Christian Grothoff
christian@grothoff.org

http://grothoff.org/christian/

1These slides are based in part on materials provided by IBM’s Academic Initiative.
Example of online processing: a travel agency

A travel agency would use online processing for:

• employee and customer information
• contacts with car rental companies
• hotels
• airline schedules

In contrast to batch processing, changes must be reflected to end-users immediately (in real-time).
Example of online processing (continued)
Example of online processing (continued)
Similarities: Online Systems and OSes

- Managing and Dispatching tasks
- Controlling user access
- Managing use of memory
- Managing concurrency of data
- Providing device independence
Characteristics of transactional systems

- Many users
- Repetitive
- Short interactions
- Shared data
- Data integrity
- Low cost / transaction
Christian Grothoff

Terminology

- Multitasking
- Multithreading
- Thread
- Reentrancy
- Commit and roll back
Transactional System Requirements

- Atomicity
- Consistency
- Isolation
- Durability

Commonly known as ACID.
Atomicity

- Either all of the updates performed by the transaction are performed or none of them are.

- Example: if a bank transfers money from one account to another, it must not just withdraw or just deposit, both tasks must either succeed or fail.
Consistency

• Databases often have consistency rules

• Example: (global) uniqueness of certain keys, up-to-date indices, etc.

• The consistency property requires that the database remains in a consistent state after the transaction after the transaction is over (assuming it was in a consistent state before)

• Database consistency must be preserved if the transaction was successful as well as if it was aborted!
Isolation

- In most database systems, multiple transactions are executed concurrently
- One transaction may read values that another transaction is updating
- Isolation requires that transactions must not see the intermediate state of other transactions.
- Example: if I transfer $500 from one of my accounts to another and a bank manager queries my total account balance at the same time, the result must not be $\pm$ $500 of my net balance.
Durability

• After a transaction has reported success, no kind of (conceivable) system failure may undo the transaction.

• In particular, this means that the database must guarantee that no other (concurrent, committing) transactions may violate the consistency or isolation properties of the transaction.

• Since I/O operations on the actual tables are usually buffered, databases use an (append-only) transaction log that can be played back to recover committed transactions for which not all table updates had been fully written.
Two-phase commit

INITIATOR
Update local resources
Prepare
Commit
SYNCPOINT

Agent of A
Update local resources
Prepare
Receive
Commit
SYNCPOINT

Agent of B
Update local resources
Prepare
Receive
SYNCPOINT

A
B
C

Phase 1
Phase 2
The Resource Recovery Services (RRS)

RRS is the syncpoint coordinator that is built into z/OS.

- Before changes happen, the Unit of Recovery (UR) is *in-reset*
- While the UR makes changes, it is *in-flight*
- Once a commit request has been made, the UR is *in-prepare*
- After the syncpoint coordinator (RSS) at the *atomic instant* makes a decision, the UR is either *in-commit* or *in-backout*
The Customer Information Control System (CICS)

CICS is a subsystem of z/OS which:

- controls transactions to run online applications
- manages the sharing of resources: many users using the same application(s) at the same time
- protects integrity of data
- prioritizes execution as needed

CICS is also available for other platforms.
CICS in a z/OS system
Languages Supported by CICS

- COBOL (including OO COBOL)
- C/C++
- Java
- PL/1
- Assembler
Platforms Supported by CICS

- z/Series (z/OS, OS/390, VSE)
- AIX
- HP-UX
- Solaris
- Windows
CICS features

- Task Control
- Basic Mapping Support
- File Control
- Program Control
- Storage Control
- Transient Data Control
- Journal Control (logging)
- Trace Control
- Dump Control
- Interval Control
Example: CICS Web Support
Compiling (COBOL) for CICS

Source

Translated Program

Compiler

Object Module

LinkEdit

Load Module

Program Library

//DFHRPL DD DSN=CICSV3.SDFHLOAD1,DISP=SHR
// DSN=CICSV3.SDFHLOAD2,DISP=SHR
// DSN=CICSV3.ULOADLIB,DISP=SHR
CICS Programming commands

CICS commands are embedded into the normal source code:

EXEC CICS
  function option option ...  
END-EXEC.

The specific syntax depends a bit on the language used.
Example CICS Programs

We will now see an example showing the use of CVDAs and the DFHVALUE function. The code:

- Tests whether the file named PAYROLL is closed
- If so, changes the UPDATE and DELETE option values for the file to UPDATEABLE and NOTDELETEABLE respectively (records can be updated but not deleted)
- DFHVALUE relates the internal representation of the status to the (human readable) value name
- Returns to CICS
Example: COBOL

WORKING-STORAGE SECTION.
01 FILE-STATUS-INFO.
   02 UOPST PIC S9(8) COMP.
   02 UUPD PIC S9(8) COMP.
   02 INFILE PIC x(8).
...
CICS-REQUESTS.
   MOVE 'PAYROLL ' TO INFILE.
   EXEC CICS INQUIRE FILE(INFILE)
      OPENSTATUS(UOPST) END-EXEC.
   IF UOPST = DFHVALUE(CLOSED)
      MOVE DFHVALUE(UPDATABLE) TO UUPD
   EXEC CICS SET FILE(INFILE)
      UPDATE(UUPD)
      NOTDELETEABLE END-EXEC.
EXEC CICS RETURN.
#define INFILE "PAYROLL "
main()
{
    long int mopst;
    long int mupd;

    EXEC CICS INQUIRE FILE(INFILE)
    OPENSTATUS(mopst);
    if (uopst == DFHVALUE(CLOSED) )
    {
        uupd = DFHVALUE(UPDATEABLE);
        EXEC CICS SET FILE(INFILE)
        UPDATE(uupd)
        NOTDELETEABLE;
    }
    EXEC CICS RETURN;
}
Example: PL/1

DCL (UPST,UUPD) FIXED BIN(31),
   INFILE CHAR(8);
...

INFILE='PAYROLL ';
EXEC CICS INQUIRE FILE(INFILE)
   OPENSTATUS(UOPST);
IF UOPST = DFHVALUE(CLOSED) THEN DO:
   UUPD = DFHVALUE(UPDATABLE);
   EXEC CICS SET FILE(INFILE)
      UPDATE(UUPD)
      NOTDELETEABLE; END;
EXEC CICS RETURN;
Example: Assembler

DFHE1STG
UOPST DS F *OPEN STATUS
UUPD DS F *UPDATE STATUS
INFILE DS CL8 *FILE NAME
...
MVC INFILE,=CL8’PAYROLL ’
EXEC CICS INQUIRE FILE(INFILE) X
OPENSTATUS(UOPST)
CLC UOPST,DFHVALUE(CLOSED)
BNE OPENLAB
MVC UUPD,DFHVALUE(UPDATEABLE)
EXEC CICS SET FILE(INFILE) x
UPDATE(UUPD)
NOTDELETEABLE
OPENLAB EXEC CICS RETURN
Basic Mapping Support (BMS)

BMS is used to interact with users using a terminal:

- EXEC CICS SEND – display a screen on the terminal
- EXEC CICS RECEIVE – read data from terminal
BMS Example

EXEC CICS
  SEND MAP ( 'ORCHM01' )
  MAPSET ('ORCHM01')
  ERASE
END-EXEC.

EXEC CICS
  RECEIVE MAP ('ORCHM01')
  MAPSET('ORCHM01')
  INTO (workstorage area)
END-EXEC.
Defining the screens

- BMS macros are a form of assembler language
- The “assembler” generates a physical map which:
  - builds the screen
  - merges variable data between program and screen
  - sends variables back to the program
The BMS Macros

BMS Maps are composed of three simple macros:

- **DFHMSD** – name of map set (one per file)
- **DFHMDI** – name of map (many per map set)
- **DFHMDF** – field screen definitions and location (many per map)

Source for these is on Marist in CICSTS13.CICS.SDFHMAC(DFHMxx).
Example

PRINT NOGEN
ORCHM01 DFHMSD TYPE=MAP,MODE=INOUT,CNRL=FREEKB,LANG=COBOL,TIOAPFX=YES
ORCHM01 DFHMDI SIZE=(24,80)
   DFHMDF POS=(01,01),LENGTH=01,ATTRB=(ASKIP,DRK,FSET), x
      INITIAL=’1’
   DFHMDF POS=(01,25),LENGTH=3,ATTRB=(ASKIP,BRT), x
      INITIAL=’PURCHASE ORDER --- FILE INQUIRY’
   DFHMDF POS(03,30),LENGTH=13,ATTRB=ASKIP, x
      INITIAL=’ORDER NUMBER’
ORDER#  DFHMDF POS=(03,44),LENGTH=10,ATTRB=(NUM,BRT,IC)
   DFHMDF POS=(04,32),LENGTH=11,ATTRB=ASKIP,INITIAL=’DEPARTMENT’
   ...
   DFHMSD TYPE=FINAL
Example User Screen

Type a department number and press enter.

Department number: A02

Average salary($): 58211.58

F3: Exit
CICS transaction flow (1/3)
CICS transaction flow (2/3)
CICS transaction flow (3/3)
Conversational Style

Conversational:

User Types Inputs

Menu
Enter account ______
Function code______

Menu
Enter account 1234_
Function code M____

Record Update
Enter account 1234
Name: Smith
Amount: $10.00
Date: 05/28/04

Menu
Enter account ______
Function code________
"Update confirmed"

PROGV000

SEND MAP
WAIT

RECEIVE MAP
READ FILE UPDATE

SEND MAP
WAIT

RECEIVE MAP
REWRITE FILE

SEND MAP
RETURN

SEND MAP
WAIT

SEND MAP
Pseudo-Conversational Style

**Pseudo-Conversational:**

**User Types Inputs**
- **Menu**
  - Enter account ______
  - Function code ______

**User Types Changes**
- **Record Update**
  - Enter account 1234
  - Name: Smith
  - Amount: $10.00
  - Date: 05/28/04

**PROGV000**
- SEND MAP...
- RETURN TRANSID(V001)....

**PROGV001**
- RECEIVE MAP...
- READ FILE...
- SEND MAP...
- RETURN TRANSID (V002)....

**PROGV002**
- RECEIVE MAP...
- READ FILE UPDATE....
- REWRITE FILE....
- SEND MAP...
- RETURN TRANSID (V000)....
CICS Program Control Constructs
The CICS Communication Area (COMMAREA)

• Area is automatically provided to you via the translator phase

• Part of the DATA DIVISION / LINKAGE SECTION in COBOL

• Initially provides one byte to be used as a status word

• Can range up to 32k in size
Example: COMMAREA in COBOL

01 DFHCOMMAREA.
   05 PROCESS-SW PIC X.
      88 INITIAL-ENTRY VALUE '0'.
      88 VERIFICATION VALUE '1'.
   05 ACCOUNT-NUMBER PIC X(10).

   ...

EXEC LINK PROGRAM(ACCTPGM)
   COMMAREA(DFHCOMMAREA)
   LENGTH(11)

END-EXEC.

CICS Programming Roadmap

- Design application
- Write & compile & test program
- Define program & transaction in CICS resources
- Define other resources (files, queues, etc.) in CICS resources
- Make resources known to CICS

The Information Management System (IMS)

IMS is another application used for online processing consisting of three components:

- Transaction Manager (TM)
- Database Manager (DB)
- System services (common services for TM and DB)
IMS Overview

 IMS System

 Transaction Manager       Database Manager

 IMS Message Queues

 IMS Databases

 IMS Logs

 z/OS Console
IMS Transaction Manager Messages

An operator of an IMS terminal can send four kinds of messages to IMS TM; the destination determines the kind of message:

**Another terminal** Logical terminal name in the first 8 bytes

**Application program** Transaction code in the first 8 bytes

**IMS TM** A “/” (slash) in the first byte (command for IMS TM)

**Message switch service** Messages for the Advanced Program to Program Communication (APPC) feature of IMS

---

IMS APPC

z/OS

IMS
Application Program

APPC

UNIX

Application Program
Questions