BACK TO MENU, F3-LOGOUT".

**Birth Details Modification Maps**

CICS Run Time

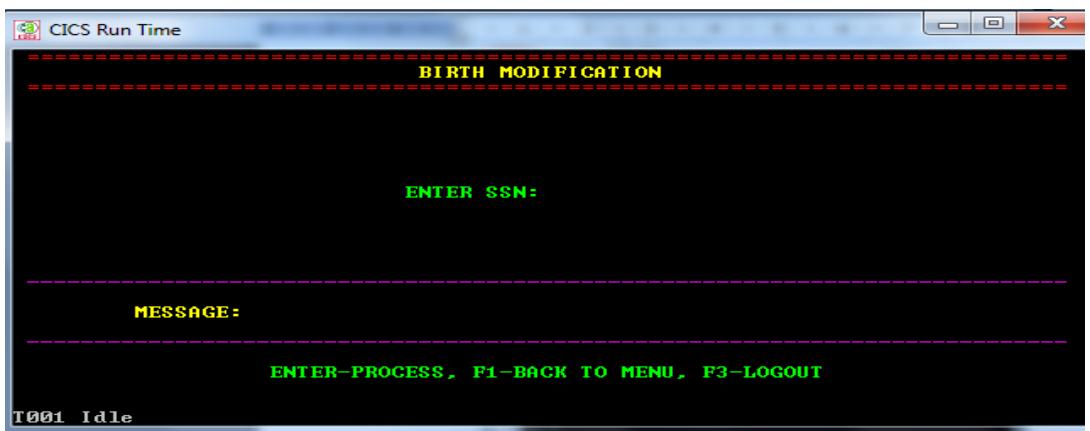
BIRTH MODIFICATION

ENTER SSN:

MESSAGE:

ENTER-PROCESS, F1-BACK TO MENU, F3-LOGOUT

T001 Idle



CICS Run Time

BIRTH DETAILS

SSN NO	:	0026SIMU1978
NAME	:	SITA
PLACE OF BIRTH	:	MUMBAI
DATE OF BIRTH<YYYY/MM/DD>	:	1978 - 09 - 29
SEX(M/F)	:	F
NAME OF FATHER	:	SUNIL
NAME OF MOTHER	:	RADHA
ADDRESS FOR COMMUNICATION	:	DADAR MUM
DATE OF REGISTRATION	:	2012-08-24

MESSAGE:

ENTER -MODIFY, F1 -BACK TO MENU, F3 -LOGOUT

T001 Idle



### Death Entry Map

CICS Run Time

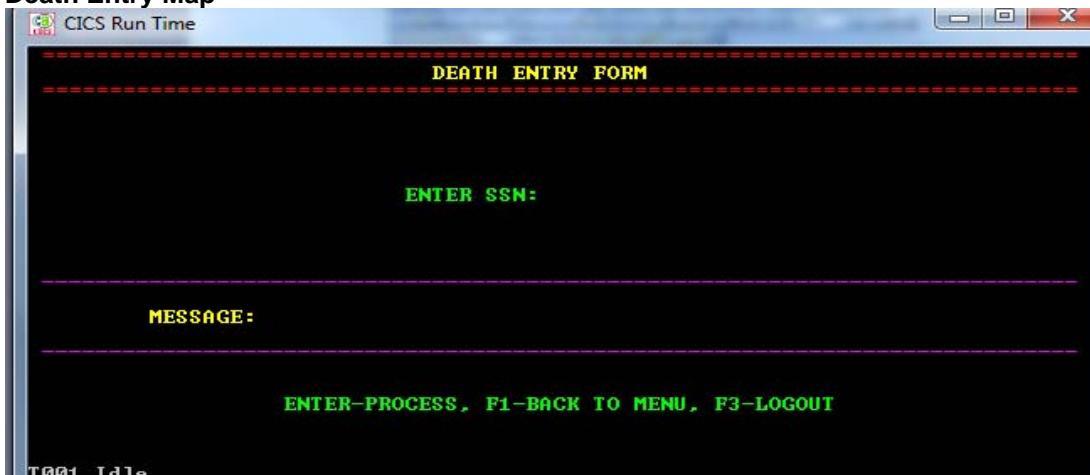
DEATH ENTRY FORM

ENTER SSN:

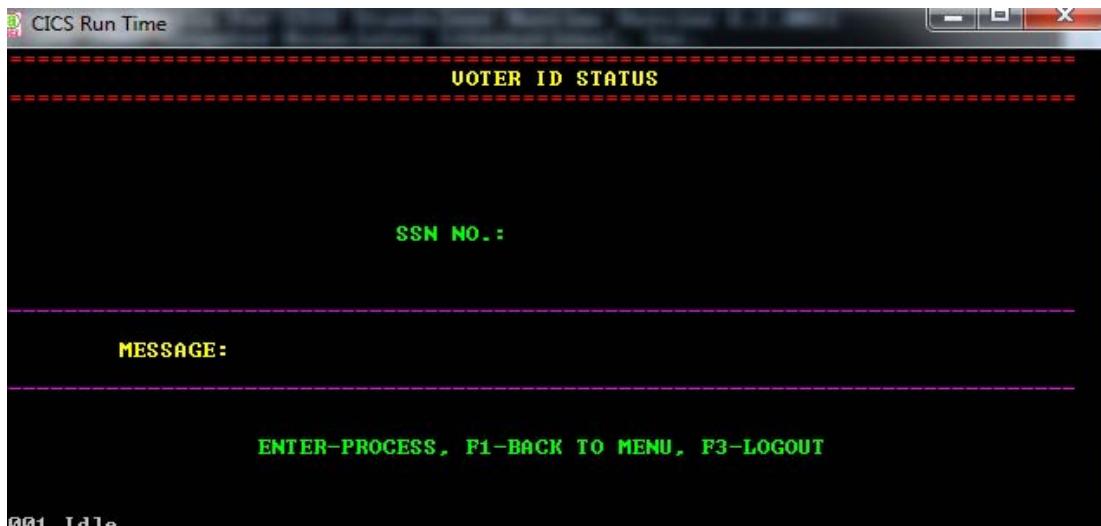
MESSAGE:

ENTER-PROCESS, F1-BACK TO MENU, F3-LOGOUT

T001 Idle



### Voter Status Map:



## Lab 2. Translating and Compiling COBOL programs

<b>Goals</b>	<ul style="list-style-type: none"> <li>To execute a simple pseudo conversational program</li> </ul>
<b>Time</b>	30 minutes

### 2.1: Code the following COBOL program

```

IDENTIFICATION DIVISION.
PROGRAM-ID. SAMP1.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01 WS-SWITCHES.
    05 WS-END-SESSION          PIC X(01) VALUE 'N'.
    01 WS-END-SESSION-MSG    PIC X(50) VALUE SPACES.
    01 WS-COMM-AREA          PIC X(01) VALUE SPACES.
*
COPY SAMPMAP.
COPY DFHAID.

LINKAGE SECTION.
01 DFHCOMMAREA      PIC X(01).
**
PROCEDURE DIVISION.
0000-MAIN SECTION.
    IF EIBCALEN = ZEROS
        PERFORM 1000-FIRST-TIME
    ELSE
        MOVE DFHCOMMAREA TO WS-COMM-AREA
        PERFORM 2000-SUBSEQUENT.
*
    IF WS-END-SESSION = 'Y'
        EXEC CICS
            SEND TEXT
                FROM(WS-END-SESSION-MSG)
                LENGTH(+50)
            ERASE
            FREEKB
        END-EXEC
        EXEC CICS
            RETURN
        END-EXEC
    ELSE
        EXEC CICS
            RETURN
            TRANSID('TRN1')
            COMMAREA(WS-COMM-AREA)
            LENGTH(+1)
        END-EXEC.

```

```

0000-EXIT.
  EXIT.
*

1000-FIRST-TIME SECTION.
  MOVE LOW-VALUES TO SAMPMAPI.
  MOVE -1 TO CHOICEL.
  EXEC CICS
    SEND MAPSET('SAMPMAP')
      MAP('SAMPMAP')
      FROM(SAMPMAPI)
      ERASE
      FREEKB
      CURSOR
    END-EXEC.
1000-EXIT.
  EXIT.
*
2000-SUBSEQUENT SECTION.
  IF EIBAID = DFHENTER
    PERFORM 2100-PROCESS-CHOICE
  ELSE IF EIBAID = DFHPF10
    MOVE 'SESSION ENDED' TO WS-END-SESSION-MSG
    MOVE 'Y' TO WS-END-SESSION
  ELSE
    MOVE 'INVALID ATTENTION KEY PRESSED' TO MESSAGEI.
*
  IF WS-END-SESSION NOT = 'Y'
    PERFORM 2500-SEND-MAP.
2000-EXIT.
  EXIT.
2100-PROCESS-CHOICE SECTION.
  PERFORM 2110-RECEIVE-MAP.
  IF CHOICEL = ZEROS OR CHOICEI = SPACES
    MOVE 'YOU MUST ENTER CHOICE' TO MESSAGEO
  ELSE
    IF CHOICEI = 1
      MOVE 'ASSIGN1 NOT READY' TO MESSAGEO
    ELSE IF CHOICEI = 2
      MOVE 'ASSIGN2 NOT READY' TO MESSAGEO
    ELSE
      MOVE 'INVALID CHOICE '    TO MESSAGEI.
2100-EXIT.
  EXIT.
2500-SEND-MAP SECTION.
  MOVE -1 TO CHOICEL.
  EXEC CICS
    SEND MAPSET('SAMPMAP')
      MAP('SAMPMAP')
      FROM(SAMPMAPI)
      ERASE
      FREEKB
      CURSOR

```

```
END-EXEC.  
2500-EXIT.  
EXIT.  
  
2110-RECEIVE-MAP SECTION.  
EXEC CICS HANDLE CONDITION  
    MAPFAIL(2110-MAP-FAIL)  
END-EXEC.  
  
EXEC CICS  
    RECEIVE MAPSET('SAMPMAP')  
        MAP('SAMPMAP')  
        INTO(SAMPMAPI)  
    END-EXEC.  
    GO TO 2110-EXIT.  
2110-MAP-FAIL  
    MOVE 'MAP FAIL OCCURRED' TO WS-END-SESSION-MSG.  
    MOVE 'Y' TO WS-END-SESSION.  
2110-EXIT.  
EXIT.
```

Example 1: Sample Program

## Lab 3. Display Current Date and Time

Goals	• To Create a Map to display current date and time
Time	1.5 Hrs

### 3.1: Creating a map to display current date and time

1. Create the following map:

DATE:	<input type="text"/>
TIME:	<input type="text"/>
MESSAGE:	<input type="text"/>
<b>F12=DISPLAY F3=EXIT F5=REFRESH</b>	

- When user presses **F12** AID key, the Date and Time should be displayed.



**Hint:** Use the ABSTIME and the FORMATTIME function.

- If user presses **F3** AID key, session should get terminated.
- If user presses **F5** AID key, screen should get refreshed.
- If user presses any other key, screen should display the appropriate error message.

## Lab 4. File Handling

<b>Goals</b>	• Getting familiar with the programs based on file handling concepts.
<b>Time</b>	8 hours

### Assignments 1:

Refer to shared file named 'File Read Prog' for CICS-COBOL coding for business logic

- The participants have to analyze the program, structure the same as per the coding standards, remove the error and successfully execute the program.
- The participants have to document the errors in the excel sheet along with steps taken to resolve the errors.
- 'EMPS' is a transaction used to return information pertaining to an employee when the "EMPID" is entered on the screen.
- Create VSAM Cluster along with the employees data.
- The map as shown below and the working storage section of the emp-info are given for your reference. Code BMS file for the given map.
- If the employee id is found the information has to be sent to the screen (Status field) with the message "Emp Id: XXX found."
- If the emp-id key is not found then status field should array the message "Key not found." and the 'EMP ID' field should be set to bright.
- If the Exit option is set to "Y" then the task has to terminated.

Structure of the VSAM KSDS dataset.

Working-Storage Section.

```

01 EMP-IOAREA.
    05 EMP-REC.
        10 EMP-KEY PIC XXX.
        10 EMP-NAME PIC X(32).
        10 EMP-SEX PIC X.
        10 EMP-DEPT PIC X(10)
        10 EMP-DESIG PIC X(5).
        10 EMP-SAL PIC 9(7).

```

EMPLOYEE INFORMATION FORM	
<b>Employee ID :</b>	
<b>Employee Name :</b>	
<b>Employee Designation:</b>	
<b>Sex:</b>	
<b>Department :</b>	
<b>Salary :</b>	
<b>Status :</b>	
<b>EXIT : X</b>	

**Assignments 2:**

1. Write a program for Customer File Maintenance Module.

- Type the customer code on the **Key** screen.
- If customer code is not present in the data file, then perform the following to add the record:
  - Display the **DETAIL** screen with the following operator message:  
**F2-ADD, F3-KEY-SCREEN, F5-REFRESH**
  - Unprotect all the fields except the customer code field, with the cursor on the Name field.
  - Key in data in all fields, and validate as follows:
    - If any of the fields is blank, position the cursor on the invalid field and display appropriate error message.
    - If all the data is valid, press the **F2** key to add the record in the dataset and the control should come back to the Key screen.
    - After the control comes to the Key screen, the Message 'Record added' should appear in the **Message** field of the Key screen.
    - In case the user presses the **F3** key, no action should be taken. The message 'No record added' should appear in the Message field of the Key screen.
- In case the record exists in the file, display following Operator Message on the Detail screen.  
**F2-MODIFY, F4-DELETE, F3-KEY-SCREEN, F5-REFRESH**
  - To modify the record, key in data in all fields (except customer code), and validate as follows:
    - If any of the fields is blank, position the cursor on the invalid field and display appropriate error message.
    - If all the data is valid, press the F2 key to modify the record in the dataset and the control should come back to the Key screen.
    - After the control comes to the Key screen, the Message 'Record Modified' should be displayed in the Message field of the Key screen.
    - In case the user presses the **F3** key, no action should be taken. The message 'No record Modified' should be displayed in the Message field of the Key screen.
  - To delete the record, press the **F4** key.

- The record should get deleted, and the control should go back to the Key screen with the appropriate message.
  
- After processing is done for add / modify / delete, the Key screen should be displayed. The customer code, which got added / modified / deleted, should get displayed on the Key screen with appropriate message 'Record added / Record modified / Record deleted'.
- For any map, if the user presses a wrong key, then display INVALID KEY message in the error-message box.
- Make use of some variable of WS-COMM-AREA, to determine whether to process KEY map, ADD map, or MODIFY map.
- Make use of sections while writing program code.
- Perform all necessary fields validations before proceeding further.

## Lab 5. Analysis, Enhancement and Debugging Assignments

<b>Goals</b>	Solve Analysis, Enhancement and Debugging Assignments-Case Study 1
<b>Time</b>	8 Hrs

### Case Study 1: Voting Eligibility

**Solve the given problems based on the case study referred in Lab 1 i.e. Voting Eligibility.**

#### Problem 1:

Refer to shared program named NYPROJ.txt for CICS-COBOL partial coding.

- The participants have to analyze the program, remove the error and successfully execute the program.
- In order to make it modular, divide the business logic into subprograms.
- The participants have to document the error in the excel sheet along with steps taken to resolve the errors.
- Wherever required, enhance the given program to fulfill the given requirements by developing the correct business logic.
- In the shared program, authorized user credentials are hard coded. Create a separate dataset which will take care of user credentials and refer to the same in the code as a good practice.
- Some of the constraints that need to take care are:
  - If the person is dead, then the respective record in the VOTER register will be deleted only if that entry is present in voter list and will be updated in the DEATH file.
  - The status is given whether the person is eligible for the voter registry.
  - If the person is alive and his age is greater than 18yrs, then that person is eligible to be in voter list.
  - If he/she is dead or not having the required age then respective message is displayed.
  - The details of the people are taken and are updated in the corresponding birth and death files.
  - If the person is eligible then those are updated in the voter registry.

#### Stretched Assignment based on the given Case Study:

Do implement COBOL Batch applications to generate the given reports.

- Generate BIRTH certificate with unique SSN.
- Accept the SSN, checks eligibility criteria to vote and generate voter-ID. Sample Report for Voter Identity Card is provided for your reference.

GOVERNMENT OF INDIA

\*\*\*\*\* VOTER IDENTITY CARD \*\*\*\*\*

CONSTITUENCY:TIRUPATI SSN NO:0002ABTI1989

ELECTORS NAME : ABHILASH B

SEX : M

DATE OF BIRTH : 1989-07-15

AGE : 23

NAME OF THE FATHER : B VENKATA NARAYANA

ADDRESS : FLAT NO: 502,MRVS APARTMENT,KBLAYOUT,TPT

REGISTERED ON (YYYY-DD-MM) : 2012-07-02

SIGNATURE OF THE REGISTRAR

## Lab 6. Analysis, Enhancement and Debugging Assignments

<b>Goals</b>	Solve Analysis, Enhancement and Debugging Assignments-Case Studies
<b>Time</b>	4 Hrs

### Case Study: The Census management system

The Census management system will be used to produce statistics that are relevant to data users and that every action in a census must be directed towards producing relevant output that meets the needs of public.

#### Functional components of the case study:

Following is a list of functionalities of the system. Wherever, the description of functionality is not adequate; you can make appropriate assumptions and proceed.

- ◆ Data processing ( Data capture, Data editing, Coding, Tabulation)
- ◆ Dissemination (publication of printed tables and reports, dissemination on computer media, on-line dissemination)
- ◆ Evaluation (demographic analysis for census evaluation, post-enumeration survey)
- ◆ Analysis of the results
- ◆ Publication of census results (descriptive reports, basic statistical reports)

Refer to shared files for BMS coding as well for COBOL Coding.

- The participants have to analyze the program, remove the error and successfully execute the program.
- File named **census.cbl** contains incomplete COBOL coding which you need to understand and complete as per the requirements of the system
- The participants have to document the error in the excel sheet along with steps taken to resolve the errors.

### Case Study: Insurance Policy Processing System (IPPS)

Refer to shared file 'INSURANCE POLICY PROCESSING SYSTEM' for the problem.

#### Online Processing:

This is the initial step which will allow the Agent/Customer to create a new policy or the customer can edit his existing policy. CICS would be used to create this OLTP (Online Transaction Processing) System

Refer to the file structure and all other required details used in familiarization to MF course.

## Lab 7. Browse records

Goals	• Write program for browsing records of the customer file
Time	3 Hrs

### 5.1: Write a program for Customer Inquiry Module.

#### Step 1: Write a program for Customer Inquiry Module.

1. Display the DETAIL screen.
2. Allow the user to enter specific CUST-CODE or go for various AID keys.
3. Provide the following keys to the user to perform an inquiry:
  - ENTER-PROCESS
  - F1-FIRST
  - F2-LAST
  - F7-PREVIOUS
  - F8-NEXT
  - F5-REFRESH
  - F3-EXIT
4. Display appropriate error messages in the error message box.

## Lab 8. Temporary Storage Queue

Goals	<ul style="list-style-type: none"><li>To create a TSQ and use it in modification</li></ul>
Time	2 hours

### 6.1: Using TSQ in the MODIFY routine

Use TSQ in the MODIFY routine (Assignment of Lab 4) to determine whether record has been changed during the current task's pseudo-conversational cycle. For this use the following procedure:

**Step 1:** READ the record for specified customer code.

**Step 2:** If record is found, store it in a TSQ.

**Step 3:** Send the DETAIL map to perform MODIFY or DELETE routine.

**Step 4:** For processing detail screen for modification, perform read with update. After this I/O, again read the record with UPDATE option and now compare this record, which is read, and the one, which is there in the TSQ. If both are same, it implies that no other task has modified that record. So continue processing. If the comparison fails, then display the message "Record has already been modified by someone. Enter the changes again".

## Lab 9. Menu Handling

Goals	<ul style="list-style-type: none"> <li>To create an Integrated Menu driven application</li> </ul>
Time	3 Hours

### 7.1: Creating a Menu driven application

**Step 1:** Integrate Assignments of lab 1, lab 4, lab 5 to create a Menu driven application.

- Accept the choice from the MENU program.
  - In case the operator selects option 1, pass control to the Maintenance (Assignment part IV) program, which will perform File Maintenance. After exiting from the File Maintenance program, the control should come back to the Menu program.
  - In case the operator selects option 2, pass control to the Inquiry (Assignment 5) program, which will perform the Inquiry. After exiting from the Inquiry program, the control should come back to the Menu program.
  - Ensure that Assignment Part IV and Assignment Part V get executed only through the Menu program. It should not get invoked through transaction code.

**Note:** Use XCTL to invoke Assignment part IV and Assignment V.



#### Hint: General Guidelines for all Assignments:

- Check for EIBCALEN=0 for invalid entry to INQUIRY and MAINTENANCE program. (The entry should be always from MENU program).
- Check the CA-PROCESS-SW in the main section to branch out to different sections.
- Use “GO TO” to take control to the EXIT paragraph in the section. (Code EXIT at the end of every section).
- Code CICS command and the associated HANDLE CONDITION in the same section.
- All working-storage variables must be initialized.
- Each section must be preceded with proper documentation.

## Lab 10. Bank Application

Goals	<ul style="list-style-type: none"> <li>To create an Integrated Bank application</li> </ul>
Time	4 Hours

### 1. Preface

The purpose of this document is to provide specification description of the module exercise & document the project results on completion.

### 2. Project Requirements

ABC Bank of India wants to create an application to automate their Business. This application should allow them to maintain their customer information and daily transactions of debit or credit. Application needs to be created in CICS and VSAM.

### 3. Design Details

#### 3.1 File Layout

##### ❖ Input/Output:

Following files needs to be created

File1: ABCCUST – To store the customer information

Field	Type	Length	Value/Format/Criteria
ABCCUST-ACCOUNT-NO	X	6	VSAM Key
ABCCUST-ACC-TYPE	X	1	S-Savings, C-Current
ABCCUST-NAME	X	20	Customer Name
ABCCUST-DOB	X	10	YYYY-MM-DD – Date of birth
ABCCUST-ADDRESS	X	30	Includes Address, City, State
ABCCUST-PHONE	X	10	
ABCCUST-JOIN-DATE	X	10	YYYY-MM-DD
ABCCUST-BALANCE	9	7v99	Comp-3 – Current Balance
Filler	X	18	
Total record length		110	

File2: ABCTRAN – To store the daily transaction

Field	Type	Length	Value/Format/Criteria
ABCTRAN-DATE	X	8	VSAM Key – YYYY-MM-DD
ABCTRAN-TRAN-NO	X	12	VSAM Key – Account number + Current time HHMMSS
ABCTRAN-TRAN-TYPE	X	1	C – Credit, D – Debit

ABCTRAN-AMOUNT	9	7V99	COMP-3
Filler	X	24	
Total record length		50	

Note: Key of the file is combination of date and transaction number. Transaction number is created based on the xxxxxxHHMMSS (Account number x(06) and current time stamp X(06))

### 3.2 Screen Design

Screen 1: This screen is the main screen and customer entry/view and updates are managed in this screen. All the fields are mandatory except Balance field. Balance field is for view only

<u>ABC BANK OF INDIA</u> <u>CUSTOMER ENTRY SCREEN</u>			
Account No : _____	Account Type : _____		
Name : _____	Date of Birth : _____-____-____		
Address : _____			
Phone : _____			
Joining Date : _____-____-____			
Balance : 0000000.00			
Message: _____			
PF2: REFRESH      PF3: EXIT      PF4: SAVE      PF5: UPDATE PF12: DELETE PF13 – TRANS SCREEN      PF14-REPORT1      PF15-REPORT2			

Screen 2: This screen can be initiated from Customer entry screen by pressing PF13. This screen is used to enter the transactions (Credit or Debit)

### ABC BANK OF INDIA

## TRANSACTION ENTRY SCREEN

Account No : \_\_\_\_\_

Transaction type : \_

Amount : 000000.00

Current Balance : 0000000.00

Message:

---



---

**PF2: REFRESH  
PF13 – CUST SCREEN**

**PF3: EXIT  
PF14-REPORT1**

**PF4: SAVE  
PF15-REPORT2**

### 3.3 Report Layout

Report Screen 1:

## ABC BANK OF INDIA Daily Report Screen

Date : \_\_\_\_-\_\_\_\_-

Account No	Transaction Time	Transaction Type	Amount
------------	------------------	------------------	--------

XXXXXX	XX:XX:XX	X
000000.00		
XXXXXX	XX:XX:XX	X
000000.00		
XXXXXX	XX:XX:XX	X
000000.00		
XXXXXX	XX:XX:XX	X
000000.00		
XXXXXX	XX:XX:XX	X

## Message:

**PF2: REFRESH**      **PF3: EXIT**      **PF7: UP**      **PF8: DOWN**  
**PF13 – CUST SCREEN**      **PF14-TRANS SCREEN**      **PF15-REPORT2**

## Report Screen 2:

## ABC BANK OF INDIA

### Customer Report Screen

Account No : \_\_\_\_\_

Customer Name : **Current Balance : 0000000.00**  
Date              Transaction              Transaction              Amount  
                    Time                        Type

## Message:

## PF2: REFRESH Down

# **PF13 – CUST SCREEN REPORT1**

PF3: EXIT

**PF14-TRANS SCREEN**

### Pseudo code

#### Screen 1: - Program 1: Customer Entry screen

1. User enters Account number and presses ENTER key: Account information needs to be displayed based on the customer data available in the File.
  - a. If the customer information NOT FOUND then allow the user to type the rest of the information.
  - b. If the customer information FOUND then allow the user to change the details, (Except Balance) based on PF5 Key press update the changes. Based on PF12 Key press delete the Account details
2. Enter all the information (Except Balance field – Protected) and press PF4: Insert the record into ABCCUST file. If the account number already exists, display the error message
3. PF13 Key press: Transfer the control to program 2 (Transaction Entry)
4. PF14 Key press: Transfer the control to Report program 1 (Daily report screen)
5. PF15 Key press: Transfer the control to Report program 2 (Customer report screen)

#### Screen 2: - Program 2: Transaction Entry screen

1. User enters Account number and presses ENTER key: Account Balance will be displayed in the protected field.
2. Enter all the information (Except Balance field – Protected) and press PF4: Insert the record into ABCTRAN file. If the account number not exists, display the error message
3. PF13 Key press: Transfer the control to program 1 (Customer Entry)
4. PF14 Key press: Transfer the control to Report program 1 (Daily report screen)
5. PF15 Key press: Transfer the control to Report program 2 (Customer report screen)

#### Report 1: - Program 3: Daily Report screen

1. User enters Date and presses ENTER key: All the transaction belongs to that date needs to be displayed.
  - a. Start the ABCTRAN file using key (Date + low values) and read all the records and write into TSQ till the date ends.
  - b. From the TSQ read the records and display into Screen.
  - c. If the date is changed then delete the TSQ and reload the new records
  - d. If no records are available for that date, display error message
2. PF7 – Read the previous page of records from TSQ and display
3. PF8 – Read the next page of records from TSQ and display
4. PF13 Key press: Transfer the control to program 1 (Customer Entry)
6. PF14 Key press: Transfer the control to Program 2 (Transaction Entry)
7. PF15 Key press: Transfer the control to Report program 2 (Customer report screen)

#### Report 2: - Program 4: Customer Report screen

1. User enters Customer number and presses ENTER key: Customer information and all the transaction belongs to that customer needs to be displayed.
  - a. Read the ABCCUST file and based on the customer number and display to the screen (Save this to Commarea).
  - b. Start the ABCTRAN file using key (low values) and read all the records, and write the matching customer records into TSQ till the file ends.
  - c. From the TSQ read the records and display into Screen.
  - d. If the Customer number is changed then delete the TSQ and reload the new records
  - e. If no records are available for that Customer, display error message

5. PF7 – Read the previous page of records from TSQ and display
6. PF8 – Read the next page of records from TSQ and display
7. PF13 Key press: Transfer the control to program 1 (Customer Entry)
2. PF14 Key press: Transfer the control to Program 2 (Transaction Entry)
3. PF15 Key press: Transfer the control to Report program 1 (Daily report screen)

## Appendices

### [Appendix A: CEBR, CECI, CEMT, CEDF](#)

#### [Appendix A-1: CEBR:](#)

The **Temporary Storage Browse (CEBR)** is a CICS supplied transaction, which browses **Temporary Storage Queue (TSQ)**. It is a convenient tool if you need to display the content to TSQ when you are monitoring an application program through EDF.

Invoking CEBR is simple. Type **CEBR**, and press **Enter** key. Then you can type the command **QUEUE <TSQ Name>**, and press **Enter** key. The content of the TSQ will be displayed.

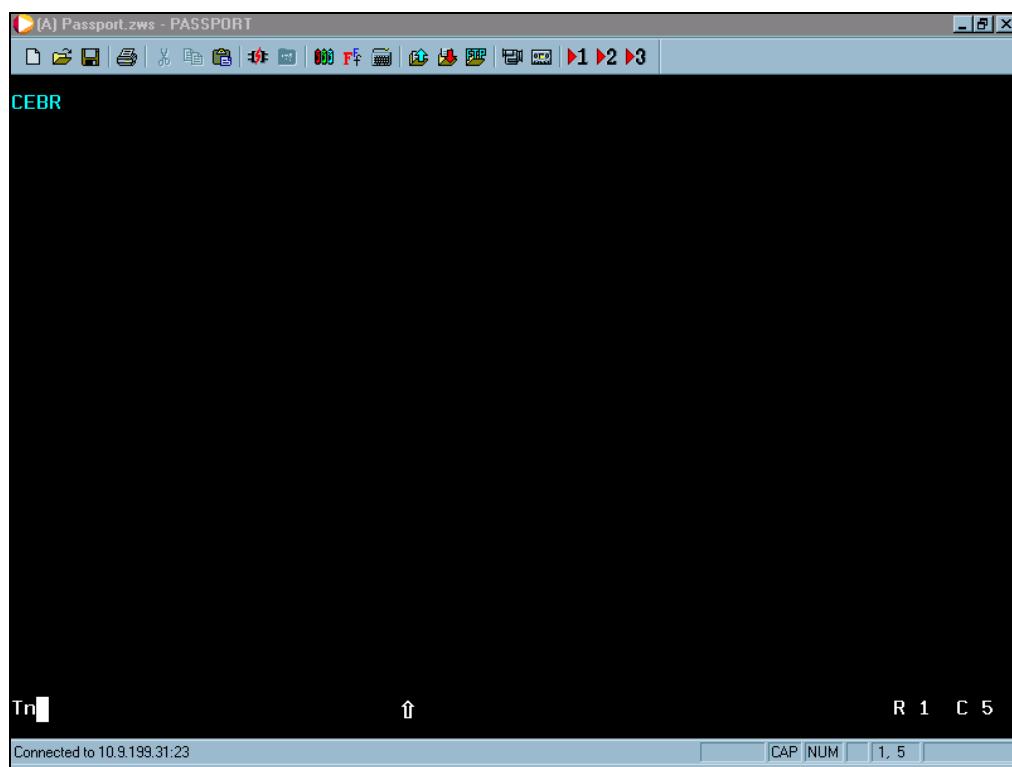
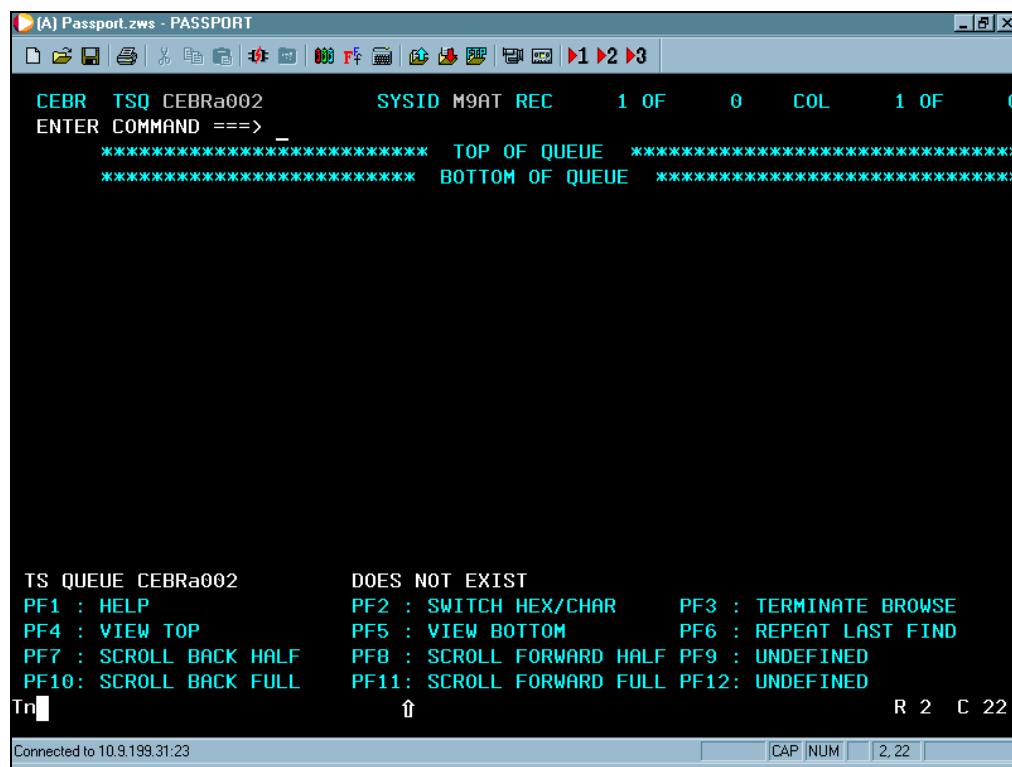
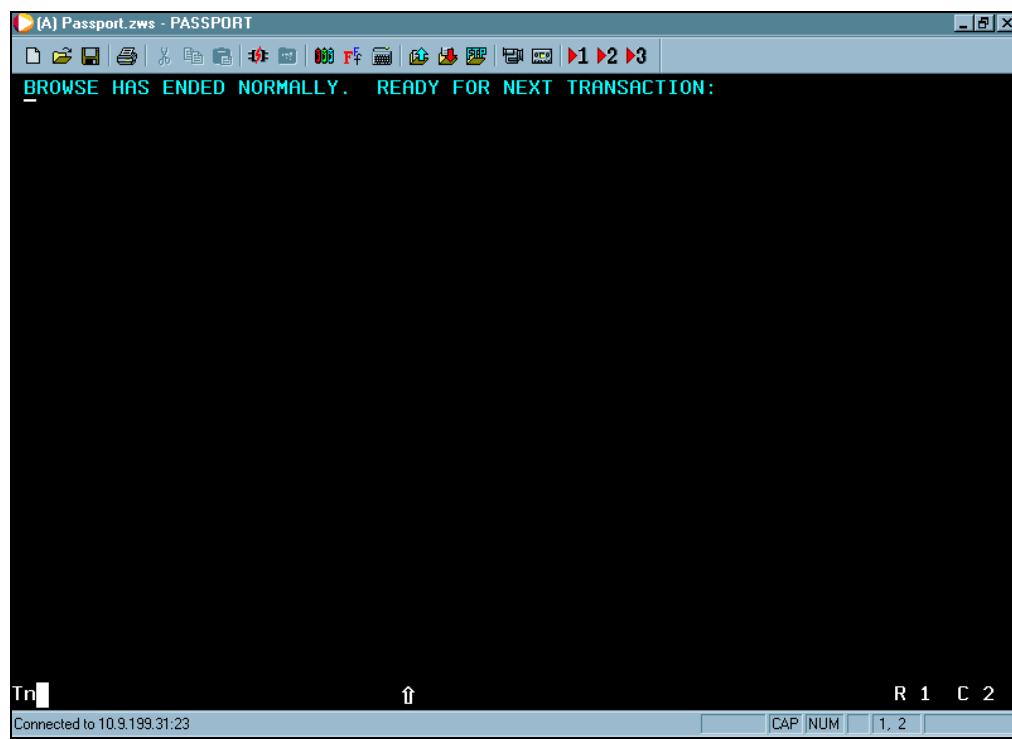


Figure 1: CEBR



**Figure 2: CEBR**

Press **PF3** key to end the session.



**Figure 3: Browse ended message**

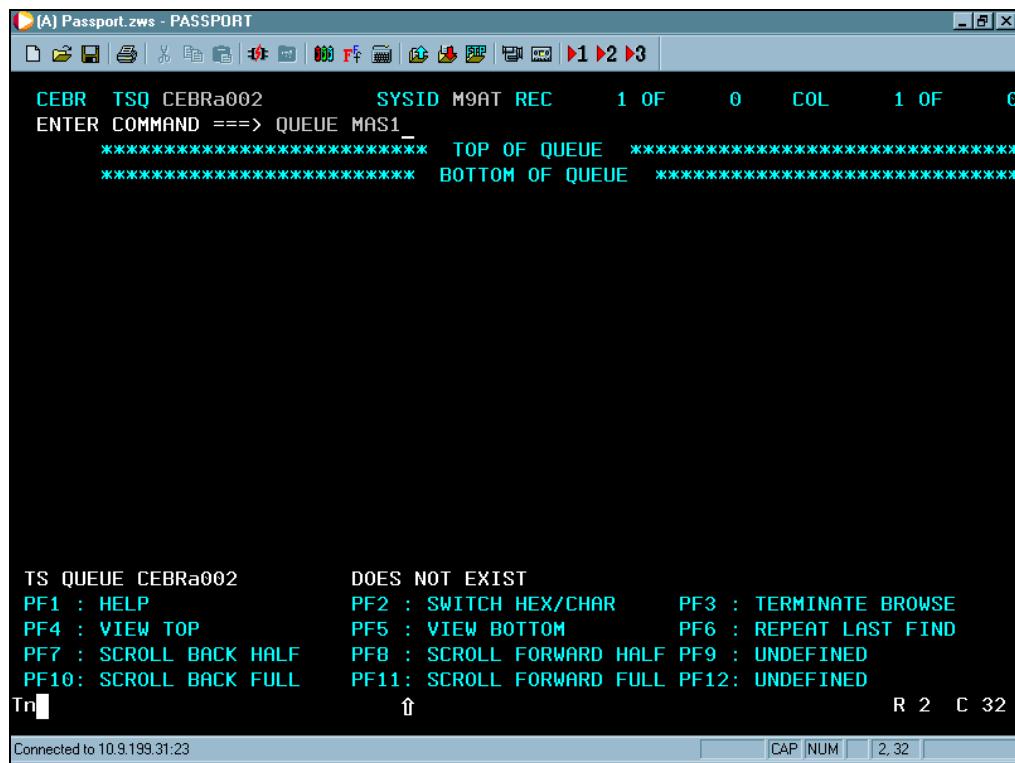


Figure 4: CEBR

## Appendix A-2: CECI

**CECI** is **Command Level Interpreter (CECI)**. It is a CICS-supplied transaction which performs interactive patching into application System.

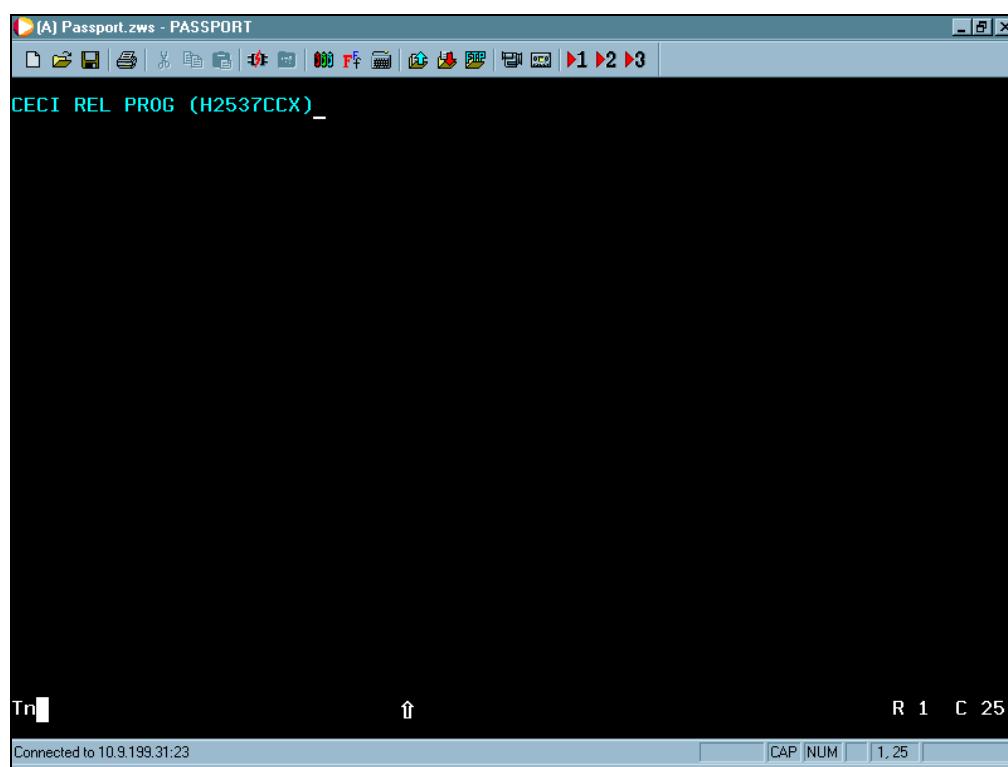
In CICS, you need to expedite modules. These can be instances wherein the module gets trapped and even after ending the expeditor session the module is not released.

**Objective:** To Release a Program:

**Step 1:** Type the command **CECI REL PROG (H2537QAC)**

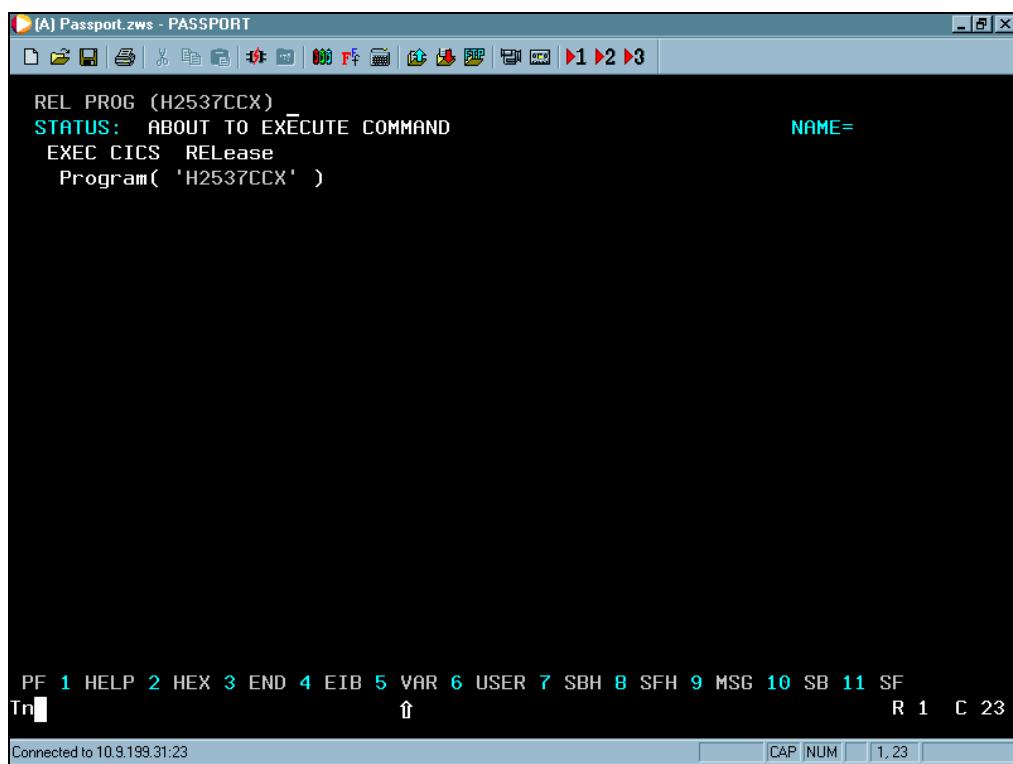
H2537QAC is the module name.

Argument used for module is PROG (Program).



**Figure 5: Typing the command**

Output is as shown below:



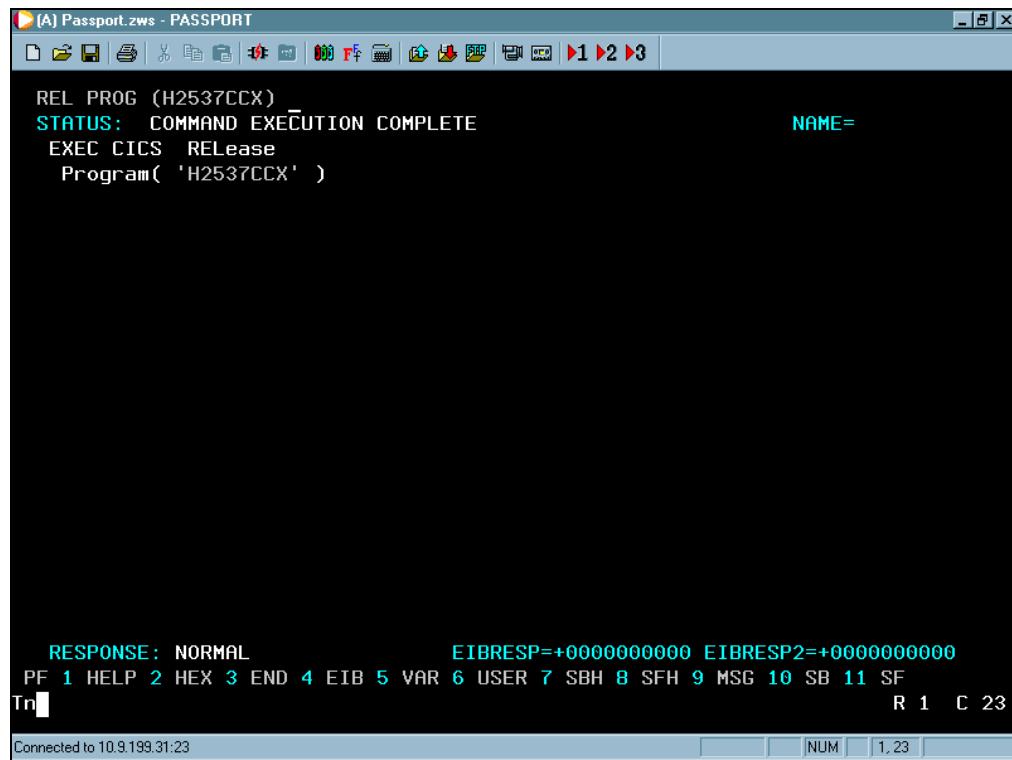
The screenshot shows a terminal window titled '(A) Passport.zws - PASSPORT'. The window contains the following text:

```
REL PROG (H2537CCX)
STATUS: ABOUT TO EXECUTE COMMAND
EXEC CICS RELEase
Program( 'H2537CCX' )
```

At the bottom of the window, there is a function key bar with labels: PF 1 HELP 2 HEX 3 END 4 EIB 5 VAR 6 USER 7 SBH 8 SFH 9 MSG 10 SB 11 SF. Below the bar, it says Tn [ ] ↑ R 1 C 23. At the very bottom, it shows 'Connected to 10.9.199.31:23' and 'CAP NUM 1,23'.

**Figure 6: Output**

Step 2: Press **Enter** key to release the program.



```
REL PROG (H2537CCX)
STATUS: COMMAND EXECUTION COMPLETE
EXEC CICS RELEaSe
Program( 'H2537CCX' )

NAME=


RESPONSE: NORMAL          EIBRESP=+0000000000 EIBRESP2=+0000000000
PF 1 HELP 2 HEX 3 END 4 EIB 5 VAR 6 USER 7 SBH 8 SFH 9 MSG 10 SB 11 SF
Tn R 1 C 23
Connected to 10.9.199.31:23
```

**Figure 7: Response: Normal**

If the program is already released, the response is INVREQ as you can see in the screen shot shown below:

The screenshot shows a CICS terminal window titled '(A) Passport.zws - PASSPORT'. The terminal displays the following text:

```
REL PROG (H2537CCX)
STATUS: COMMAND EXECUTION COMPLETE
EXEC CICS RELEASE
Program( 'H2537CCX' )

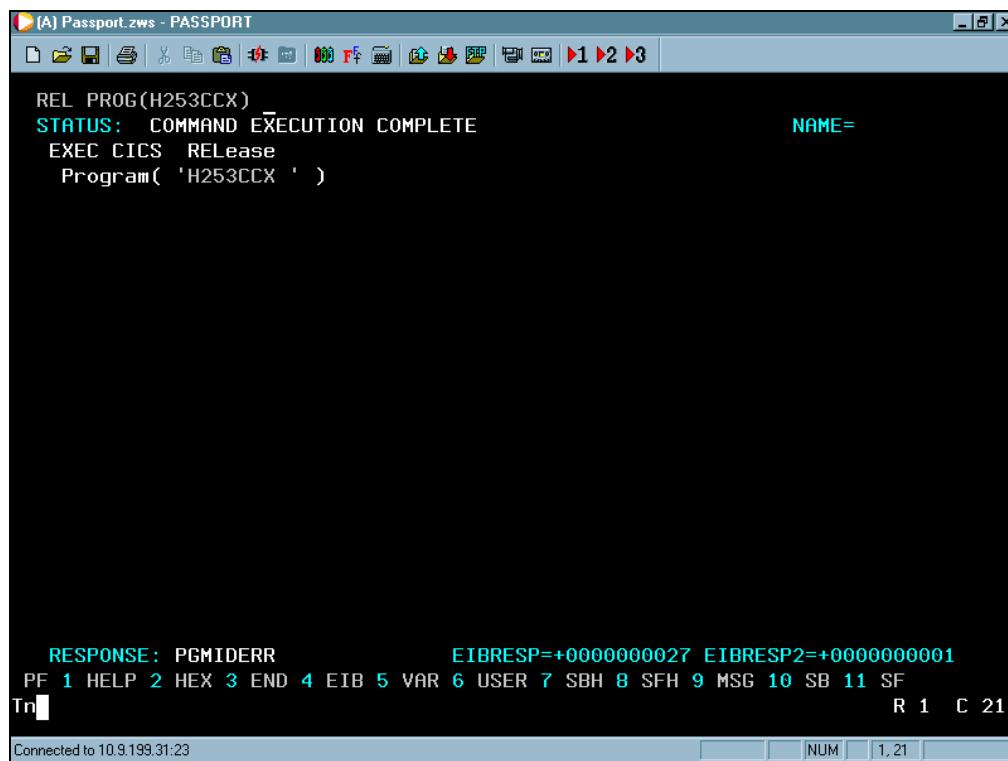
RESPONSE: INVREQ
PF 1 HELP 2 HEX 3 END 4 EIB 5 VAR 6 USER 7 SBH 8 SFH 9 MSG 10 SB 11 SF
Tn[ ] R 1 C 23

EIBRESP=+0000000016 EIBRESP2=+0000000006
```

The bottom status bar indicates the connection is 'Connected to 10.9.199.31:23' and shows the cursor position at 'R 1 C 23'.

Figure 8: Response: INVREQ

Also if the program name is not valid, Response is PGMDERR. It means that the module name entered is invalid or such a module does not exist.



```

(A) Passport.zws - PASSPORT
REL PROG(H253CCX)
STATUS: COMMAND EXECUTION COMPLETE
EXEC CICS RELEASE
Program( 'H253CCX' )

NAME=


RESPONSE: PGMDERR
EIBRESP=+0000000027 EIBRESP2=+0000000001
PF 1 HELP 2 HEX 3 END 4 EIB 5 VAR 6 USER 7 SBH 8 SFH 9 MSG 10 SB 11 SF
Tn R 1 C 21
Connected to 10.9.199.31:23
[ ] [ ] NUM [ ] 1,21 [ ]

```

**Figure 9: Response: PGMDERR**

### Appendix A-3: CEMT

**CEMT (Enhanced Master Terminal)** transaction is a CICS supplied transaction, which manipulates the CICS environment, such as transactions, programs, files, TAQs, and tasks.

#### Major functions:

- **INQUIRE:** To inquire about the status of CICS environments
- **SET:** To update the status of the CICS environments
- **PERFORM:** For further system operations

**Example:**

**Objective:** Close a file in CICS environment.

**Steps:**

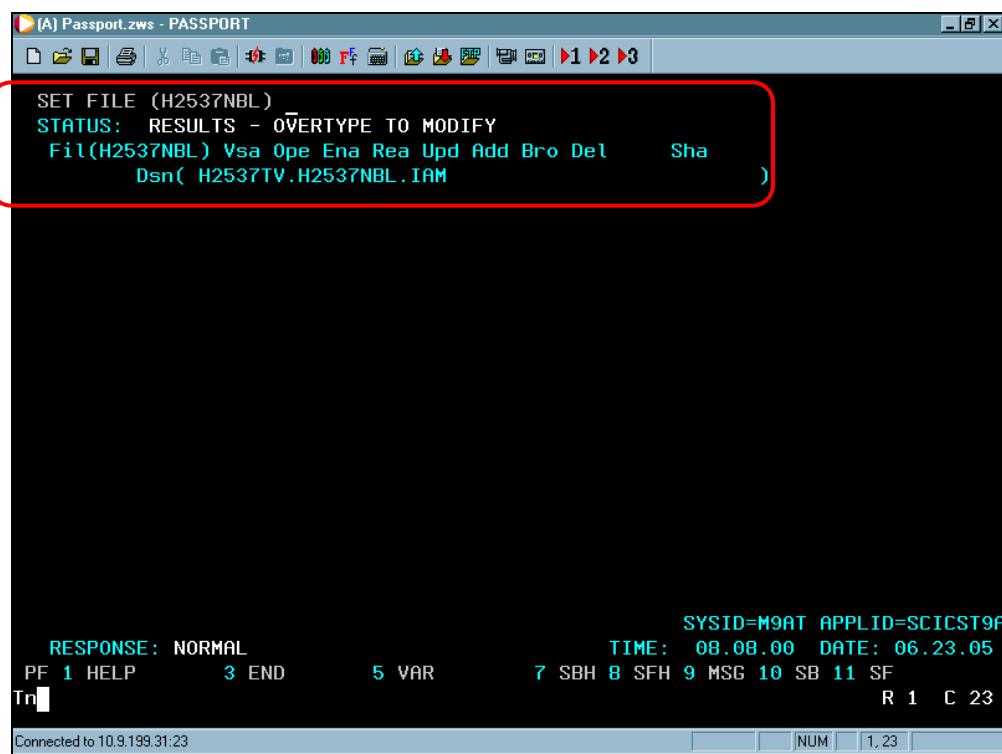
**Step 1:** Type Command: **CEMT SET FILE (H2537NBL)**



The screenshot shows a terminal window titled '(A) Passport.zws - PASSPORT'. The window contains a black text area where the command 'CEMT SET FILE (H2537NBL)' is being typed. The window has a standard Windows-style title bar and a toolbar at the top. At the bottom, there is a status bar displaying 'Connected to 10.9.199.31:23', 'R 1 C 25', and a cursor position of '1, 25'.

**Figure 10: Typing the command**

Output is as shown below:



```
SET FILE (H2537NBL)
STATUS: RESULTS - OVERTYPE TO MODIFY
Fil(H2537NBL) Vsa Ope Ena Rea Upd Add Bro Del     Sha
              Dsn( H2537TV.H2537NBL.IAM )
```

RESPONSE: NORMAL

PF 1 HELP 3 END 5 VAR 7 SBH 8 SFH 9 MSG 10 SB 11 SF

Tn R 1 E 23

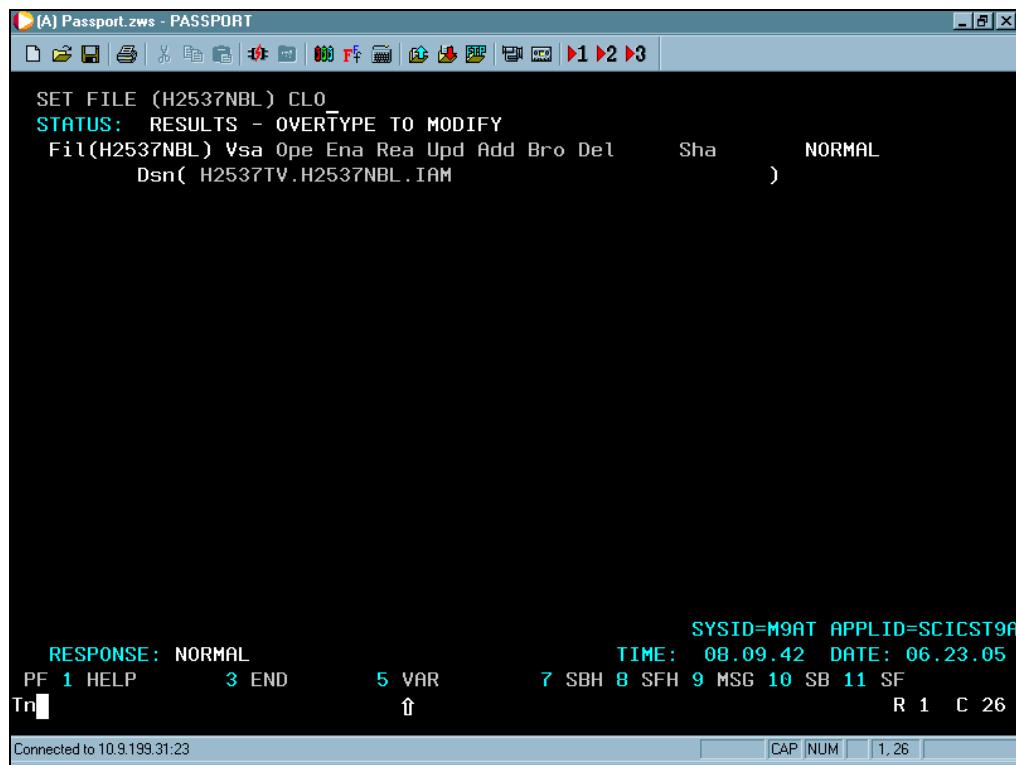
SYSID=M9AT APPLID=SCICST9A

TIME: 08.08.00 DATE: 06.23.05

Connected to 10.9.199.31:23

Figure 11: Output

**Step 2:** Type **Clo** and press **Enter** key.



```
SET FILE (H2537NBL) CLO.
STATUS: RESULTS - OVERTYPE TO MODIFY
Fil(H2537NBL) Vsa Ope Ena Rea Upd Add Bro Del      Sha      NORMAL
Dsn( H2537TV.H2537NBL.IAM )
```

RESPONSE: NORMAL SYSID=M9AT APPLID=SCICST9A  
PF 1 HELP 3 END 5 VAR TIME: 08.09.42 DATE: 06.23.05  
Tn 7 SBH 8 SFH 9 MSG 10 SB 11 SF R 1 C 26

Connected to 10.9.199.31:23 CAP NUM 1,26

Figure 12: Typing Command

Output is as shown below:

(A) Passport.zws - PASSPORT

SET FILE (H2537NBL) CLO  
STATUS: RESULTS - OVERTYPE TO MODIFY  
Fil(H2537NBL) Vsa Clo Une Rca Upd Add Bro Del Sha NORMAL  
Dsn( H2537TV.H2537NBL.IAM )

RESPONSE: NORMAL SYSID=M9AT APPLID=SCICST9A  
PF 1 HELP 3 END 5 VAR TIME: 08.10.46 DATE: 06.23.05  
Tn 7 SBH 8 SFH 9 MSG 10 SB 11 SF R 1 C 27

Connected to 10.9.199.31:23

**Figure 13: Output**

As you can see, the file was earlier **Open (Ope)** and now after keying in the command '**Clo**' the status of the file has changed to **Close (Clo)**.

To open the file, issue command '**Ope**' in similar fashion.

End the transaction by pressing **PF3** key.

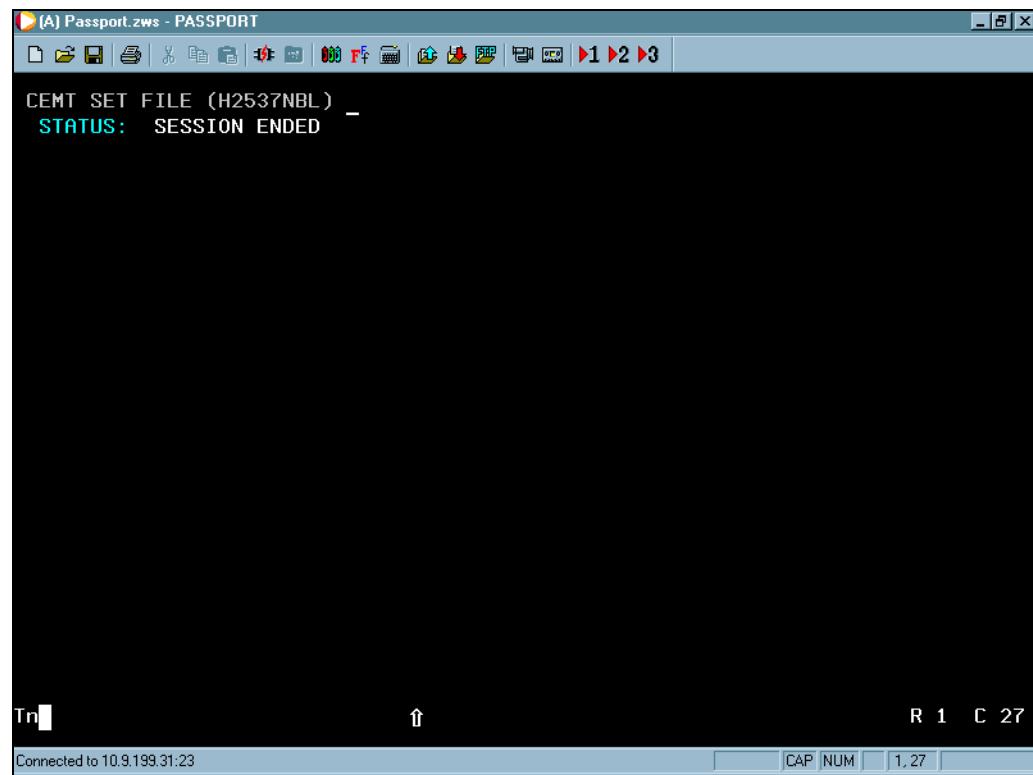


Figure 14: Session ended message

You can see that **Status: Session Ended**.

**Note:** The command is not case sensitive.

Using **CEMT** you can also NEWCOPY module.

Say you have made changes to your module but it has not reflected in the region. You need to do a new copy (this means overlay the previous version).

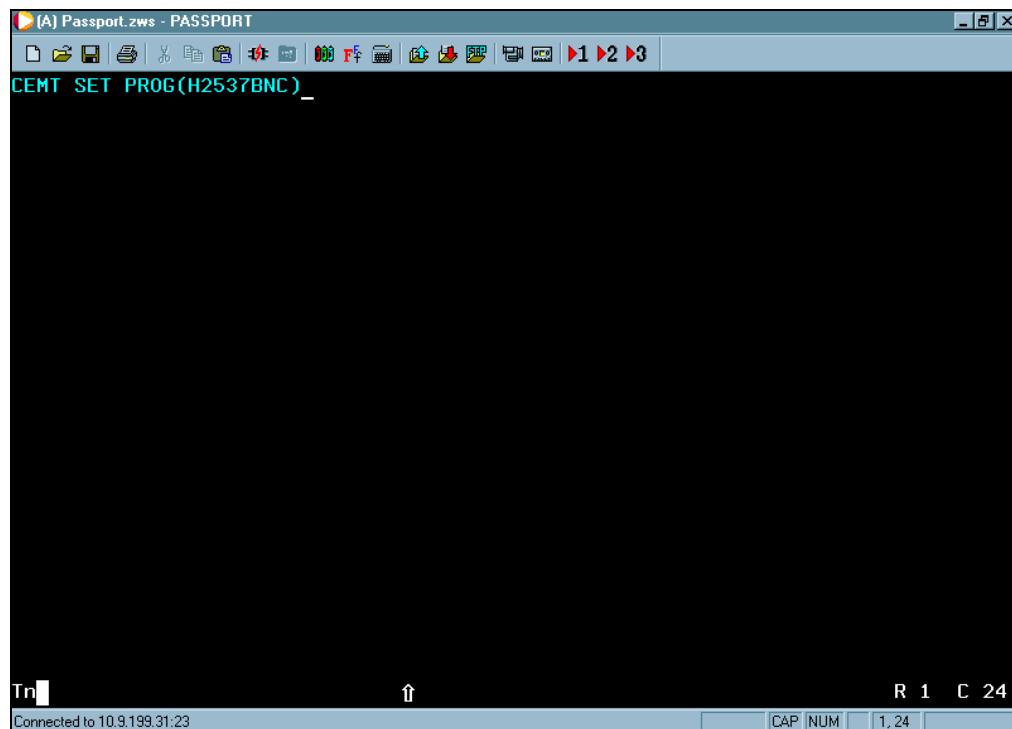


Figure 15: Typing command

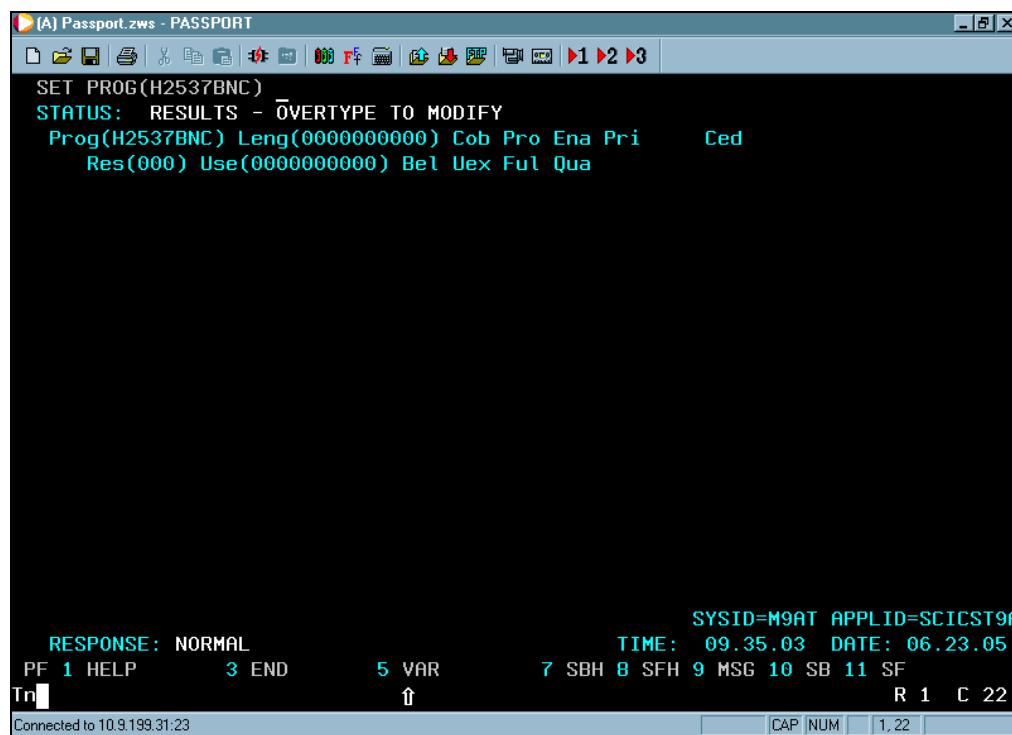
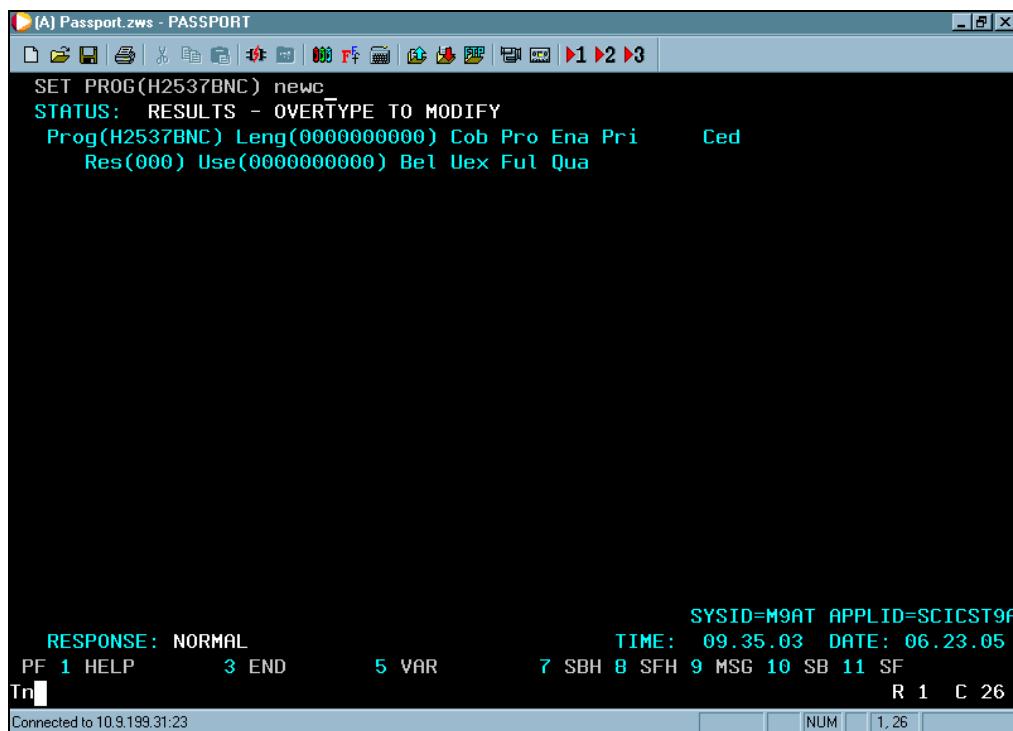


Figure 16: Output

As you can see, the Length is (00000000). This means that the module is not present in the region.



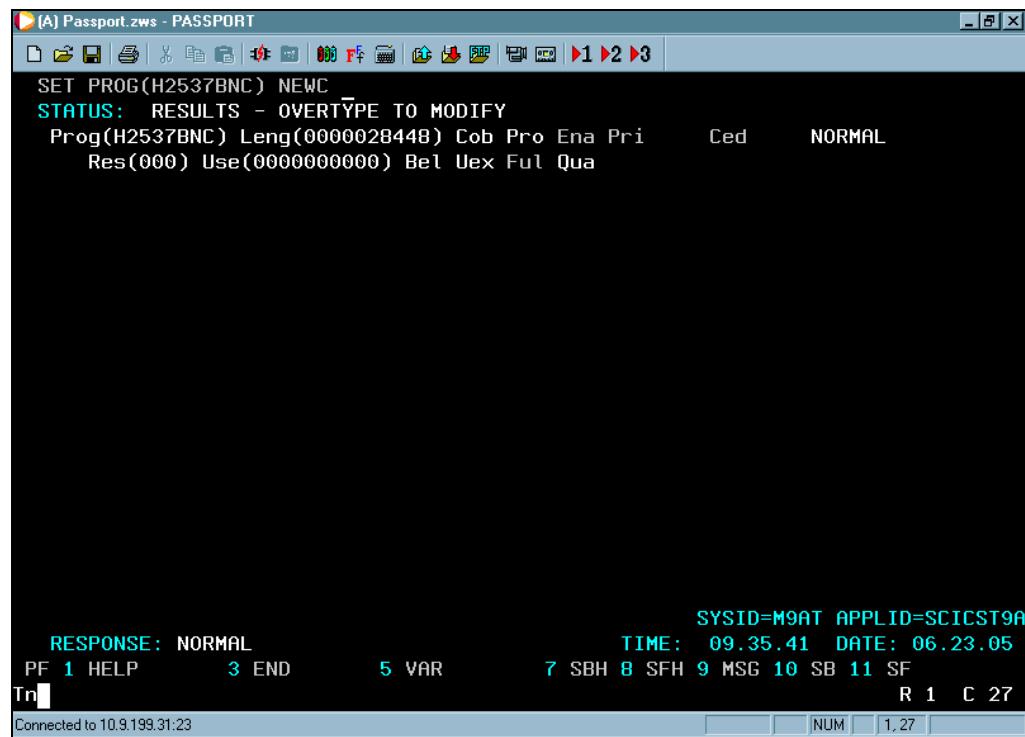
The screenshot shows a CICS terminal window titled '(A) Passport.zws - PASSPORT'. The terminal displays the following text:

```
SET PROG(H2537BNC) newc
STATUS: RESULTS - OVERTYPE TO MODIFY
Prog(H2537BNC) Leng(0000000000) Cob Pro Ena Pri      Ced
             Res(000) Use(0000000000) Bel Uex Ful Qua

RESPONSE: NORMAL                               SYSID=M9AT APPLID=SCICST9A
PF 1 HELP      3 END      5 VAR      7 SBH 8 SFH 9 MSG 10 SB 11 SF
Tn[ ] TIME: 09.35.03 DATE: 06.23.05
Connected to 10.9.199.31:23      R 1   C 26
[ ] [ ] [NUM] [1,26]
```

Figure 17: Output

Suppose you type the command **NEWC**.



```
(A) Passport.zws - PASSPORT
SET PROG(H2537BNC) NEWC
STATUS: RESULTS - OVERTYPE TO MODIFY
Prog(H2537BNC) Leng(0000028448) Cob Pro Ena Pri Ced NORMAL
Res(000) Use(0000000000) Bel Uex Ful Qua

RESPONSE: NORMAL
TIME: 09.35.41 DATE: 06.23.05
SYSID=M9AT APPLID=SCICST9A
PF 1 HELP 3 END 5 VAR 7 SBH 8 SFH 9 MSG 10 SB 11 SF
R 1 C 27

Connected to 10.9.199.31:23
```

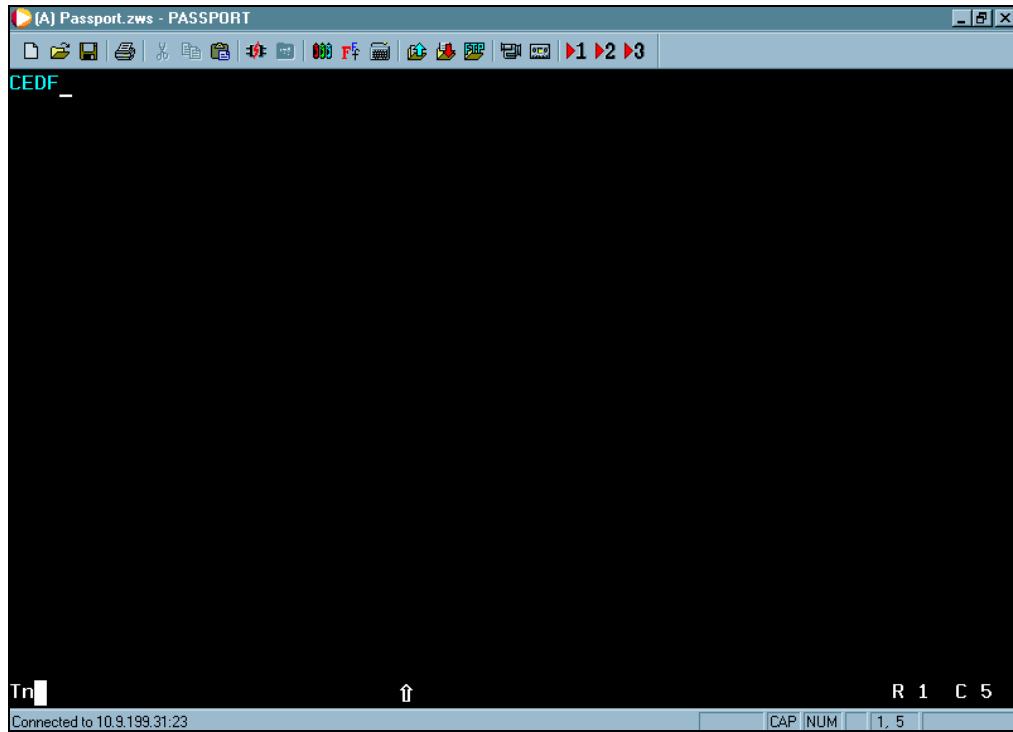
Figure 18: Output

The Length changes indicating that the module is now present in the region.

#### Appendix A-4: CEDF

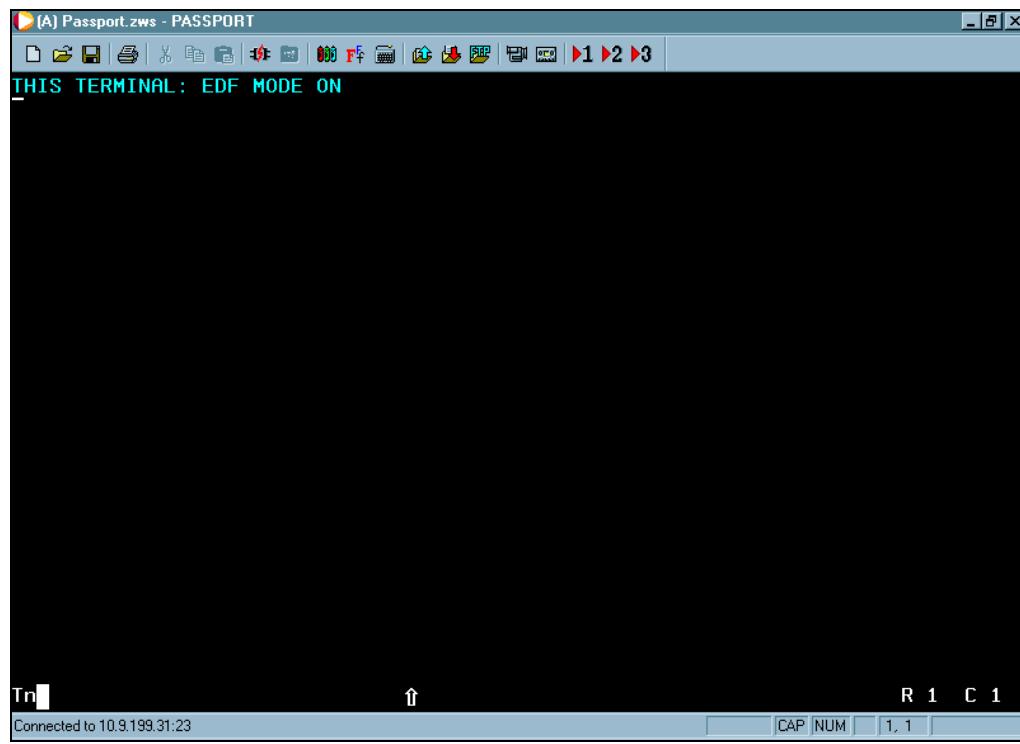
CEDF is the Execution Diagnostic Facility.

**Step 1:** Type **CEDF** on the screen, and press **Enter** key.



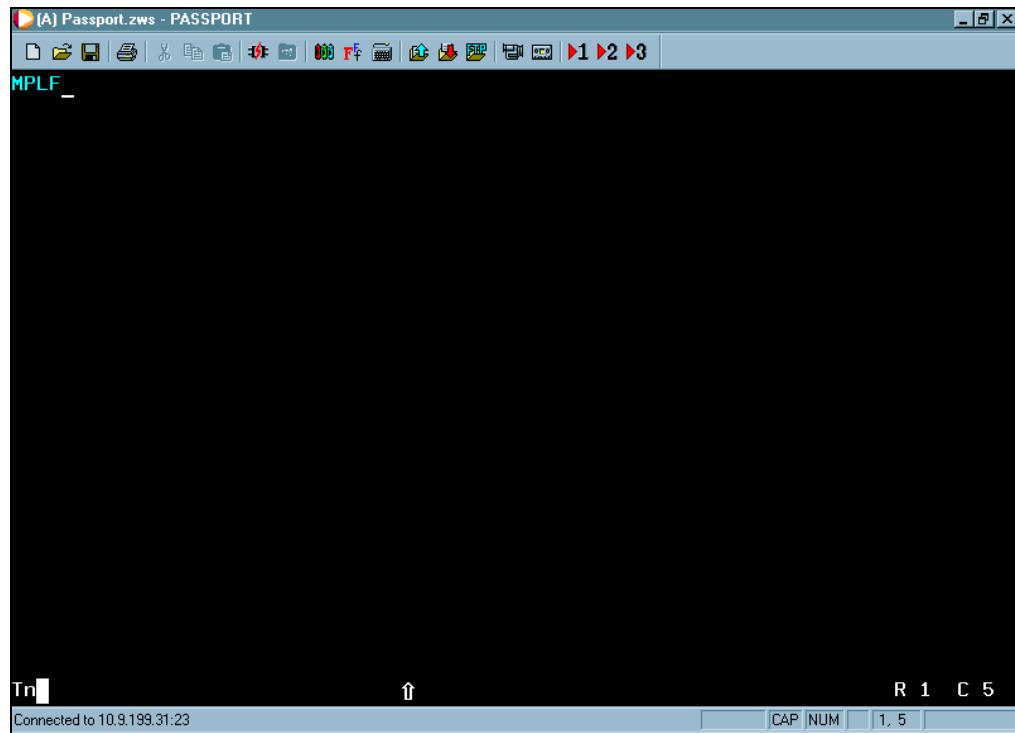
**Figure 19: Typing the CEDF command**

Output will be as shown below.



**Figure 20: Output**

**Step 2:** Type the transaction to spawn, and press **Enter** key.



**Figure 21: Typing the command**

Output will be as shown below:

(A) Passport.zws - PASSPORT

TRANSACTION: MPLF PROGRAM: H2550I01 TASK: 0003966 APPLID: SCICST9A DISPLAY: 00  
STATUS: PROGRAM INITIATION

```

EIBTIME      = 94324
EIBDATE      = 0105174
EIBTRNID     = 'MPLF'
EIBTASKN     = 3966
EIBTRMID     = 'a002'

EIBCPOSN     = 4
EIBCALEN     = 0
EIBAID       = X'7D'                                AT X'001690EA'
EIBFN        = X'0000'                                AT X'001690EB'
EIBRCODE     = X'0000000000000000'                  AT X'001690ED'
EIBDS        = '.....'
+ EIBREQID    = '.....'

ENTER: CONTINUE
PF1 : UNDEFINED          PF2 : SWITCH HEX/CHAR      PF3 : END EDF SESSION
PF4 : SUPPRESS DISPLAYS   PF5 : WORKING STORAGE     PF6 : USER DISPLAY
PF7 : SCROLL BACK         PF8 : SCROLL FORWARD      PF9 : STOP CONDITIONS
PF10: PREVIOUS DISPLAY    PF11: EIB DISPLAY          PF12: UNDEFINED
Tn [ ]                   ⇧                           R 1 C 1
Connected to 10.9.199.31:23
[ ] CAP NUM [ ] 1,1 [ ]

```

Figure 22: Output

## Create Maps using CA-Realia Simulator

### Goals

- Learn to create, edit and compile Maps

### Lab Setup

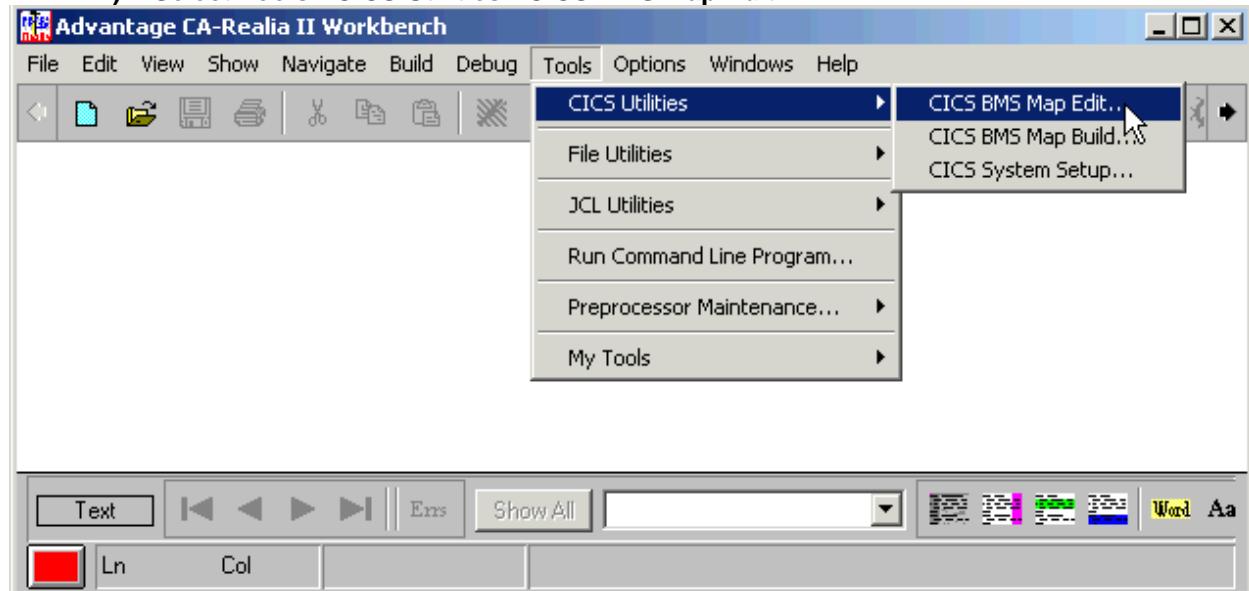
Successful installation of Advantage CA-Realia WorkBench 3.2 IDE

#### I. Start the WorkBench IDE

- Select Start->Programs->Computer Associates->Advantage->CA-Realia->WorkBench ID

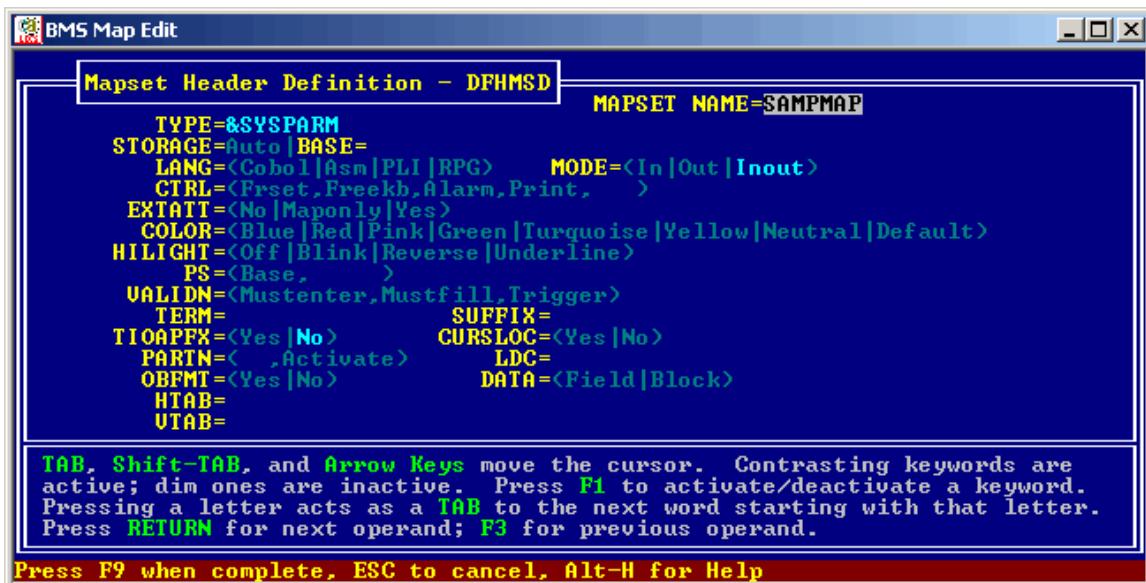
#### II. Create the Map

- Select Tools->CICS Utilities->CICS BMS Map Edit



- Select F4 to Edit the Mapset Header definition

- Navigate using tab keys
- Select TYPE=&SYSPARM,CTRL->Freekb,MODE=Inout



```

Mapset Header Definition - DFHMSD          MAPSET NAME=SAMPMAP
  TYPE=&SYSPARM
  STORAGE=Auto|BASE=
    LANG=<Cobol|Asm|PLI|RPG>      MODE=<In|Out|Inout>
    CTRL=<Frset,Freekb,Alarm,Print,>
  EXATT=<No|Maponly|Yes>
  COLOR=<Blue|Red|Pink|Green|Turquoise|Yellow|Neutral|Default>
  HIGHLIGHT=<Off|Blink|Reverse|Underline>
  PS=<Base,>
  VALIDDN=<Mustenter,Mustfill,Trigger>
  TERM=           SUFFIX=
  TIOAPFX=<Yes|No>      CURSLOC=<Yes|No>
  PARTN=< ,Activate>      LDC=
  OBFMT=<Yes|No>      DATA=<Field|Block>
  HTAB=
  VTAB=

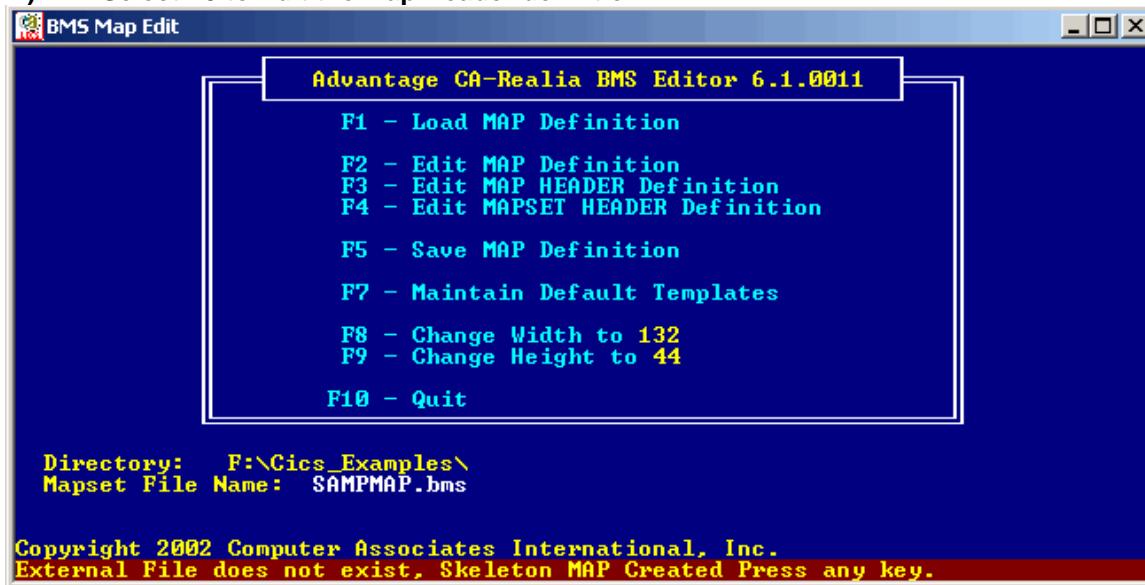
```

**TAB, Shift-TAB, and Arrow Keys** move the cursor. Contrasting keywords are active; dim ones are inactive. Press **F1** to activate/deactivate a keyword. Pressing a letter acts as a **TAB** to the next word starting with that letter. Press **RETURN** for next operand; **F3** for previous operand.

Press **F9** when complete, **ESC** to cancel, **Alt-H** for Help

(3) Select F9 to complete the definition of Map Header

iii) Select F3 to Edit the Map Header definition



(1) Navigate using tab keys  
(2) Select CTRL->Freekb,EXATT->No,COLUMN->,LINE->1,TIOAPFX->Yes

**BMS Map Edit**

**Map Header Definition - DFHMDI**

**MAP NAME=SAMPMAP**

```

SIZE=<24, 80>
CTRL=<Frset,Freekb,Alarm,Print, >
EXTATT=<No |Maponly|Yes>
COLOR=<Blue |Red |Pink |Green |Turquoise |Yellow |Neutral |Default>
PS=<Base | >
HIGHLIGHT=<Off |Blink |Reverse |Underline>
VALIDDN=<Mustenter,Mustfill,Trigger>
COLUMN=<    1 |Next |Same>
LINE=<    1 |Next |Same>
FIELDS =No          CURSLOC=<Yes |No>
PARTN=<   ,Activate>      OBFMT=<Yes |No>
TIOAPPFX=<Yes |No>        HEADER=Yes
      DATA=<Field|Block>    TRAILER=Yes
JUSTIFY=<Left,Right |First,Last>

```

**TAB, Shift-TAB, and Arrow Keys move the cursor. Contrasting keywords are active; dim ones are inactive. Press F1 to activate/deactivate a keyword. Pressing a letter acts as a TAB to the next word starting with that letter. Press RETURN for next operand; F3 for previous operand.**

**Press F9 when complete, ESC to cancel, Alt-H for Help**

(3) Select F9 to complete the definition of Mapset Header

iv) Select F2 to Edit the map definition

(1) Move the Arrow keys and position it at the point where the heading (constant) has to appear

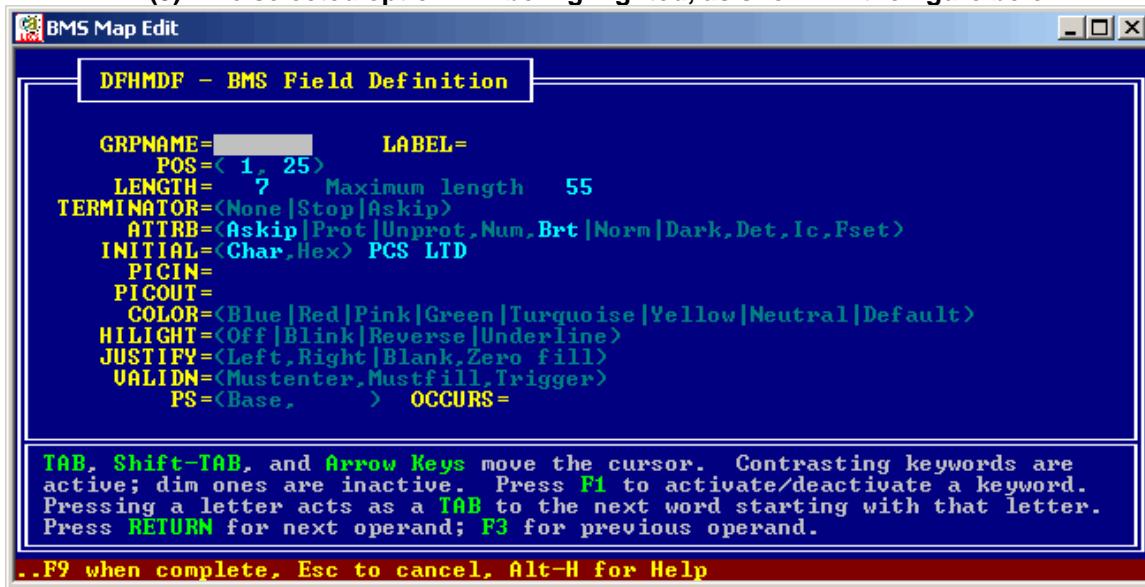
**BMS Map Edit**

**sampmap** ROW 01 COL 028 ..Alt-H For Help, Esc to Exit

v) Press F2 to select the attributes of the field

- (1) Navigate using tab keys
- (2) To select individual attributes such as Brt, Askip press F1

(3) The selected option will be highlighted, as shown in the figure below



```

DFHMDF - BMS Field Definition

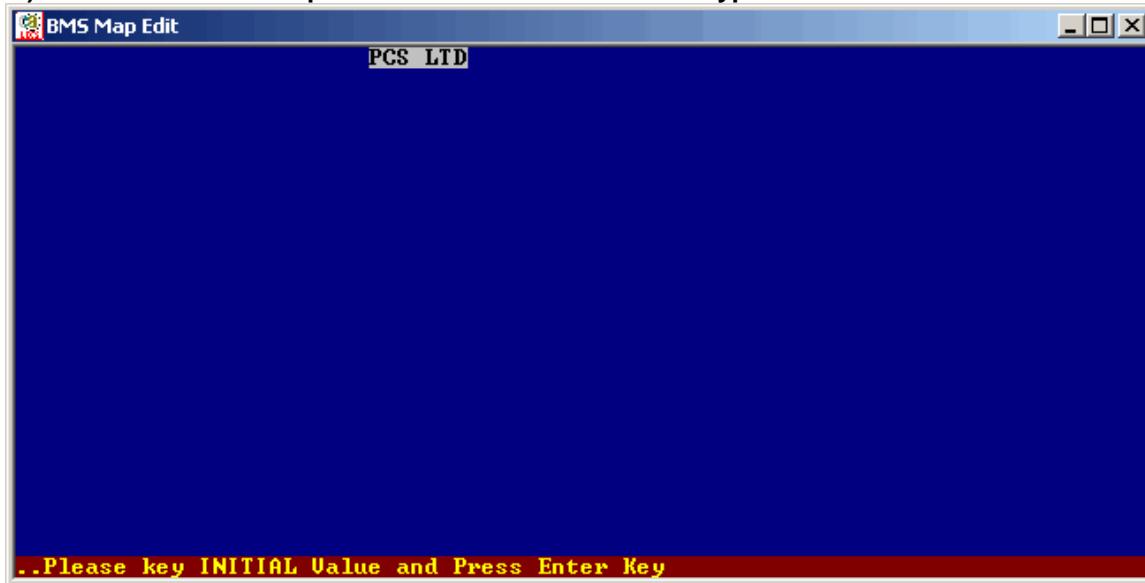
GRPNAME=          LABEL=
POS=< 1, 25>
LENGTH=    ? Maximum length   55
TERMINATOR=<None|Stop|Askip>
ATTRB=<Askip|Prot|Unprot,Num,Brt|Norm|Dark,Det,Ic,Fset>
INITIAL=<Char,Hex> PCS LTD
PICIN=
PICOUT=
COLOR=<Blue|Red|Pink|Green|Turquoise|Yellow|Neutral|Default>
HIGHLIGHT=<Off|Blink|Reverse|Underline>
JUSTIFY=<Left,Right|Blank,Zero fill>
VALIDN=<Mustenter,Mustfill,Trigger>
PS=<Base,> OCCURS=

```

**TAB, Shift-TAB, and Arrow Keys** move the cursor. Contrasting keywords are active; dim ones are inactive. Press **F1** to activate/deactivate a keyword. Pressing a letter acts as a **TAB** to the next word starting with that letter. Press **RETURN** for next operand; **F3** for previous operand.

..F9 when complete, Esc to cancel, Alt-H for Help

vi) Press F9 to complete the definition of the field. Type the Initial value for the field



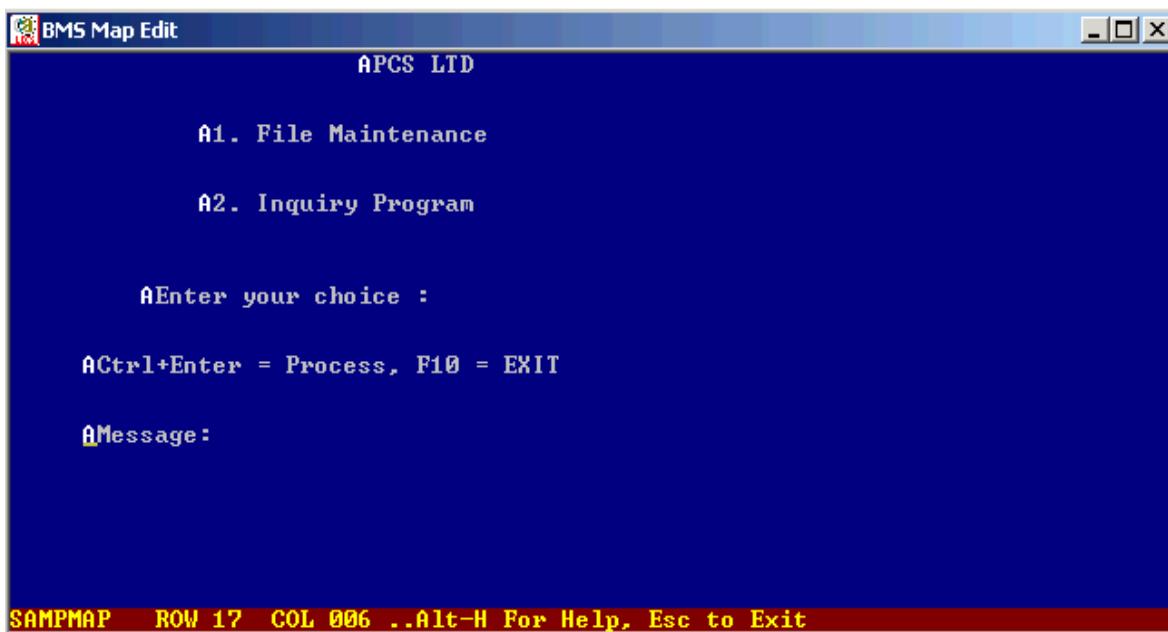
```

PCS LTD
```

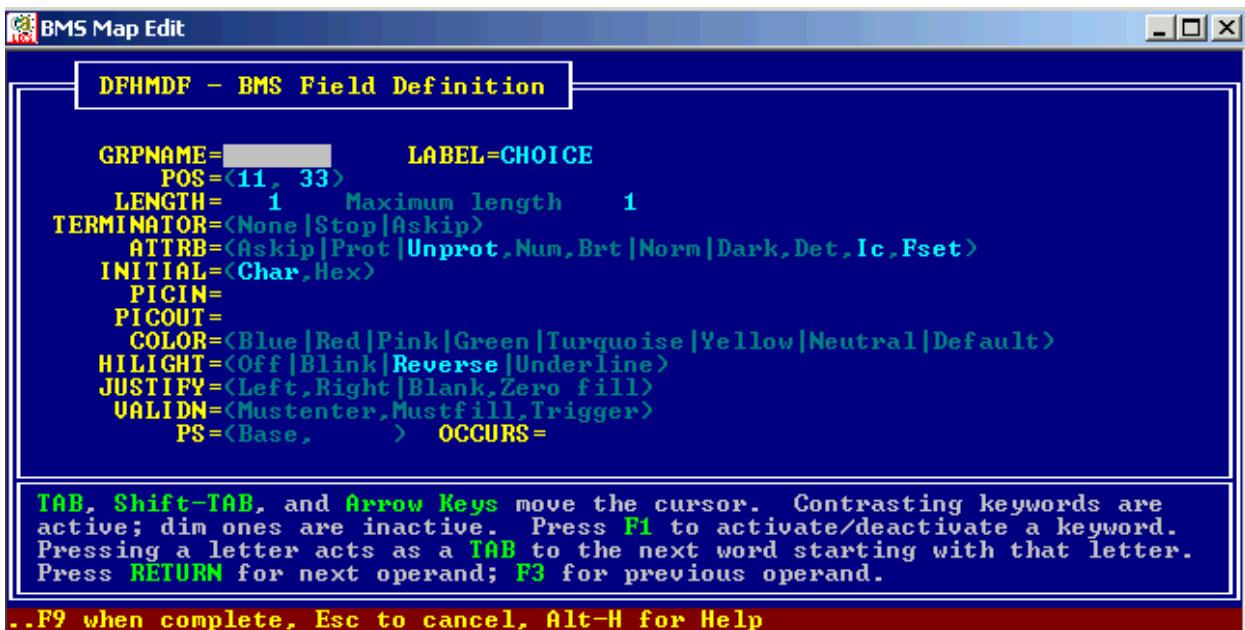
..Please key INITIAL Value and Press Enter Key

<<to do>>

Similarly, create a Map with the following fields (constants)



(1) Create a Data entry field named choice with the following attributes



(2) Create a Stopper field for the choice field, with the following attributes

**BMS Map Edit**

**DFHMDF - BMS Field Definition**

```

GRPNAME=          LABEL=
    POS=<11, 35>
    LENGTH=      1 Maximum length   45
    TERMINATOR=<None|Stop|Askip>
        ATTRB=<Askip|Prot|Unprot,Num,Brt|Norm|Dark,Det,Ic,Fset>
    INITIAL=<Char,Hex>
    PICIN=
    PICOUT=
    COLOR=<Blue|Red|Pink|Green|Turquoise|Yellow|Neutral|Default>
    HIGHLIGHT=<Off|Blink|Reverse|Underline>
    JUSTIFY=<Left,Right|Blank,Zero fill>
    VALIDDN=<Mustenter,Mustfill,Trigger>
    PS=<Base,> OCCURS=

```

**TAB, Shift-TAB, and Arrow Keys move the cursor. Contrasting keywords are active; dim ones are inactive. Press F1 to activate/deactivate a keyword. Pressing a letter acts as a TAB to the next word starting with that letter. Press RETURN for next operand; F3 for previous operand.**

**..F9 when complete, Esc to cancel, Alt-H for Help**

- (3) The map with the choice field and the stopper field should resemble the following screen

**BMS Map Edit**

**APCS LTD**

A1. File Maintenance  
A2. Inquiry Program

AEnter your choice : U-AX

ACtrl+Enter = Process, F10 = EXIT

AMessage :

**SAMPMAP ROW 11 COL 033 ..Alt-H For Help, Esc to Exit**

- (4) Create the Message field (Display-only) with the following attributes

**BMS Map Edit**

**DFHMDF - BMS Field Definition**

```

GRPNAME=          LABEL=MESSAGE
    POS=<17, 22>
    LENGTH= 30 Maximum length 30
TERMINATOR=<None|Stop|Askip>
    ATTRB=<Askip|Prot|Unprot,Num,Brt|Norm|Dark,Det,Ic,Fset>
INITIAL=<Char,Hex>
PICIN=
PICOUT=
COLOR=<Blue|Red|Pink|Green|Turquoise|Yellow|Neutral|Default>
HIGHLIGHT=<Off|Blink|Reverse|Underline>
JUSTIFY=<Left,Right|Blank,Zero fill>
VALIDN=<Mustenter,Mustfill,Trigger>
PS=<Base,      > OCCURS=

```

**TAB, Shift-TAB, and Arrow Keys move the cursor. Contrasting keywords are active; dim ones are inactive. Press F1 to activate/deactivate a keyword. Pressing a letter acts as a TAB to the next word starting with that letter. Press RETURN for next operand; F3 for previous operand.**

**..F9 when complete, Esc to cancel, Alt-H for Help**

- (5) Press F9 to complete the definition of the Message field
- (6) Define a stopper field for the Message field

**(7) Define a Dummy field with the following attributes**

**BMS Map Edit**

**DFHMDF - BMS Field Definition**

```

GRPNAME=          LABEL=
    POS=<23, 68>
    LENGTH= 1 Maximum length 12
TERMINATOR=<None|Stop|Askip>
    ATTRB=<Askip|Prot|Unprot,Num,Brt|Norm|Dark,Det,Ic,Fset>
INITIAL=<Char,Hex>
PICIN=
PICOUT=
COLOR=<Blue|Red|Pink|Green|Turquoise|Yellow|Neutral|Default>
HIGHLIGHT=<Off|Blink|Reverse|Underline>
JUSTIFY=<Left,Right|Blank,Zero fill>
VALIDN=<Mustenter,Mustfill,Trigger>
PS=<Base,      > OCCURS=

```

**TAB, Shift-TAB, and Arrow Keys move the cursor. Contrasting keywords are active; dim ones are inactive. Press F1 to activate/deactivate a keyword. Pressing a letter acts as a TAB to the next word starting with that letter. Press RETURN for next operand; F3 for previous operand.**

**..F9 when complete, Esc to cancel, Alt-H for Help**

**BMS Map Edit**

**DFHMDF - BMS Field Definition**

```

GRPNAM=           LABEL=DUMMY
    POS=<23, 68>
    LENGTH= 1      Maximum length 12
TERMINATOR=<None|Stop|Skip>
    ATTRB=<Skip|Prot|Unprot,Num,Brt|Norm|Dark,Det,Ic,Fset>
INITIAL=<Char,Hex>
PICIN=
PICOUT=
COLOR=<Blue|Red|Pink|Green|Turquoise|Yellow|Neutral|Default>
HIGHLIGHT=<Off|Blink|Reverse|Underline>
JUSTIFY=<Left,Right|Blank,Zero fill>
VALIDN=<Mustenter,Mustfill,Trigger>
PS=<Base,       > OCCURS=

```

**TAB, Shift-TAB, and Arrow Keys move the cursor. Contrasting keywords are active; dim ones are inactive. Press F1 to activate/deactivate a keyword. Pressing a letter acts as a TAB to the next word starting with that letter. Press RETURN for next operand; F3 for previous operand.**

**..F9 when complete, Esc to cancel, Alt-H for Help**

(8) The final map should resemble the screen displayed below

**BMS Map Edit**

**APCS LTD**

A1. File Maintenance

A2. Inquiry Program

AEnter your choice : U\_AX

ACtrl+Enter = Process, F10 = EXIT

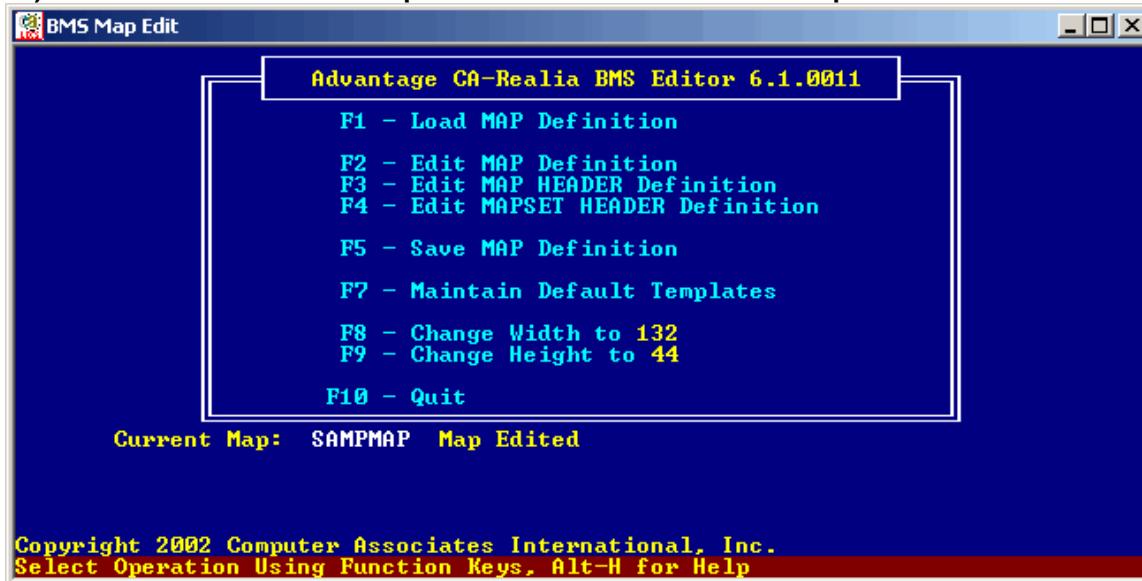
AMessage: A-----AX

A-

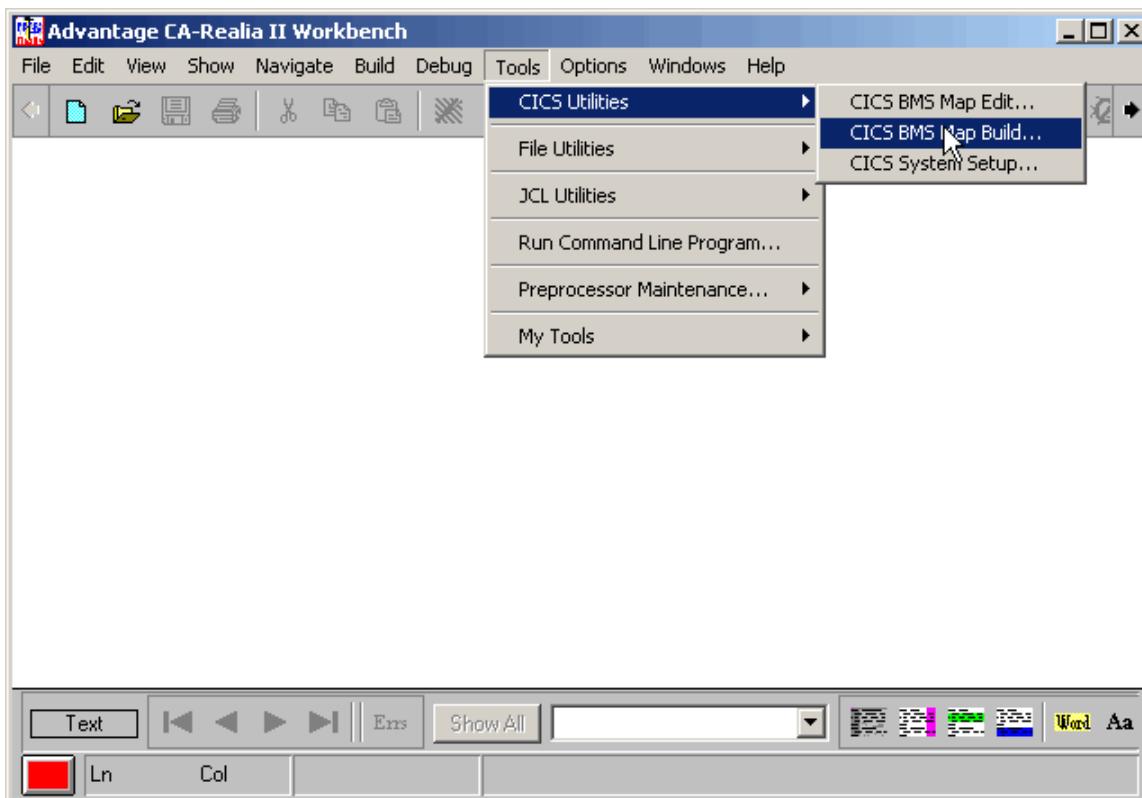
**SAMPMAP ROW 23 COL 068 ..Alt-H For Help, Esc to Exit**

(9) Press ESC to complete the definition of the map

vii) Press F5 to save the Map definition and F10 to exit the Map



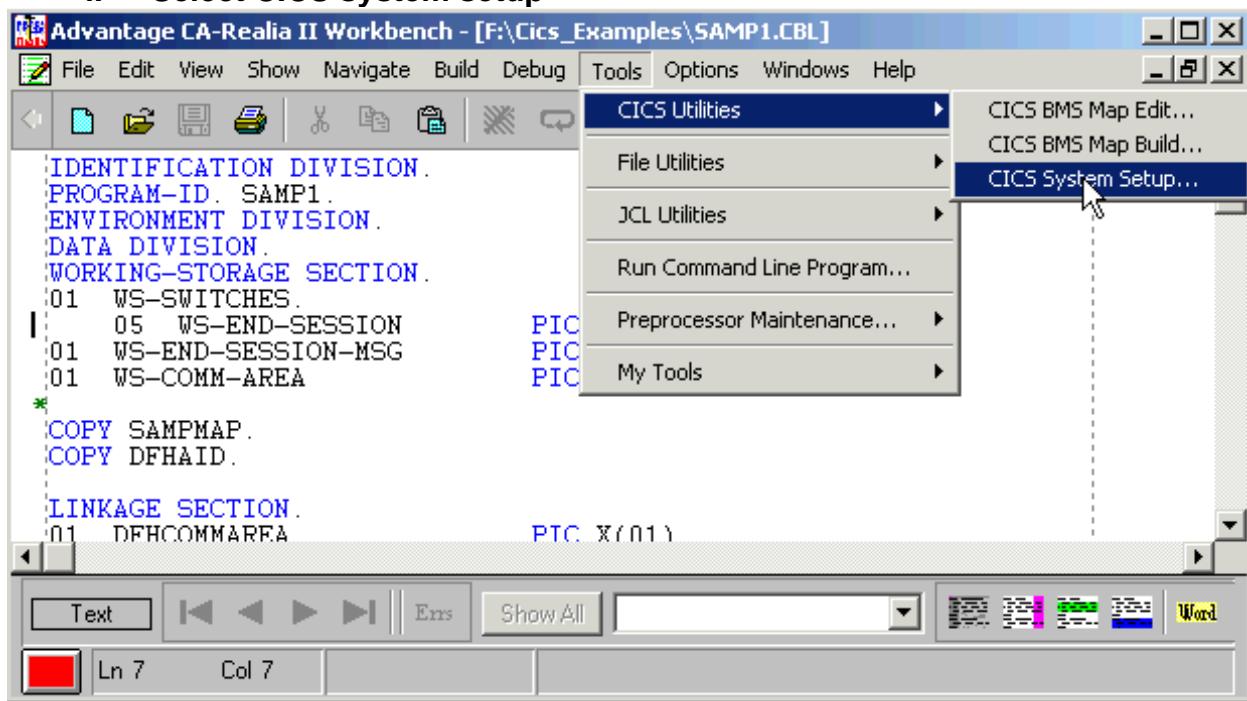
### III. Compile the Map

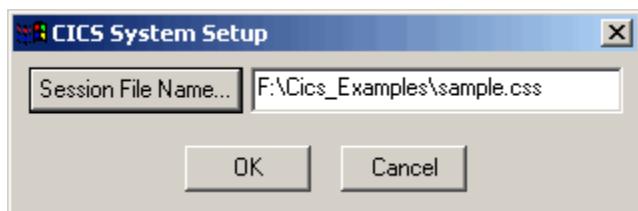


**<<to do>>**

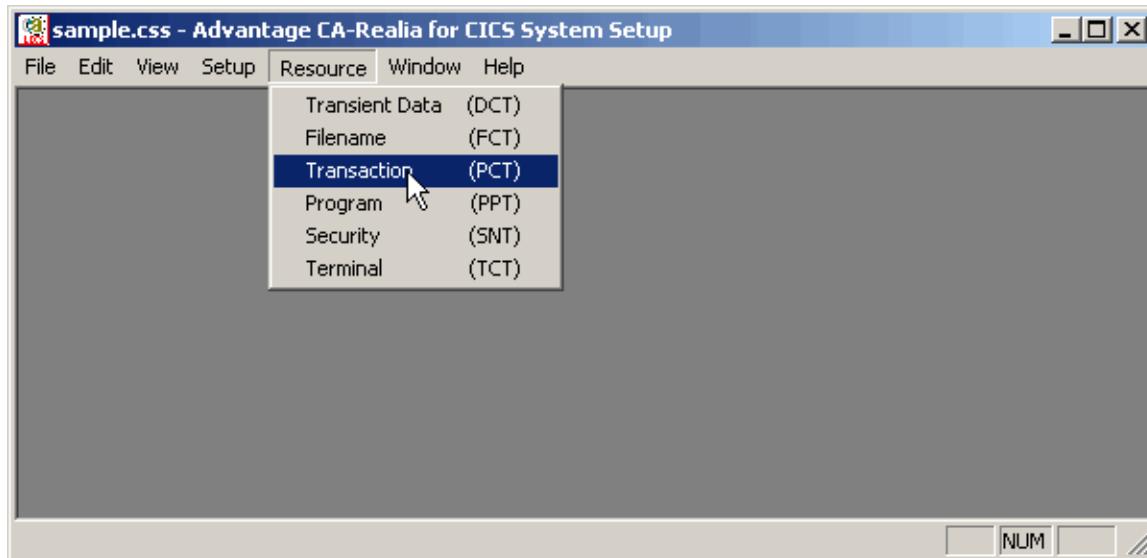
- 1) Select the Mapset from the appropriate directory.
- 2) View the additional files generated, after successful compilation of the mapset.
- 3) View the sampmap.bms,sampmap.cbl file in any editor

*Translating and compiling COBOL programs using Ca-Realia*

**I. Code the following COBOL program in the Workbench IDE(Editor).**
**II. Add appropriate entries in the PPT and PCT for the COBOL Program and the Mapset.**
**i. Select CICS system setup**

**ii. Select the CSS file from the appropriate folder**



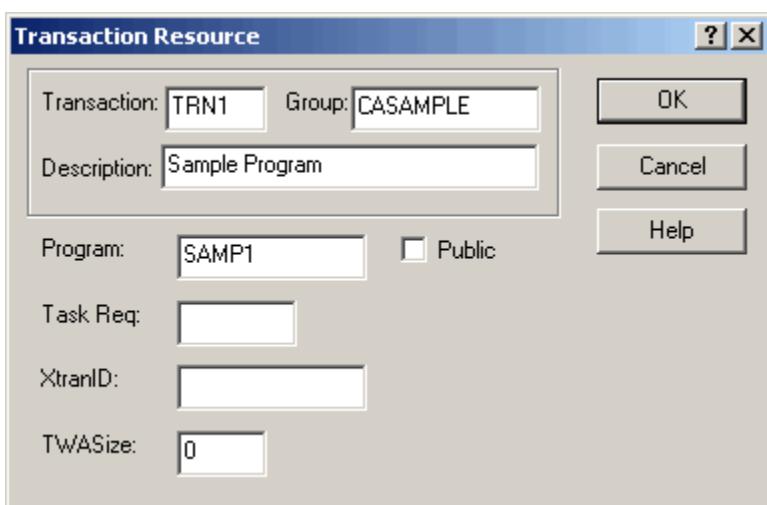
ii. Select Resource->Transaction (PCT), to add entries in the Program Control Table



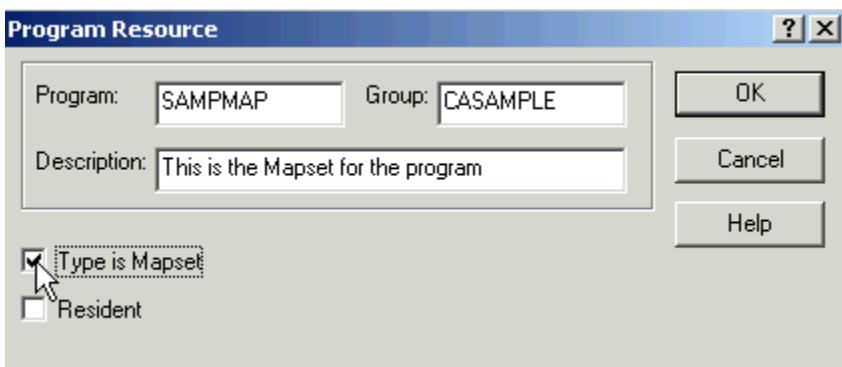
iii. Select Edit->Insert to add entries in the PCT

sample.css - Advantage CA-Realia for CICS System Setup - [Transaction Definition]					
Transact	Insert	Ins	Description	Program	Taskreq
CINT	Copy		Session Initialize routine	CINT	None
CMAP	Delete	Del	Map view Utility	CMAP	None
CSPG	Rename		SEND PAGE Utility	CSPG	None
CSSN	Modify	Enter	Security Routine	CSSN	None
	REALCICS				

- iv. Add the following entry to associate the Transaction with the program.

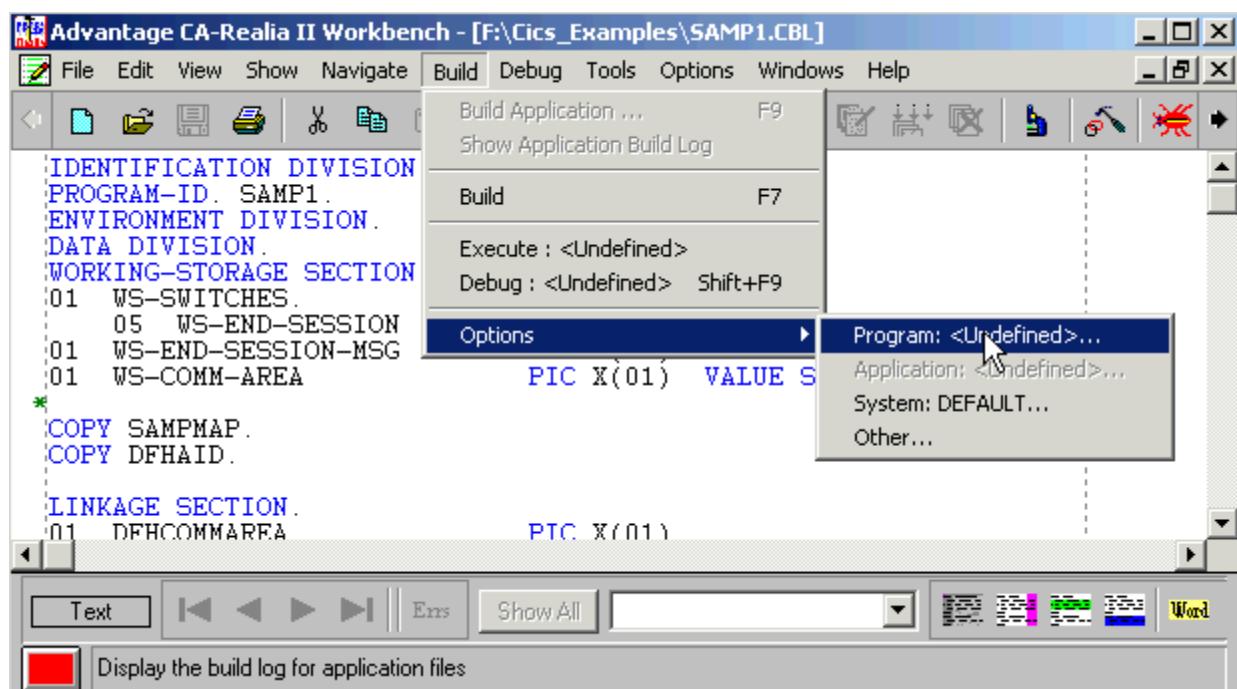


- v. Select Resource->Program to add entries in the PPT  
vi. Add the following entries to add the program and the Mapset in the PPT.

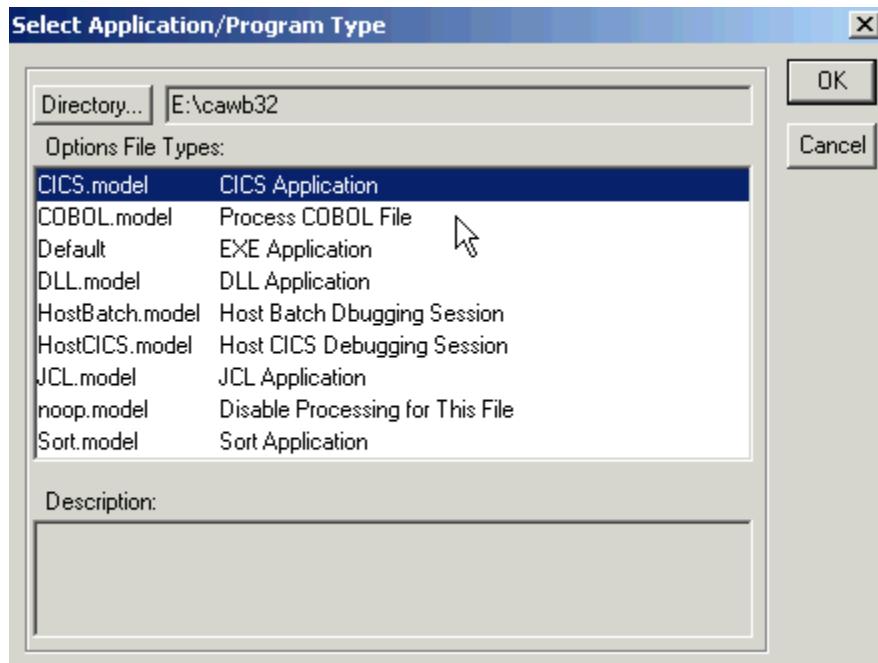


### III. Build the Cobol Program.

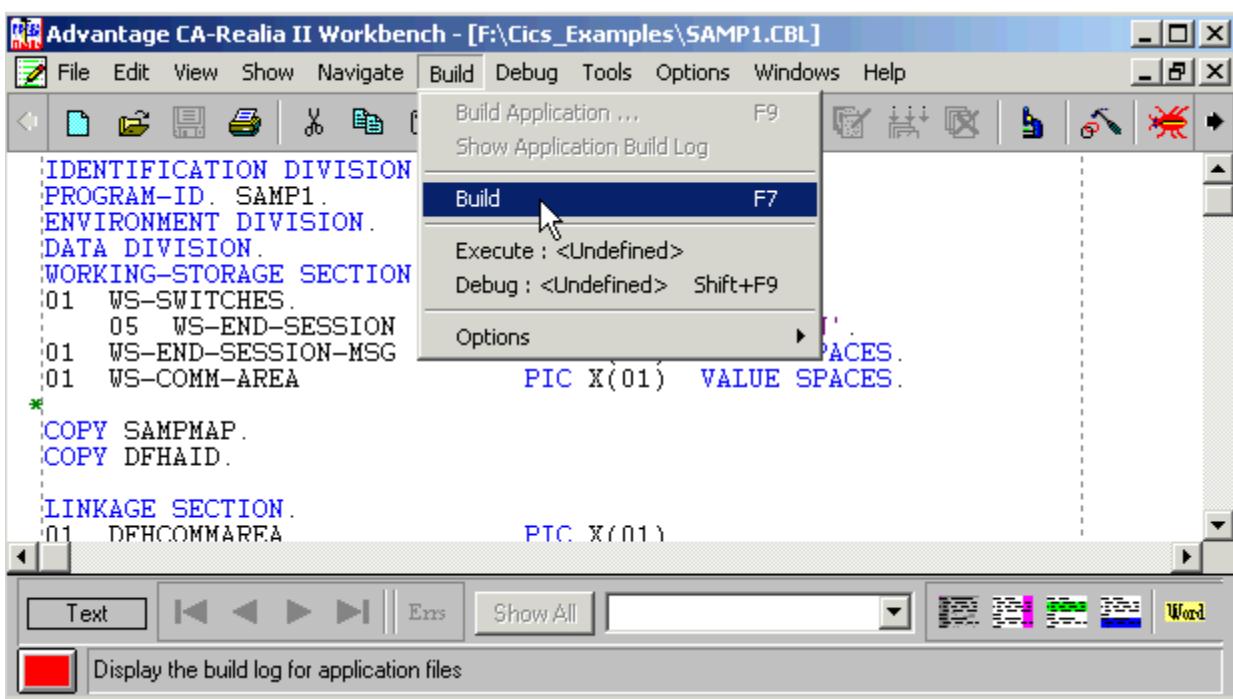
- Select Build->Options->Program:<Undefined>...



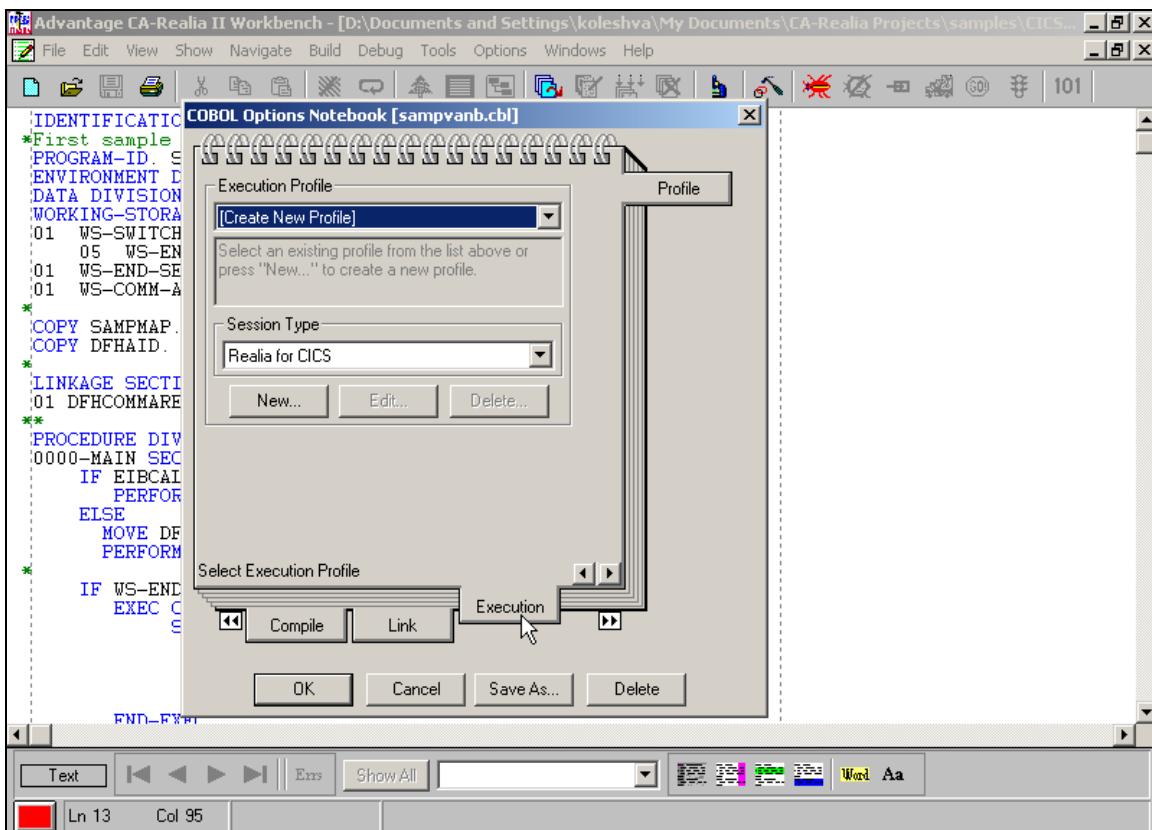
## 2. Select CICS Application



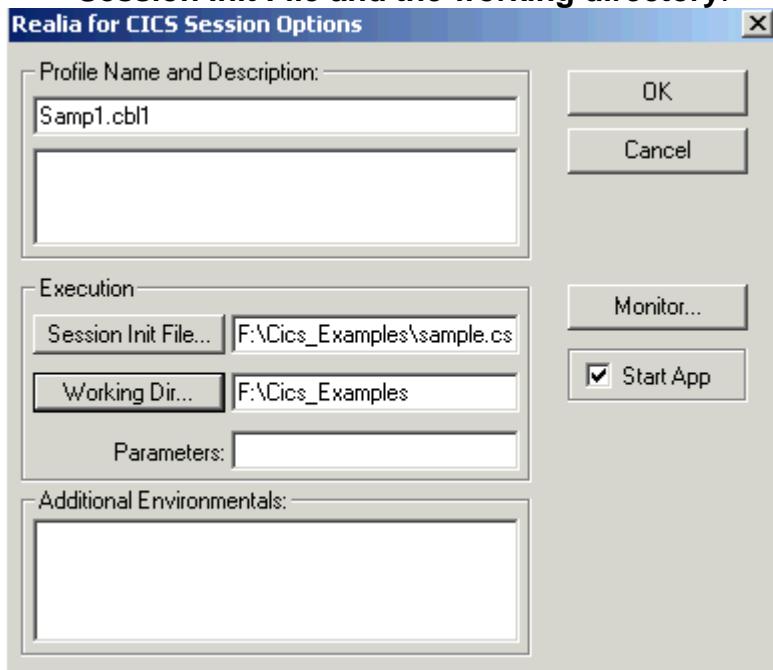
## 3. Build the COBOL program



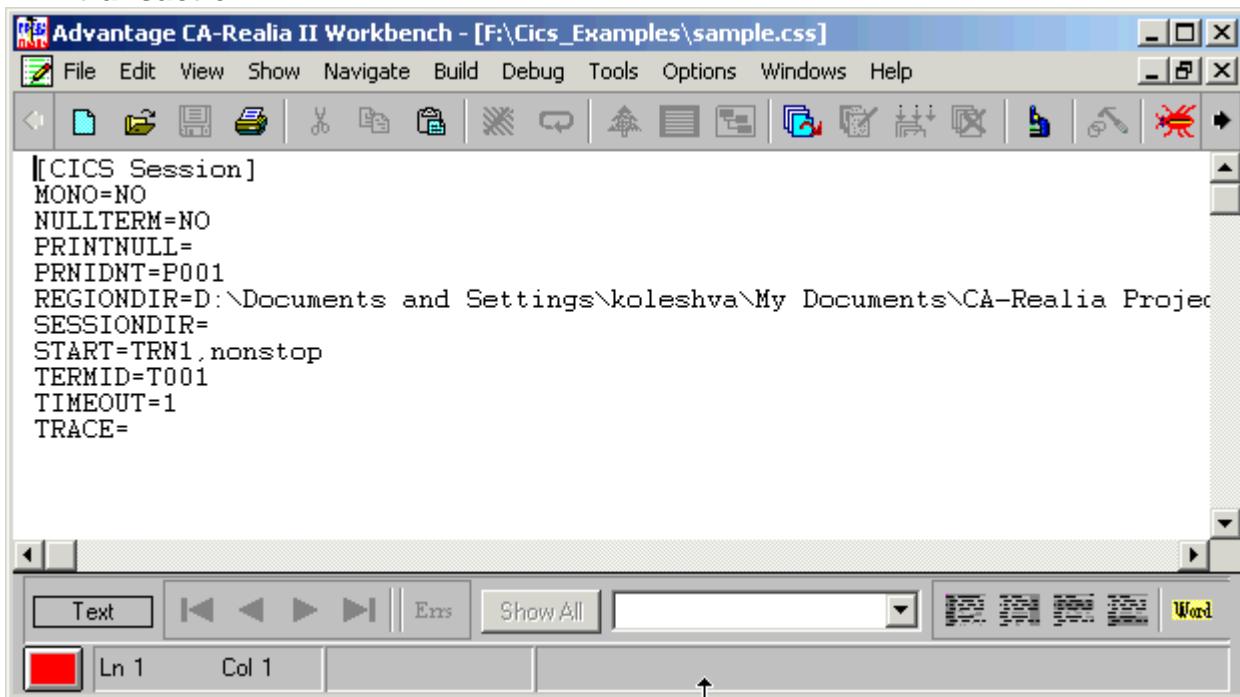
#### 4. Select Build->Options->Program:<Undefined>...



5. On the above screen select New... and then select the Session Init File and the working directory.



6. Open the file sample.css in any editor and enter TRN1 as the transaction.

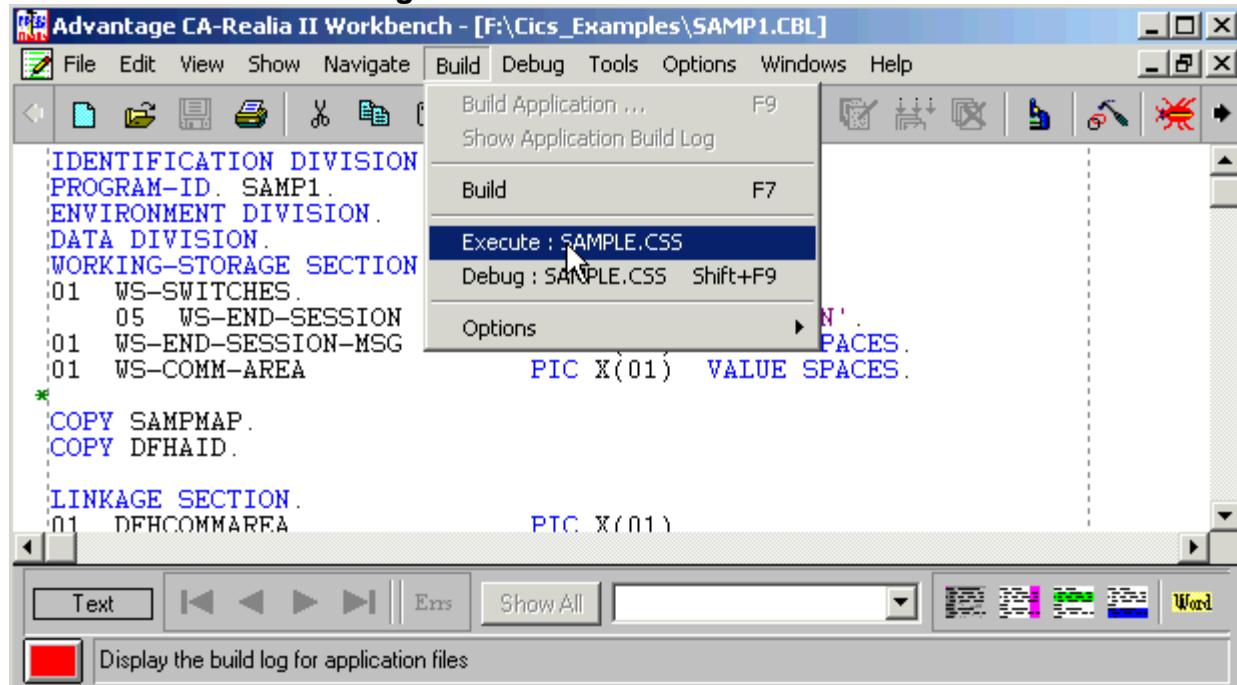


```

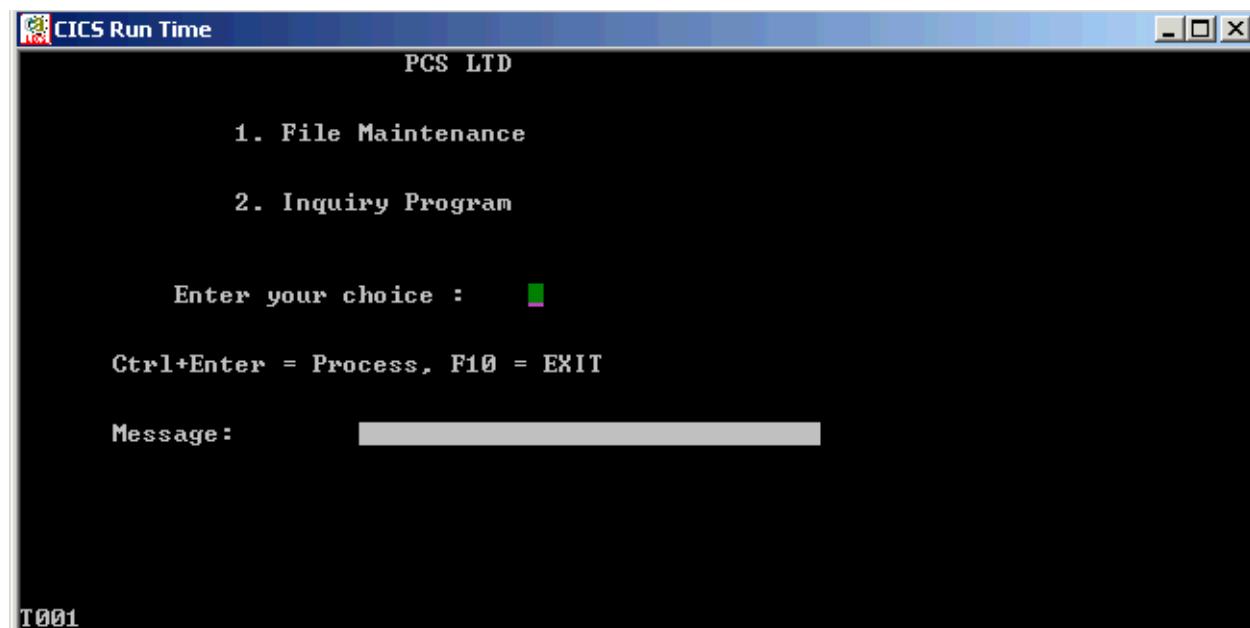
[[CICS Session]]
MONO=NO
NULLTERM=NO
PRINTNULL=
PRNIDNT=P001
REGIONDIR=D:\Documents and Settings\koleshva\My Documents\CA-Realia Project
SESSIONDIR=
START=TRN1.nonstop
TERMID=T001
TIMEOUT=1
TRACE=

```

## 7. Execute the Program



## 8. The screen as shown below is displayed

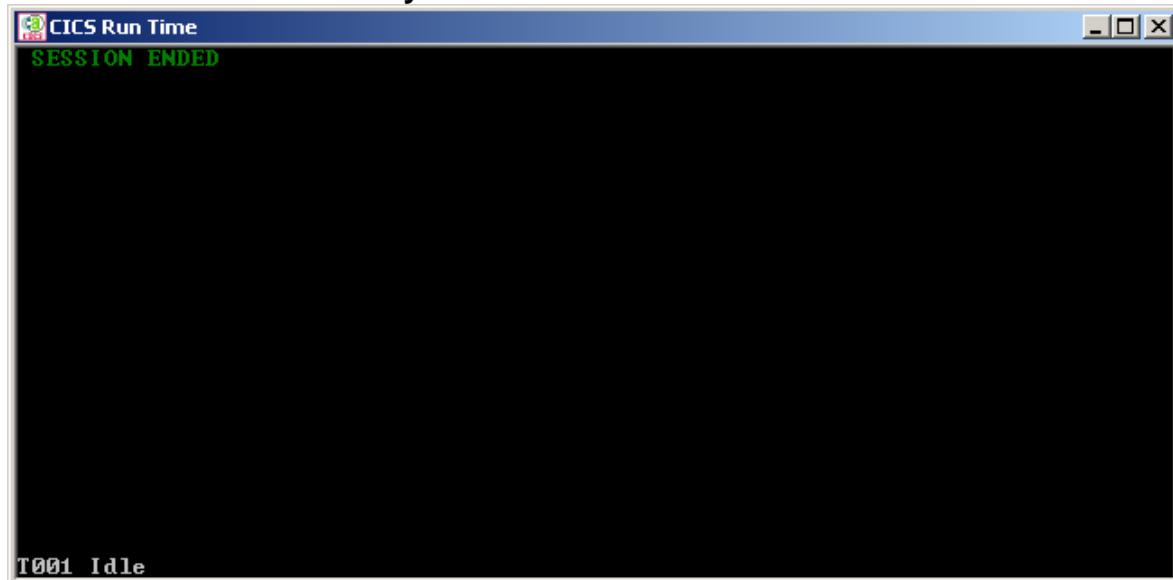


**<< to do >>**

- 1) Enter the choice as 1,2,3 and note the message.

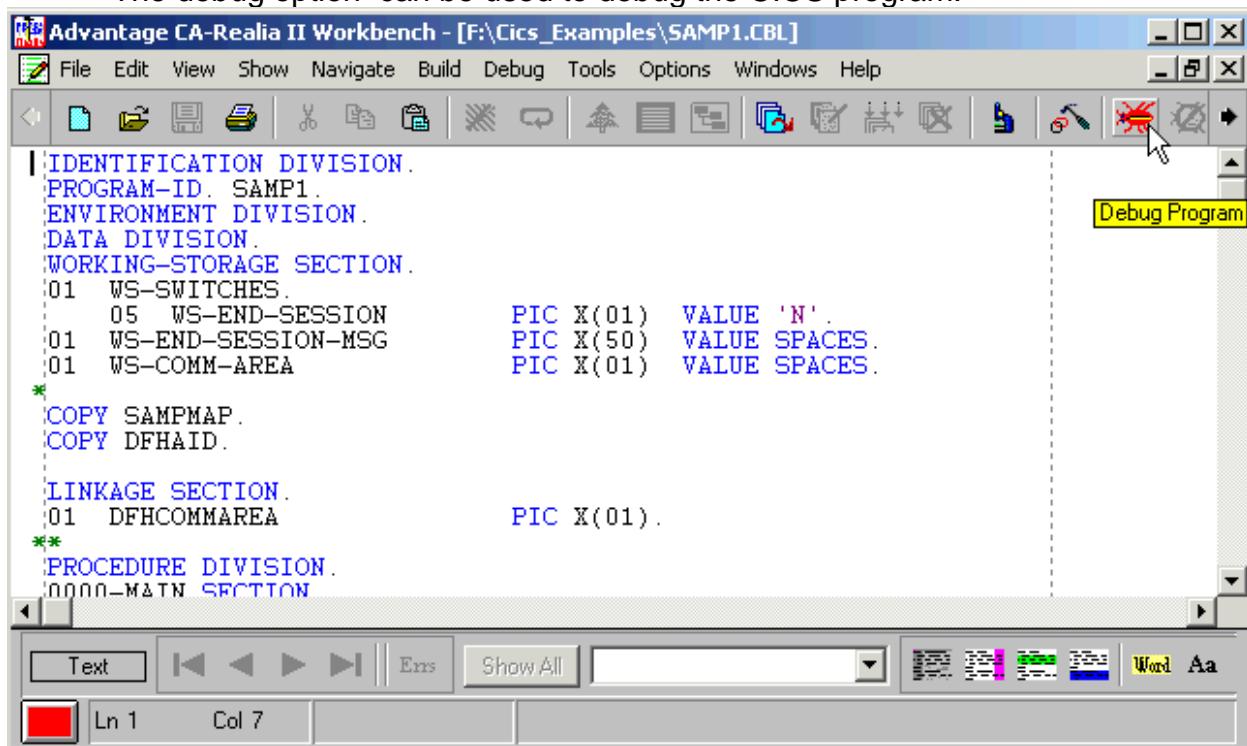
- 2) Press an invalid AID key and note the message

#### 9. Press the F10 key to terminate the session



#### IV. Debug the CICS Program

The debug option can be used to debug the CICS program.



The screenshot shows the "Advantage CA-Realia II Workbench" interface with the title bar "Advantage CA-Realia II Workbench - [F:\Cics\_Examples\SAMP1.CBL]". The main area displays the source code of a CICS program:

```

IDENTIFICATION DIVISION.
PROGRAM-ID. SAMP1.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01 WS-SWITCHES.
    05 WS-END-SESSION      PIC X(01)  VALUE 'N'.
01 WS-END-SESSION-MSG   PIC X(50)  VALUE SPACES.
01 WS-COMM-AREA        PIC X(01)  VALUE SPACES.

*COPY SAMPMAP.
*COPY DFHAID.

LINKAGE SECTION.
01 DFHCOMMAREA          PIC X(01).

**PROCEDURE DIVISION.
00000-MATN SECTION

```

The toolbar at the top includes icons for File, Edit, View, Navigate, Build, Debug, Tools, Options, Windows, and Help. The "Debug" icon is highlighted with a yellow box and the label "Debug Program". The bottom of the window has standard Windows-style controls for zooming and moving.

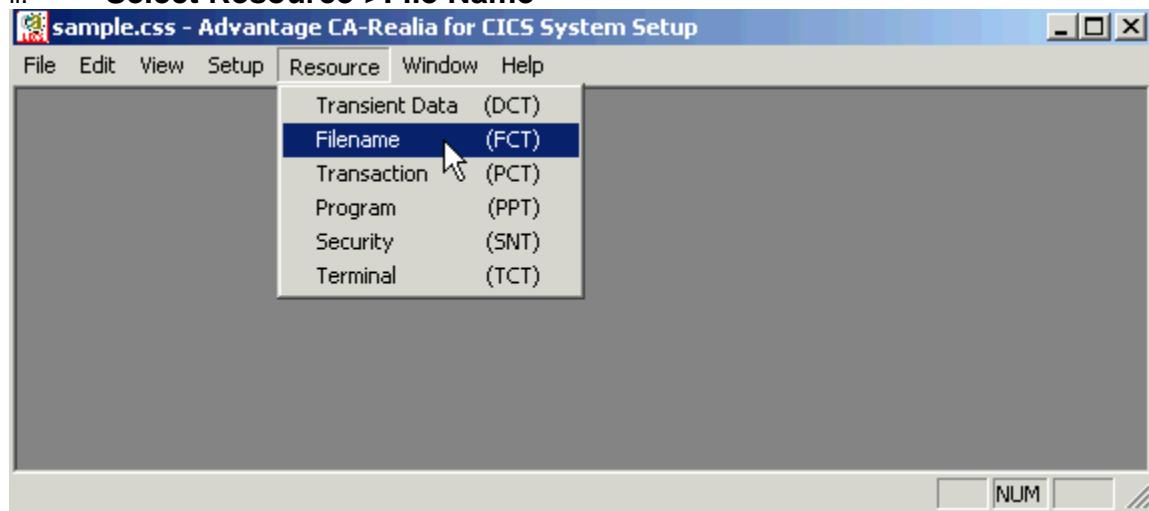
## *File Handling using CA-Realia*

### I. Add the dataset to the FCT.

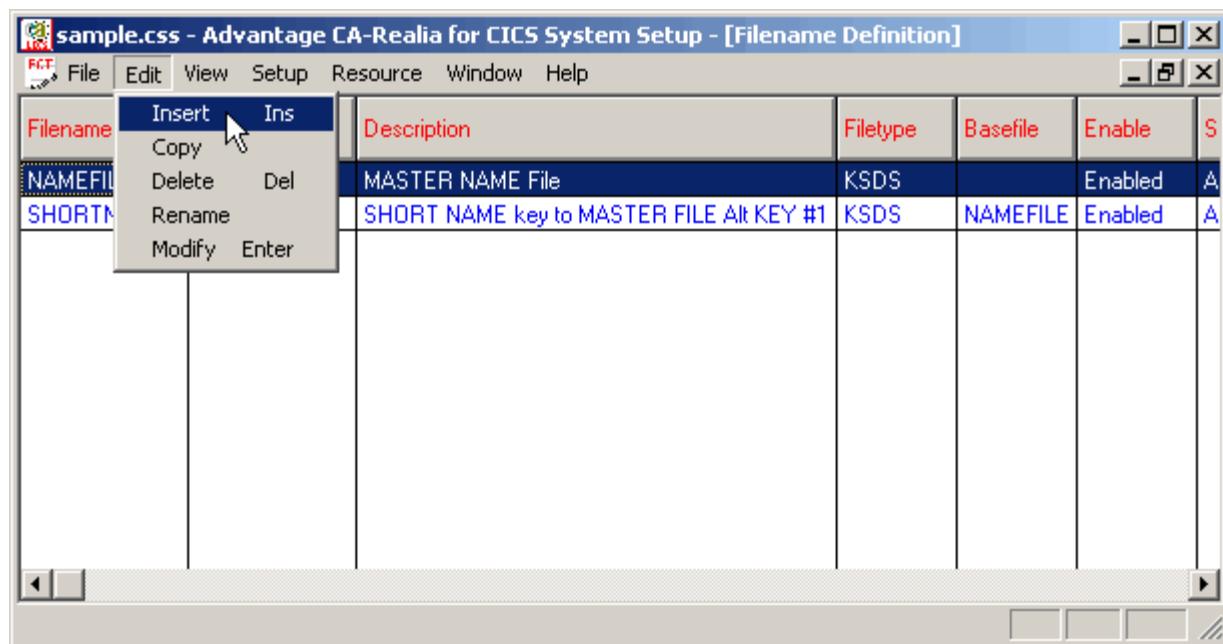
#### i. <>to do >>

1. Select Tools->CICS utilities-> CICS system setup...
2. Select sample.css from the appropriate folder

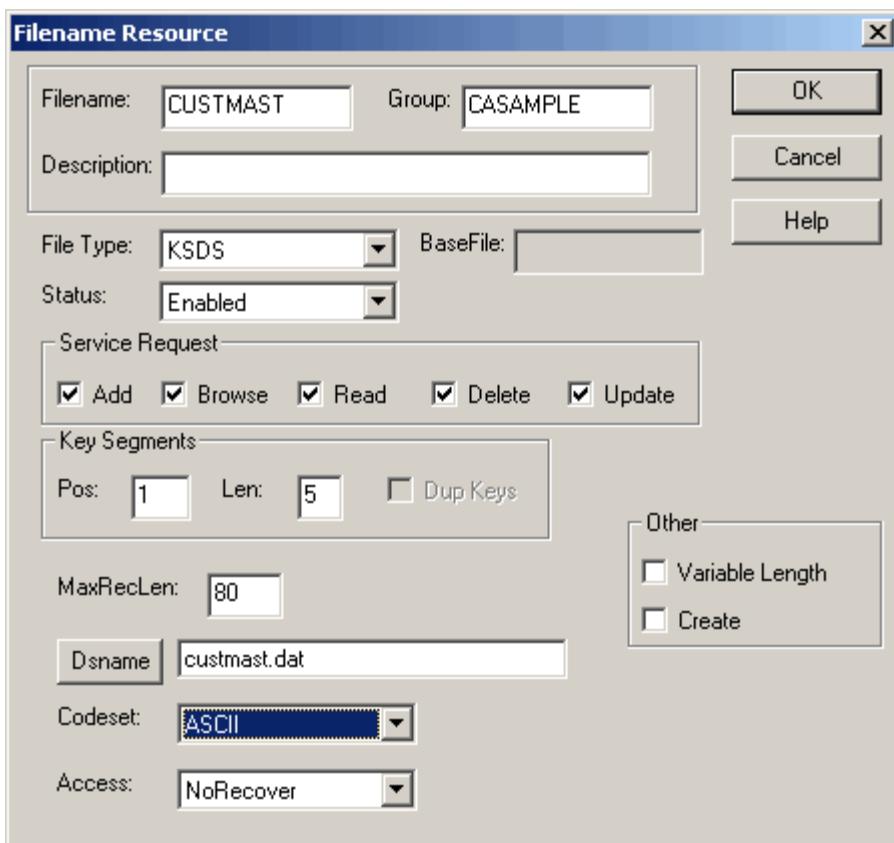
#### ii. Select Resource->File Name



#### iii. Select Edit->Insert



- iv. Specify the file type, length of the key field, maximum record length, dsname



#### v. The Filename gets added in the FCT

sample.css - Advantage CA-Realia for CICS System Setup - [Filename Definition]							
Filename	Group	Description	Filetype	Basefile	Enable	S	
CUSTMAST	CASAMPLE	Customer Program	KSDS		Enabled	A	
NAMEFILE	CASAMPLE	MASTER NAME File	KSDS		Enabled	A	
SHORTNAM	CASAMPLE	SHORT NAME key to MASTER FILE Alt KEY #1	KSDS	NAMEFILE	Enabled	A	

After adding the dataset to the FCT, the dataset can now be referred to in any program, which references the dataset.

**Appendix B: Table of Figures**

Figure 1: CEBR.....	34
Figure 2: CEBR.....	35
Figure 3: Browse ended message.....	36
Figure 4: CEBR.....	37
Figure 5: Typing the command.....	38
Figure 6: Output.....	39
Figure 7: Response: Normal.....	40
Figure 8: Response : INVREQ .....	41
Figure 9: Response: PGIMIDERR.....	42
Figure 10: Typing the command.....	43
Figure 11: Output.....	44
Figure 12: Typing Command .....	45
Figure 13: Output.....	46
Figure 14: Session ended message.....	47
Figure 15: Typing command.....	48
Figure 16: Output.....	48
Figure 17: Output.....	49
Figure 18: Output.....	50
Figure 19: Typing the CEDF command.....	51
Figure 20: Output.....	52
Figure 21: Typing the command.....	52
Figure 22: Output.....	53