IMS Application Programming Techniques

Kenny Blackman
kblackm@us.ibm.com
IBM

8/02/2010
Session
Device and Data Independence

Application program

MASK

AIB
Return/Reason Code

PSB

TP
I/O-ALT
PCB
Status Code

MASK

DATA BASE
PCB
Status Code

DEVICE

DBD
NAME
ADDRESS
PAYROLL
Application Interface Block (AIB)

- An application program can refer to a PCB by a given NAME, not an address (PCBNAME is 8 bytes).
  - For the I/O-PCB, the name is 'IOPCBbbb'
  - For DB-PCB, the name is specified in the PSBGEN:
    - PCBNAME=... parameter on PCB macro
    - LIST=Y|N - Display PCBNAME in PSB listing?

Most DL/I calls can be issued in two ways:

Using a PCB (standard technique):
  CALL xxxTDLI ( <count>,FUNC,PCB,I/O AREA,... )
  CEETDLI

Using an AIB (new technique):
  CALL AIBTDLI ( <count>,FUNC,AIB,I/O AREA,... )
  AERTDLI
  CEETDLI
Alternate PCBs

- I/O PCB
  - RECEIVE

- ALTERNATE Response PCB
  - LTERM=Lterm B
  - Lterm B
  - Device A
  - Lterm A

- ALTERNATE Express PCB
  - Device C

- Modifiable ALTERNATE PCB
  - PROGRAM D
  - PROGRAM D
  - Device C
# IMS Application Program Runtime Environments

## IMS TM Control Region (CTL)

### Functions
- Queuing
- Scheduling
- Logging
- I/O
  - Data Base
  - Terminal

<table>
<thead>
<tr>
<th>IMS TM Control Region (CTL)</th>
<th>Application Programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUPPORTED BY IMS TM/DB CTL</td>
<td>STAND ALONE</td>
</tr>
<tr>
<td>Message Region (MPP, IFP, JMP)</td>
<td>DB Batch Region (DLI)</td>
</tr>
<tr>
<td>Batch Message Driven</td>
<td>TM Batch Region (DB2)</td>
</tr>
<tr>
<td>Process (BMP)</td>
<td></td>
</tr>
<tr>
<td>Batch Non-Message Driven</td>
<td></td>
</tr>
<tr>
<td>Process (BMP, JBP)</td>
<td></td>
</tr>
</tbody>
</table>

### Supported by IMS TM/DB CTL

<table>
<thead>
<tr>
<th>Function</th>
<th>IMS</th>
<th>User</th>
<th>Some</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheduled By</td>
<td>IMS</td>
<td>YES</td>
<td>SOME</td>
<td>SOME</td>
</tr>
<tr>
<td>Online DB's</td>
<td>YES</td>
<td>YES</td>
<td>Programs</td>
<td>YES</td>
</tr>
<tr>
<td>OS/VS Files</td>
<td>NO</td>
<td>YES</td>
<td>Programs</td>
<td>YES</td>
</tr>
<tr>
<td>MSG Q</td>
<td>YES</td>
<td>YES</td>
<td>Changeable</td>
<td>NO</td>
</tr>
<tr>
<td>I/O PCB</td>
<td>YES</td>
<td>YES</td>
<td>Changeable</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Message Queuing

INPUT

TRAN

MESSAGE 1
Device A

MESSAGE 2
Device B

MESSAGE 3
Device A

MESSAGE 4
Device C

OUTPUT

REPLY 1
A

REPLY 2
B

REPLY 4
C

APPC

IMS
Connect
Message Format Service
Call Sequence

GET UNIQUE

GET NEXT

PROCESS MESSAGE

INSERT

GO GET MESSAGE

GO BACK/RETURN IF NO MORE

TRANCODE FIRST SEGMENT

SEGMENT 2

SEGMENT 1

SEGMENT N
Problem W/O Conversational

Processing requirement

Interactive program
- DATA
- ADDITIONAL REQUESTED DATA
- Device A

PGM A

Multi-screen input
- DATA 1
- DATA 2
- DATA 3
- Device B

PGM B

Need temporary storage

Solution 1
- USE SCREEN?
  1. Data lost if screen cleared
  2. More data sent over comm lines

Solution 2
- UPDATE TEMP. DB?
  OR
- UPDATE PERM. DB?
  1. Complicates program logic
  2. Multiple db retrievals and updates
  3. Input error, pgm abend
     - Scratch temp db record
     - Backout perm db record
Conversational Solution - Scratch Pad Area

Device B
- NAME: JOHN DOE
- ADDRESS: 123 MAPLE CHIC ILL 60999
- ACCT: 12-3456-789

ACCOUNT OK
- ITEM
- QTY
- DESC
- 1
- THING
- 2
- BLUE
- 2
- 3

ORDER ENTRY PROGRAM

MESSAGE REGION

S.P WORK AREA

CONTROL REGION

SPA
- JOHN DOE...56-789
- THING..2 BLUE
- 10.00 ... .50 11.00

ITEM
- PRICE
- SHIP
- 1
- 10.00
- .50
- SUB
- 10.00
- TAX
- .50
- SHIP
- .50
- TOTAL
- 11.00
Conversational Call Flow

GET UNIQUE
GET NEXT
DETERMINE WHAT TO DO THIS TIME
PROCESS
INSERT
INSERT
GO GET ANOTHER CONVERSATION

SCRATCH PAD AREA

ll | zzzz | trancode | user data
---|------|----------|-------------
2  | 4    | 8        | variable

FIRST SEGMENT OF A CONVERSATIONAL TRANSACTION

ll | zz | data (no transaction code)
---|----|------------------------
2  | 2  | variable

Note Application Program cannot modify the first 6 bytes of the SPA (LL and ZZZZ). IMS TM uses these fields to identify the SPA.
Programming Tips

• All programs should be written to go back and get another message (GU I/O PCB) ('QC' status if none exist)
• Data base requests should be handled with the fewest number of calls
• Message processing programs should not do lengthy data base scans or updates
• Never issue file open/close operations - beware of COBOL "displays" and PL/I "put datas"
• Issue all message segment gets/inserts as consecutive calls if possible
• The PSB should be completely accurate in procopt's selected for each segment type
• Large, single segment replies take fewer calls and less overhead (check system standard for maximum size)
• Do not issue GN calls to the i/o pcb for single segment messages
IMS Managed Service Flow Program-to-Program Switch

Device A

MSG-Q

IMSA-PGMA
GU IOPCB
Process Data
ISRT ALTPCB
UOW1

MSG-Q

IMSA/IMS(B(MSC)
TRNBP/GM-B
GU IOPCB
Process data
ISRT IOPCB
UOW2
WebSphere Process Server Managed Service Flow

Device A

WebSphere Process Server
BPEL

SCA Service
IMS TM Resource Adapter

IMS Connect

IMS MSG-Q

IMSA-PGMA
GU IOPCB
ISRT ALTPCB
UOW1a

IMSA-PGM-B
GU IOPCB
ISRT IOPCB
UOW1b

Global Transaction

UOW1

SHARE in Boston

15
IMS Enterprise Suite Soap Gateway
Business Events

WebSphere Business Events/
Monitor

Device A

WebSphere Business Events/
Monitor

IMS Soap Gateway

IMS Connect

IMSA-PGMA
GU IOPCB
ISRT ALTPCB

Event

DATA

WebSphere Business Events
WebSphere Business Monitor

SHARE in Boston
Asynchronous Model

Device A

IMS-ApplT1
GU IOPCB
ISRT IOPCB
Message + appl-token

IMS-ApplT2
GU IOPCB
Message + appl-token
ISRT IOPCB

MSG-Q

Device B

READ_DATA
Message + appl-token

PROCESS_DATA

SEND_DATA
Message + appl-token
IMS Asynchronous via OTMA IMS Connect

Device A

IMS-Appl T1
GU IOPCB
Message+
Appl-token
ISRT ALTPCB

Device B

TCP/IP-B
IMS TM RA,
IMS SOAP Gateway or
RYO
Resume_TPIPE

IMS Connect
Message
Socket

IMS-Appl T2
GU IOPCB
Message+
appl-token
ISRT IOPCB

UOW1
UOW3
UOW2
WebSphere MQ Asynchronous via OTMA IMS Bridge

Device A

IMS-ApplT1
GU IOPCB
ISRT ALTPCB

Device B

OTMA
DFSYPRX0
TMEMBER
TPIPE
SEND_DATA

Queue
messages

Queue
messages

MSA/IMSB Appl-T2

MB
MQGET

MQCON
MQPUT
MQDISC

UOW1
UOW3
UOW2
WebSphere MQ via ESAF Asynchronous

Device A

IMSA-PGMA GU IOPCB
MQCONN MQ-B
MQPUT – MQPMO_Syncpoint
MQDISC
ISRT IOPCB

UOW1

Queue

messages

Device B

MB
MQGET
MQDISC
UOW2

SHARE in Boston
Synchronous Model

Device A
- MSG-Q
- IMS-AppIT1
  - GU IOPCB
  - Send Message
  - WAIT
  - Receive Message
  - ISRT IOPCB

Device B
- Message
- READ_DATA Message
- PROCESS_DATA
- SEND_DATA Message
CPI-C/APPC Synchronous

Device A

MSG-Q

- IMSA-PGMA
- GU IOPCB
- CMALLC Allocate TP-B
  Type=NONE/CONFIRM/SYNCPT
- CMSEND SEND_DATA
- CMRCV RECEIVE_and_WAIT
- CMDEAL DeAllocate

ISRT IOPCB

Device B

- IMSA/IMSB
  LU6.2
  TP-B
- CMRCV RECEIVE
- CMSEND SEND_DATA
- CMDEAL DeAllocate

Conversation

Session

UOW1
User Written TCP/IP Synchronous

Device A

IMSA-PGMA
GU IOPCB

Socket
Connect - TCP/IP-B
WRITE_DATA

WAIT

READ_DATA
CLOSE
ISRT IOPCB

Connection

datagrams

Device B

TCP/IP-B
READ_DATA
Socket
Connect - TCP/IP-C

READ_DATA
WRITE_DATA
CLOSE

UOW1

UOW2
WebSphere MQ via ESAF Synchronous

Device A

IMSA-PGMA
GU IOPCB

MQCONN MQ-B
MQPUT NO-SYNCPPOINT

WAIT

MQGET (NO)SYNCPPOINT

MQDISC
ISRT IOPCB

Queue

Device B

MB
MQGET
MQPUT
MQDISC

messages

UOW1

UOW2
DB2 Stored Procedure via ESAF Synchronous

EXEC SQL:
CALL PROC ( :aaa : bbb ...)

IMS Managed UOW1

User Defined Function (UDF)

Web Service

ISRT IOPCB

DB2 SP

IMS-A-PGMA
GU IOPCB

Device A

data
IMS Java and DB2 Attach Facility

Device A

IMSA-PGMA
IMS-Java JMP/JBP JDBC GU IOPCB:
CALL PROC ( :aaa : bbb ...
data
ISRT IOPCB

RRS Managed

UOW1

Web Service

User Defined Function (UDF)
**IMS TM MPP - JVM**

IMS 10 Apar PK82214  
LE Apar PK99010  
DB2 Apar PK93123 – use ESAF no RRS
XML DB Highlights - Decomposed data

- Retrieve - Compose XML document from any existing traditional database
- Insert - Decompose XML docs back into same DB
- Same data can be read by existing IMS applications
XML DB Highlights - Intact Data

- Insert/Retrieve/Delete new XML documents **INTACT** in new IMS databases
- **Intact data** is not expected to be understood by other IMS applications
  - XML Documents span IMS segments
  - Stored in Unicode
IMS Application Programming techniques

• Summary
  • IMS Application Programming provides simple device/data independent model
    • For online processing
    • For Batch Processing