Unit 2. CICS Concepts

Objectives

- **Application Programming Techniques**
- **Multitasking**
- **Multithreading**
- **Reentrant and Quasi-Reentrant Programs**
- **Conversational and Pseudo –Conversational Transactions**
- **Application Programming Logical levels**
- **Execution of CICS Application Program**

Figure: 2-1. Objectives

Notes :

Application Program Techniques

General Format:

The general format to code a CICS command in a application Program is as follows:

```
EXEC CICS Function

[option(argument value)]

[option(argument value)]

…….

…….

…….

END-EXEC.
```

Figure: 2-2. Application Programming Techniques

Notes:

Function : CICS Service Request.

Option : One of the options available to the command.

Argument Value : Determines the characteristics of the value to be placed for the option as detailed Information.

Each CICS command should end with a delimiter (END-EXEC)

Languages supporting CICS

The different programming languages that support CICS applications are:

- **COBOL**
- **ASSEMBLER**
- **PL/1**
- **C**
- **C++**

Figure: 2-3. Languages supporting CICS

Notes:

CICS translator converts the EXEC CICS commands into CALL statements for a specific programming language. There are CICS translators for Assembler, PL/I, C, C++ and COBOL.

Syntax

For COBOL
```
EXEC CICS
……
……
……
END-EXEC.
```

For PL/1
```
EXEC CICS
……
……
……
……;
```

For ASSEMBLER
```
EXEC CICS
……
……
……
……
```

For C
```
EXEC CICS
……
……
……
……;
```

For C++
```
EXEC CICS
……
……
……
……
```

Figure: 2-4. Syntax

Notes:

Multitasking

- **Execution of more than one task concurrently whether the task uses the same program or different programs.**
- **CICS manages the multitasking of the CICS tasks within its own region.**
- **CICS provides a multitasking environment where more than one CICS task or transaction can run at the same time.**
- **Multitasking is taken care by Scheduler of the operating system.**

Figure: 2-5. Multitasking

Notes:
The work associated with the single terminal, that is a task, will not keep CICS very busy. Therefore 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.

Therefore, within CICS there can be many tasks, hence the term “Multi-tasking”. Besides the task that is executing there may be other tasks that are ready to execute. Other tasks my be waiting for an I/O operation to complete.

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.

Multithreading

- **It is a 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.**

- **CICS manages multithreading of CICS tasks within its own region.**

- **CICS provides the multithreading environment where more than one CICS Task, which uses the same Program, runs concurrently.**

Figure: 2-6. Multithreading

Notes:

Many tasks can be present within CICS (multi-tasking), thus it is possible that more than one task may require the same application program e.g. there may be 4 terminals all making stock enquiries, CICS will create one task for each terminal and load one copy of the program. The 4 tasks will share the use of the program i.e. they will multi-thread through the code.

Reentrant Program

A program which does not modify itself so that it can reenter to itself and continue processing after an interruption by the operating system which, during the interruption, executes other OS tasks including OS tasks of the same program. It is also called as a “reenterable” program or “serially reusable” program.

Reentrant programs are used by online system, and they make it possible for OS to establish the multitasking environment which most online systems require.

A non-reentrant program is a program, which modifies itself so that it cannot Reenter to itself. Therefore it cannot be used in the multithreading environment. Batch programs are the typical examples of non-reentrant program.

Figure: 2-7. Reentrant program

Notes:

Quasi-Reentrant Program

A Reentrant program under CICS environment is called as a Quasi-reentrant Program. It is a CICS program, which does not modify itself during execution. So it can reenter to itself and continue processing after an interruption by CICS.

The Quasi-reentrant program is a feature under the CICS/Multithreading environment. The difference between the Reentrant and Quasi-reentrant program is that whatever operations and functions that the operating system performs during a SVC in the case of Reentrant program is performed by CICS during a CICS command in the case of Quasi-reentrant program.

Figure: 2-8. Quasi-reentrant program

Notes:

Conversational Transaction

- **A conversation is nothing but the interaction of the program with the terminal user.**
- **During the conversation program waits until the user responds, holding Resources unnecessarily.**
- **Only after the completion of the users response (i.e. pressing the ENTER Key) does the program proceed to the next processing.**
- **It is a very inefficient way of conversing with the user.**

Figure: 2-9. Conversational Transaction

Notes:

Transactions can be designed in one of two ways and, although the differences between them are not apparent to the end user, they will affect the way the program is coded and will have implications in other areas such as performance.

If the application designer chooses the conversational approach then the whole dialogue takes place within one CICS transaction.

This means that all processing takes place in one CICS task.

Pseudo-Conversational Transaction

- **In pseudo conversational transaction, when a program or transaction attempts a conversation with a terminal user (i.e. sending a message, expecting a response from the user), it terminates the task after sending a message with a linkage for the next task or program.**

- **When the user completes the response (i.e. pressing the ENTER key), the next task (or program) is automatically initiated by CICS. This task receives the message from the terminal and process it.**

- **From the system point of view, this is a multitask operation, whereas from the user's point of view, this is a normal conversation. This is why it is called "pseudo-conversational".**

Figure: 2-10. Pseudo-Conversational Transaction

Notes:
With pseudo-conversational design, the dialogue will be the same externally but will be implemented in a different way.

Sequence of CICS tasks, where each task process one input and one output message, that is known as a message pair.

The main benefit is that a task only exists at the time taken to process a message pair. This is short, a fraction of a second normally, compare to the seconds or even minutes taken by the user to think and enter data.

It also means that records are not locked across the conversation.

Pseudo-Conversational Techniques

- **Multiple transaction identifiers and multiple programs.**
- **Multiple transaction identifiers but one program.**
- **One transaction identifier and one program.**

Figure: 2-11. Pseudo-Conversational Techniques

Notes:

The programming methodology in which the task will not wait for the terminal users response, but frees the resources after sending the message is called a pseudo conversational technique. Terminating the task everytime the application needs a response from the user and then starting the next transaction when the user presses any attention key is pseudo conversational processing.

Application Program Logical Levels

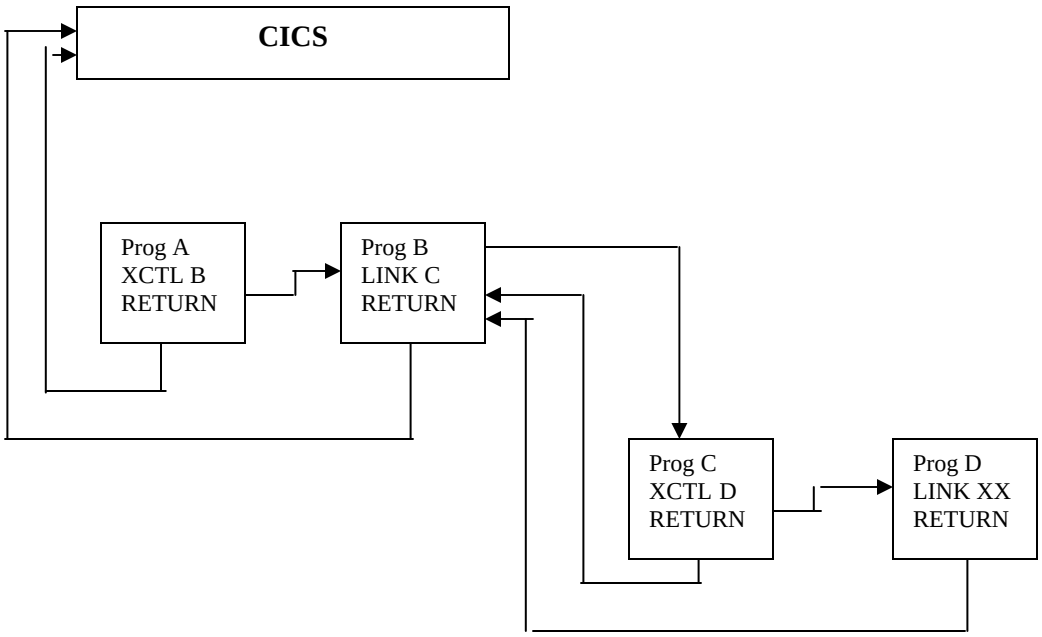


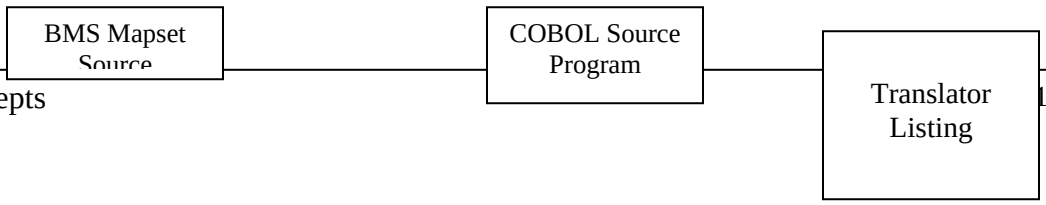
Figure: 2-12. Application program logical levels

Notes:

Link command is used to pass control from an application program at one logical level to another application at next lower logical level. The calling (LINKing) program expects control to be returned to it. Data can be passed to the called program through a special communication area called COMMAREA.

XCTL command is used to pass control from one application program to another application program at the same logical level. The calling (XCTLing) program does not expect control to be returned. data can be passed to the called program through a special communication area called COMMAREA.

Execution of CICS Application Program



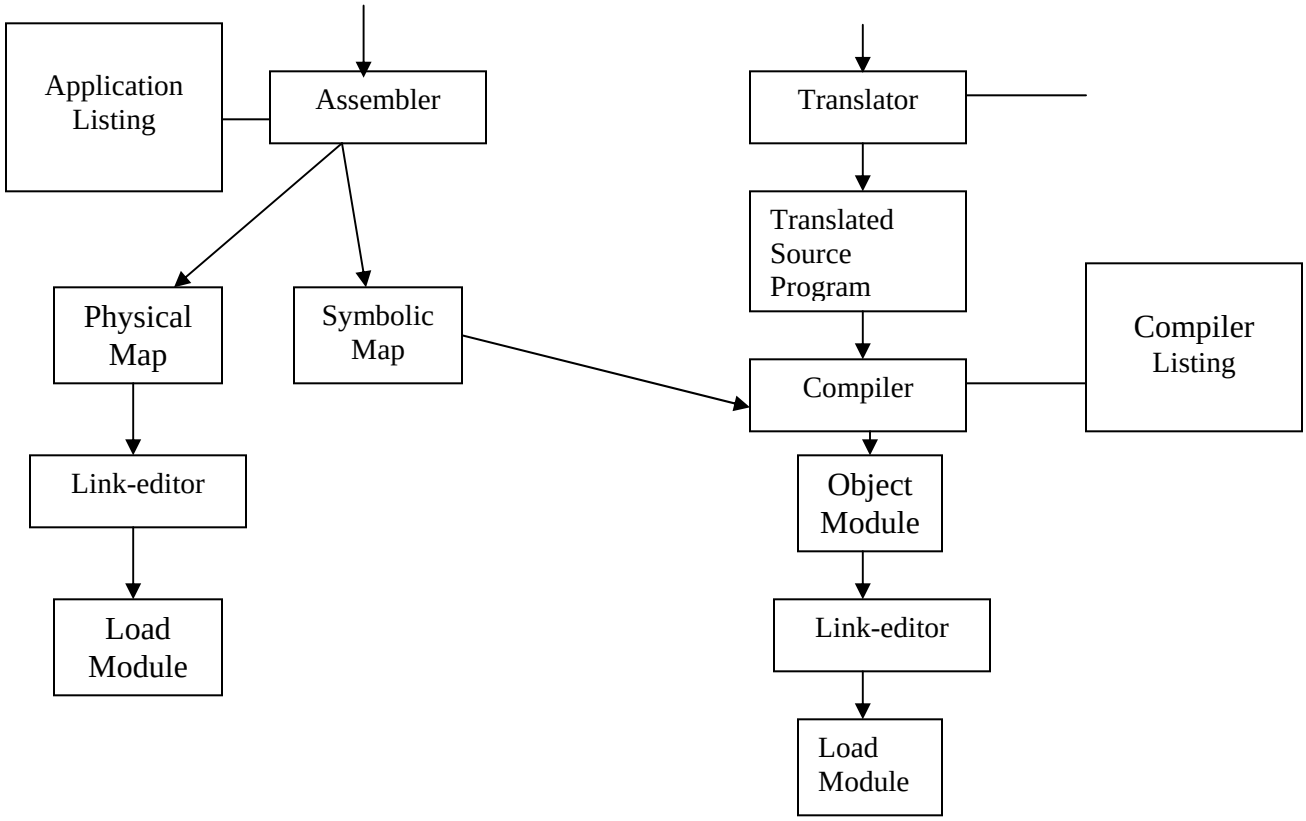


Figure: 2-13. Execution of CICS Application Program

Notes:

If an Application program has embedded SQL statements we have to precompile before going to translation.