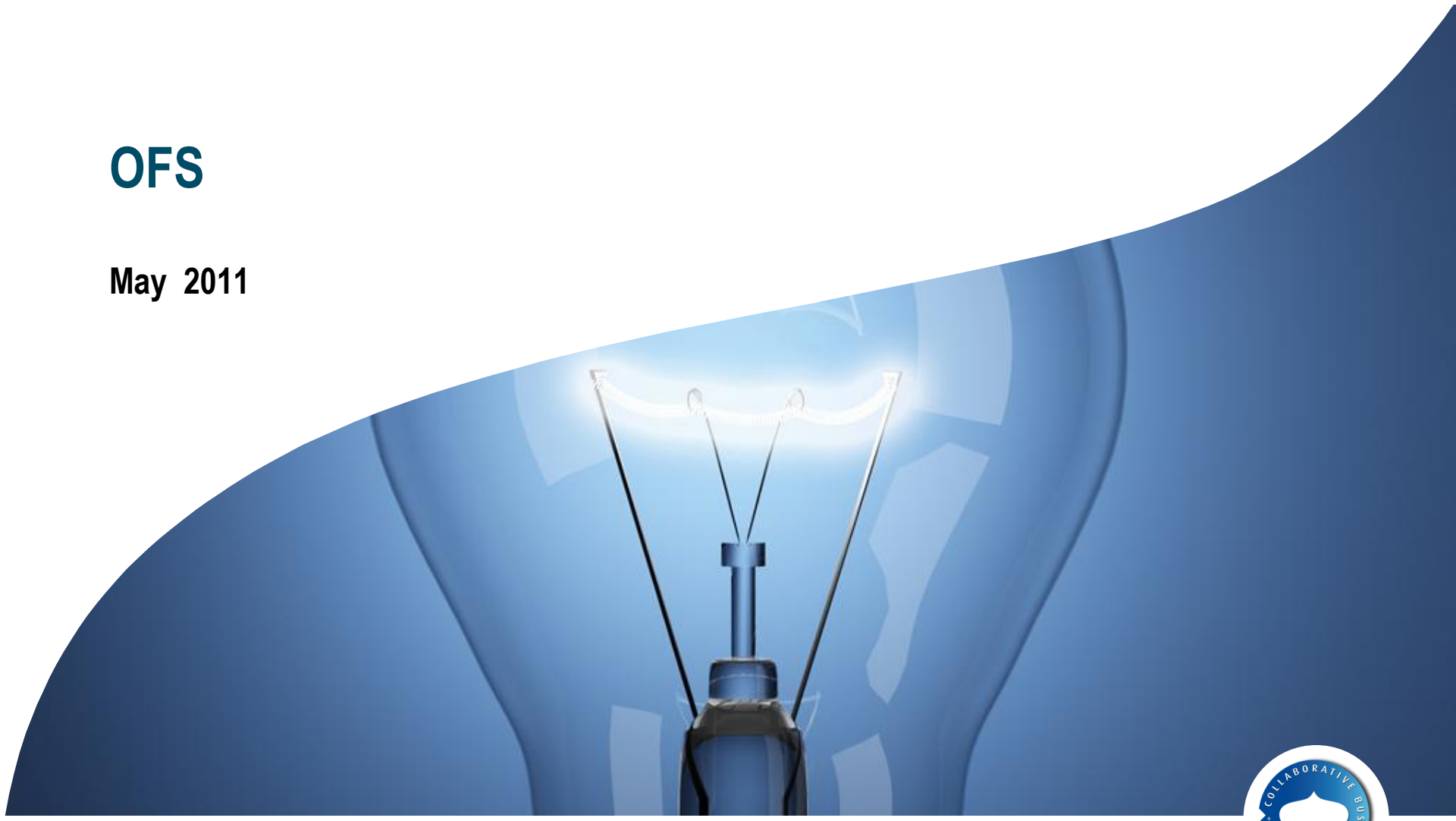
Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

OFS

May 2011



TOGETHER. FREE YOUR ENERGIES

Day-wise Plan

Session - I

- What is OFS
- Why OFS
- Functionality of OFS
- Architecture of OFS
- Components of OFS
- Processing modes of OFS

Session - II

- Applications involved in OFS
- Message Structure
- Messages Types

Day-wise Plan

Session - III

- Online OFS sample
- Batch OFS sample
- ENQUIRY using OFS

Session - IV

- Subroutine call using OFS sample
- Data capture using OFS sample
- Error Messages in OFS

Objective

At the end of this session, participants will

- Understand the message structure in OFS
- Usage of OFS in various applications like TC connector and for Interfaces

Introduction

What is OFS?

- The Open Financial Service module (OFS) provides an interface to allow the update and interrogation of T24 applications
- Any file can be updated using OFS
- OFS is an interface that allows transactions and queries request to be processed by T24

Introduction

- Data available in external source can be processed
- Also enquiries related application can be retrieved using OFS
- Can be used in various external applications like ATM, Internet Banking

Why OFS?

OFS is the only standard gateway to Temenos database

OFS syntax are understandable by Temenos database

Request for enquiries and updation of tables are done through the OFS formatted messages

Interface takes care of running any OFS business function which helps to avoid direct user intervention

Functionality of OFS

Supports multi-company processing

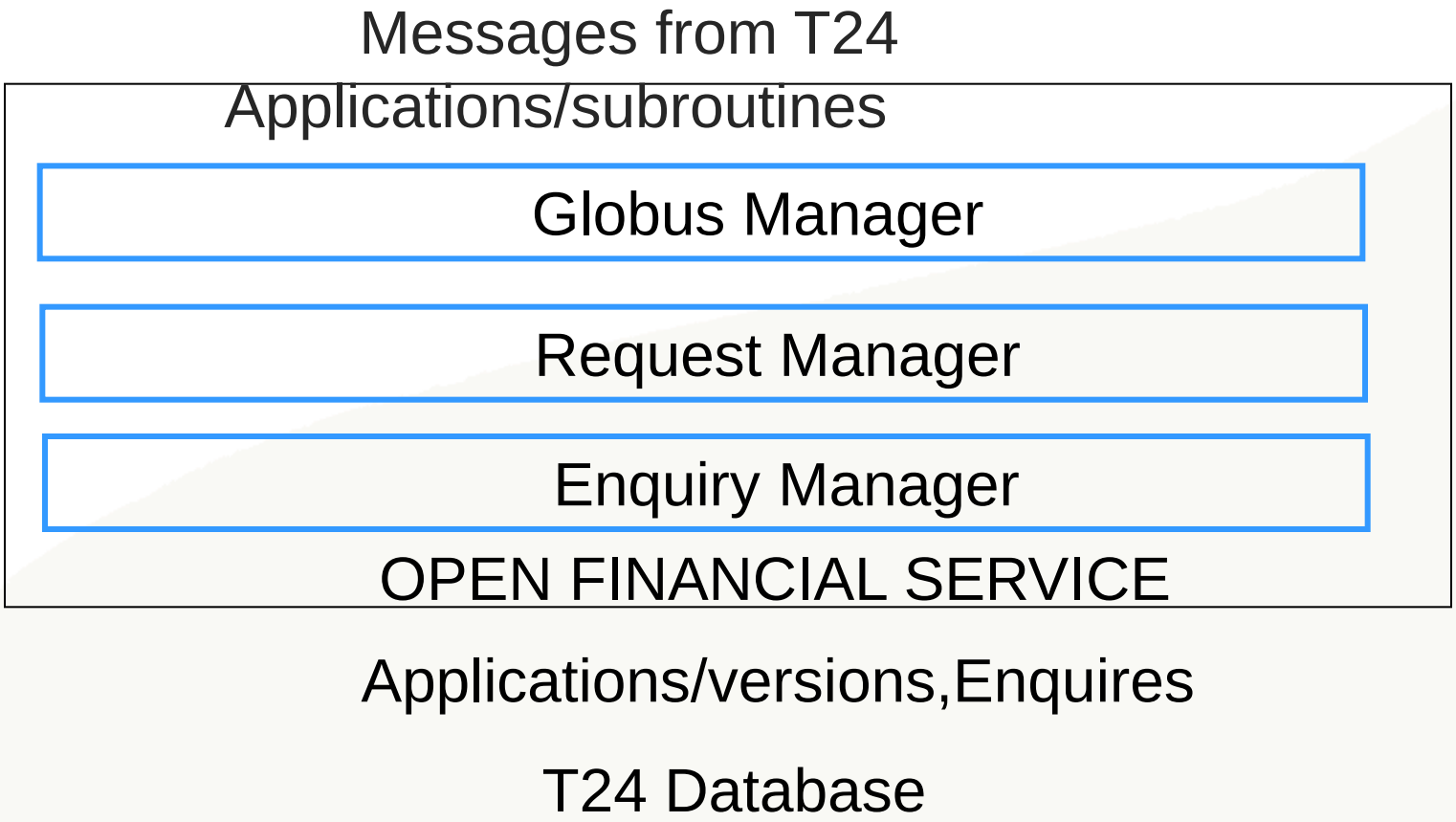
Error messages handling

Platform independent

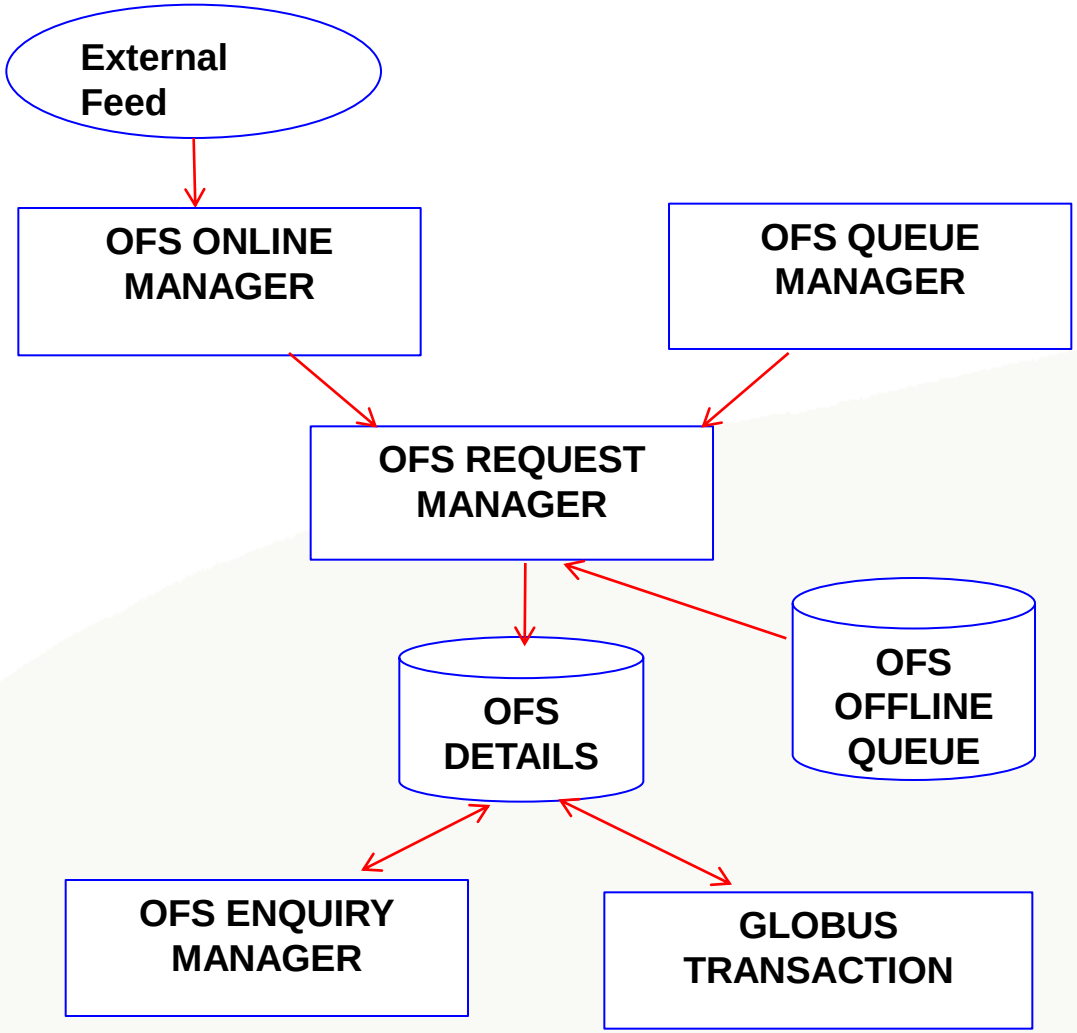
Support for sub-values

The incoming messages can be customized by user defined routines

Architecture



Architecture



Components of OFS

- OFS QUEUE MANAGER
- OFS ONLINE MANAGER
- OFS REQUEST MANAGER
- OFS CONNECTION MANAGER
- OFS SESSION MANAGER

Components of OFS – OFS QUEUE MANAGER

OFS.QUEUE.MANAGER – is used for Batch communication with the OFS module

It also supports processing of offline messages (in batch mode)

Writes the reply for the processed message to a file or directory depending on the configuration done in the related OFS.SOURCE record

It also performs message logging depending on the level setup in related OFS.SOURCE record

Components of OFS – OFS ONLINE MANAGER

OFS ONLINE MANAGER – Receives messages from the external application (e.g. ATM Switch software) or user via a TELNET link

Passes the request message to the OFS.REQUEST.MANAGER to perform the required processing

Passes the reply message returned to the external application

Components of OFS – OFS REQUEST MANAGER

Receives the request message from one of the OFS module routines (subroutine call)

Calls the appropriate message parser based on the format of the message

Validate syntax structure of OFS

Components of OFS – OFS REQUEST MANAGER

Identifies the type of request (Application/enquiry)

Calls the subroutine given in the request if it is found to be a Subroutine request

Also passes the reply message to the caller application/subroutine

Components of OFS – OFS CONNECTION MANAGER

Connector uses Connection manager to send and receive messages from Temenos 24

Reads the records from OFS.SOURCE

Components of OFS – OFS SESSION MANAGER

Handles OFS requests coming from Temenos T24 Browser clients, forwards them to Temenos T24 server for processing and send back the response

Handles Browser sessions create/destroy and manage operations

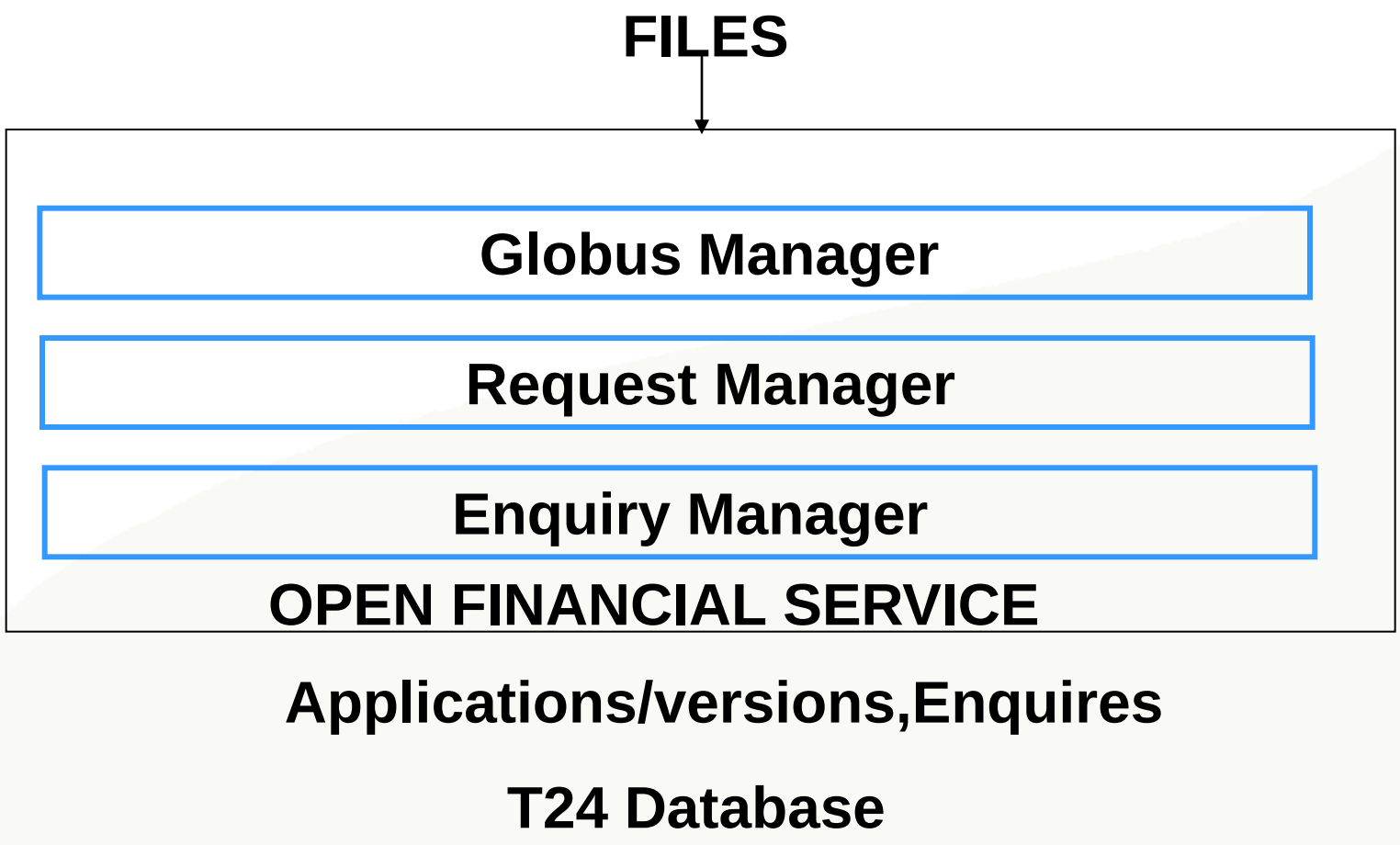
Parses xml messages

Processing Modes in OFS

The following are various processing modes in OFS

- BATCH MODE
- ONLINE MODE
- GLOBUS MODE

BATCH MODE


FILES
Globus Manager
Request Manager
Enquiry Manager
OPEN FINANCIAL SERVICE
Applications/versions,Enquires
T24 Database

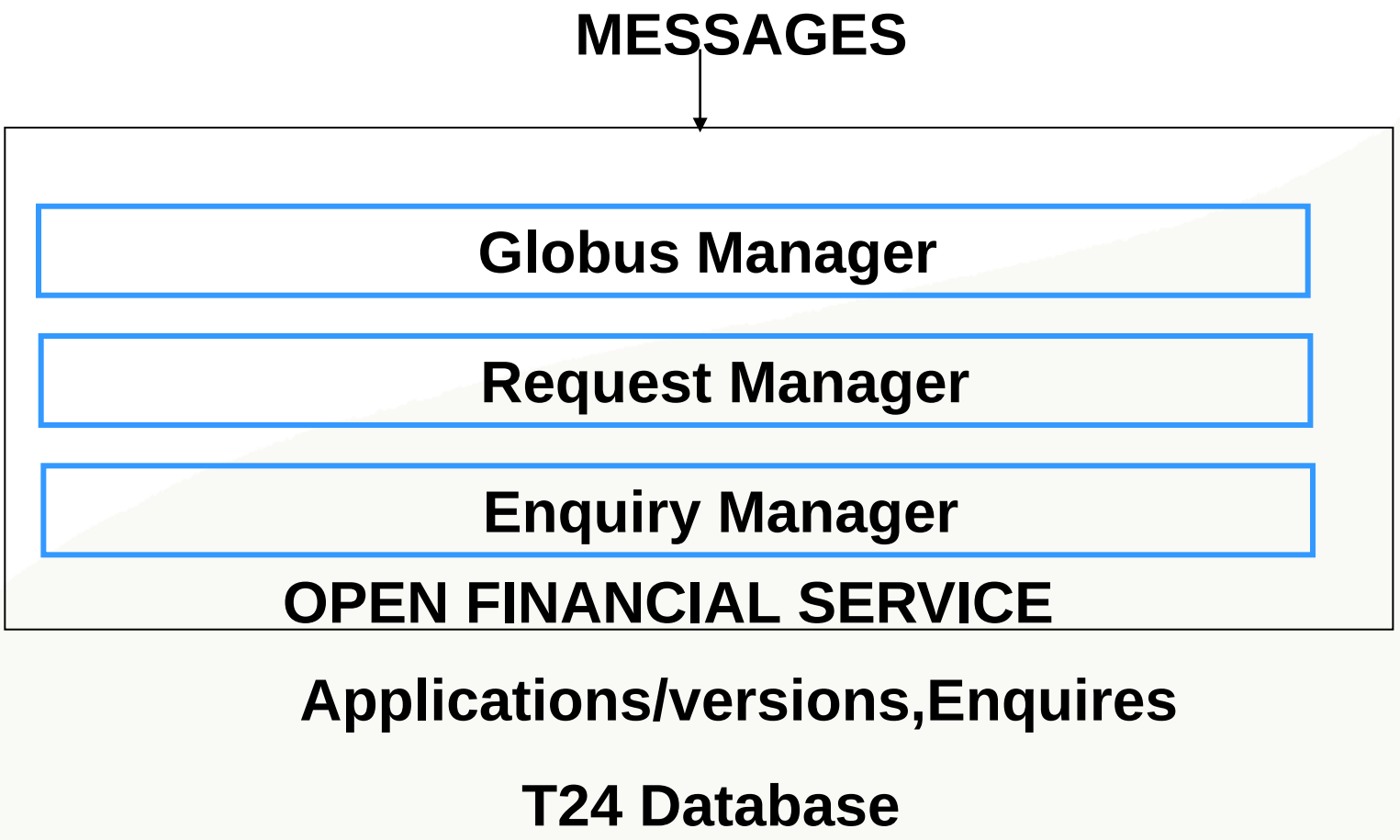
BATCH MODE

Files contain Batch of data

In case of Offline processing messages are picked up from queue files

In OFS Batch mode the OFS Queue manager runs the phantom process
Batch mode is controlled by both OFS.SOURCE and EB.PHANTOM

ONLINE MODE


MESSAGES
Globus Manager
Request Manager
Enquiry Manager
OPEN FINANCIAL SERVICE
Applications/versions,Enquires
T24 Database

ONLINE MODE

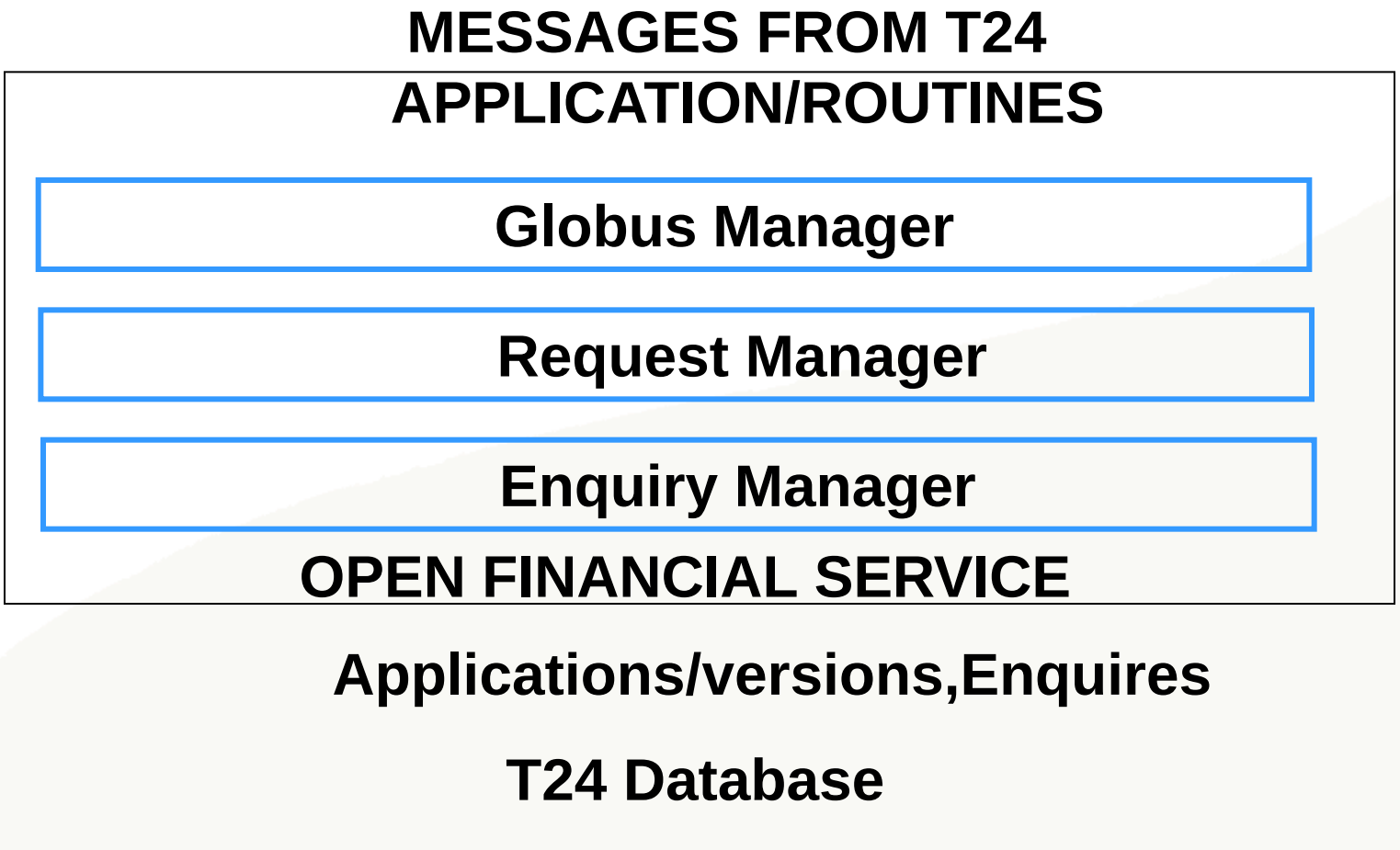
OFS online mode allow real time update to Temenos 24 via a live TELNET

Messages are accepted and processed individually

When OFS is set to work in online mode, the program
EB.AUTO.INIT.PROCESS is executed upon login

When EB.AUTO.INIT.PROCESS is launched it looks for a record on the EB.AUTO.PROCESS file with an id equal to the UNIX / Windows login id of the user logging in.

GLOBUS MODE


MESSAGES FROM T24
APPLICATION/ROUTINES
Globus Manager
Request Manager
Enquiry Manager
OPEN FINANCIAL SERVICE
Applications/versions,Enquires
T24 Database

GLOBUS MODE

To setup Globus mode an OFS.SOURCE record has to be created

Globus mode does not make use of phantom

Request and responses are shared using subroutine call

For releases T24 and below, this call can be made from a version/version control subroutine or an application if you are building the application or from a custom-built T24 routine

GLOBUS MODE

From TEMENOS T24 R5 release onward, OFS.GLOBUS.MANAGER routine cannot be called from a VERSION / VERSION.CONTROL routine

A check has been made in EB.CALL.API to prevent this call, with a view to ensure integrity of the updates

Applications in OFS

OFS.SOURCE

EB.PHANTOM

OFS.REQUEST.DETAIL

Applications of OFS - OFS.SOURCE

This is required to be set up for OFS processing for TELNET

Before passing the message through telnet, we have to ensure whether OFS.SOURCE is set up or not.

To set up OFS.SOURCE

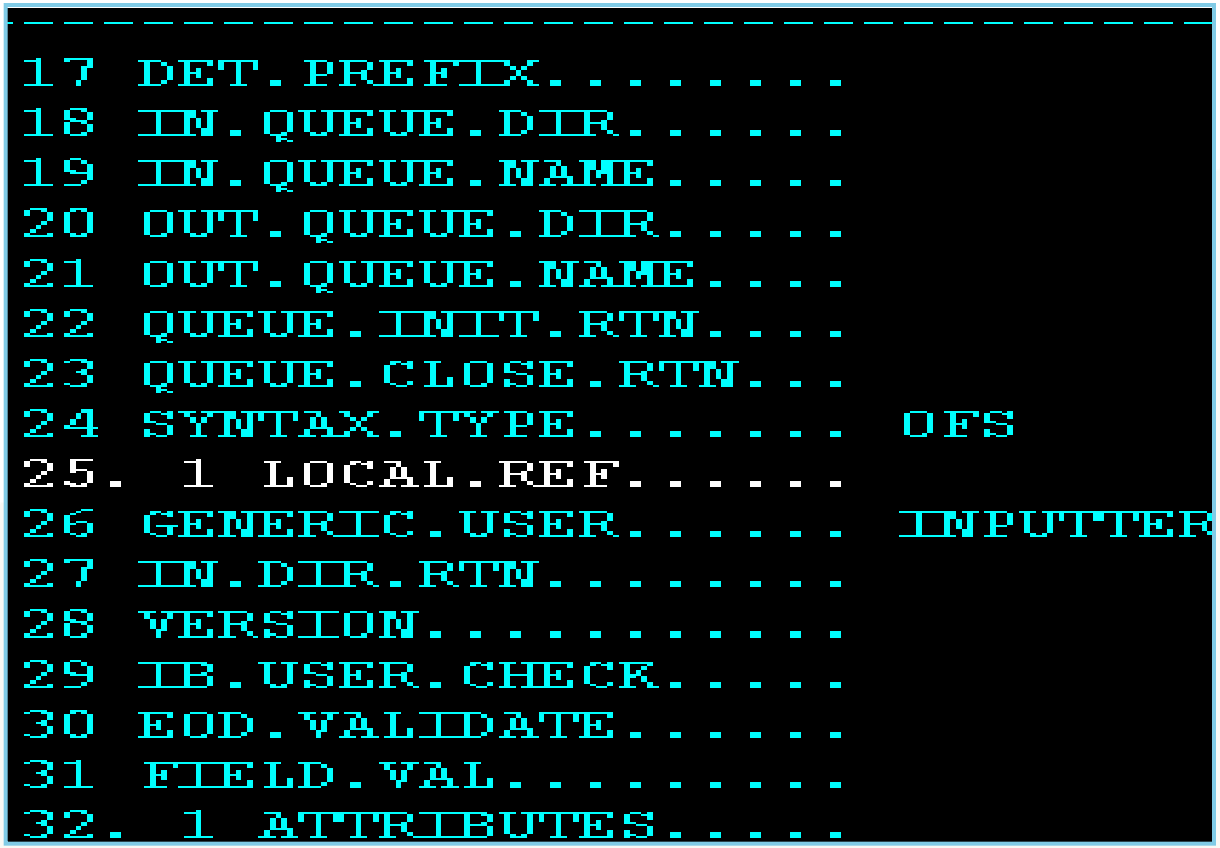
- Open OFS.SOURCE application
- Create an id in OFS.SOURCE application
- Fill up all the required fields in the OFS.SOURCE application.

OFS.SOURCE cont …..

OFS.SOURCE – FOR ONLINE

```
     SOURCE.NAME.....  TEST
-------------------------------
  1  DESCRIPTION.....  TEST
  2  SOURCE.TYPE.....  TELNET
  3.  1  LOGIN.ID.....  ANY
  4.  1  EB.PHANT.ID....
  5  MAX.CONNECTIONS...
  6  RESTRICT.LINK.....
  7  INITIAL.ROUTINE...
  8  CLOSE.ROUTINE.....
  9  IN.MSG.RTN........
 10  OUT.MSG.RTN.......
 11  MSG.PRE.RTN.......
 12  MSG.POST.RTN......
 13  LOG.FILE.DIR......
 14  LOG.DETAIL.LEVEL..  NONE
 15  OFFLINE.QUEUE.....
 16  MAINT.MSG.DETS....
```

OFS.SOURCE – FOR ONLINE



OFS.SOURCE

DESCRIPTION :

- This is a mandatory input field.
- Which specifies the description of OFS.SERVICE record defined in this record

SOURCE.TYPE :

- ONLINE/TELNET is another source type
- This is not most widely used by our consultants
- Mostly we are using for testing purpose to test the messages that we built is working fine or not

OFS.SOURCE - FIELDS

LOGIN ID :

- Specifies the UNIX/NT login
- This is a mandatory input if the SOURCE.TYPE is TELNET

EB.PHANTOM :

- We need to create a Phantom record and attach that id in this field

OFS.SOURCE

MAX CONNECTION:

- Specifies the maximum no of online OFS connections for the specified service which can be active at any one time (min input:1, max input: 99,999)

LOG DETAIL LEVEL :

- FULL means all the log details will stored in the log directory

OFS.SOURCE

DET PREFIX :

- The details will be prefixed with the value given in this field. SAMPLE
- The details of the record processed will be in OFS.REQUEST.DETAIL with the id prefixed with SAMPLE

INQUEUE DIR :

- The data to be processed will be stored in IN (OFSTEMPIN) directory

OFS.SOURCE

OUT.QUEUE.DIR :

- When u verify the EB.PHANTOM record the record from OFSTEMPIN will be moved to OFSTEMPOUT and the record will be created in FT File in INAU status

SYNTAX TYPE : OFS

PASSING MESSAGE USING ONLINE MODE

After setting up the OFS.SOURCE for online, We have to pass the OFS message from the jshell prompt

In the shell prompt pass the below command

Then follow the below OFS message structure to update the application

```
jsh _____, -->tSS TEST
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>2652</t24pid><t24ofssource>TEST</t24ofssource><clientIP/></tSS>
```

Message structure

OFS Message Structure has got 5 parameters each separated by ‘,’

- Operation
- Options (VersionName/Functions/Operation)
- Userid/password
- Transaction id
- Message to be passed

Message structure

OPERATION

- Operation specifies the Temenos application to be run by OFS E.g. ACCOUNT,FUNDSTRANSFER
- OFS transaction cannot be used update live files

OPTIONS

- T24 function to be used with the application (e.g. Input-I/Authorize-A/Delete-D/Reverse-R)
- If the function is not supplied then it is assumed as input

Message structure

USER INFORMATION

- User sign on name – Valid sign on name available in T24 database
- Password – Password of the sign name
- Company – In case of multicompany setup

Message structure

TRANSACTION ID

- The transaction id portion of the message structure contains the transaction number/key for the application that are run
- The transaction id technically the ID of record in the file used in the transaction

MESSAGE DATA

- The message data portion of the message structure contains the data required to create or update the transaction

Example of sample OFS Message

```
jsh          ~ -->tss TEST
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>2948</t24pid><t24ofssource>TEST</t24ofssource><clientIP/></tSS>
ACCOUNT,OFSACCOUNT,INPUTT/123456,,CUSTOMER::=100781,CATEGORY::=1001,CURRENCY::=USD
```

Applicati on name

Version Name

Username/pas sword

Message to be passed

Return message

Message is passed in OFS Format, the response is returned back to user to acknowledge the message

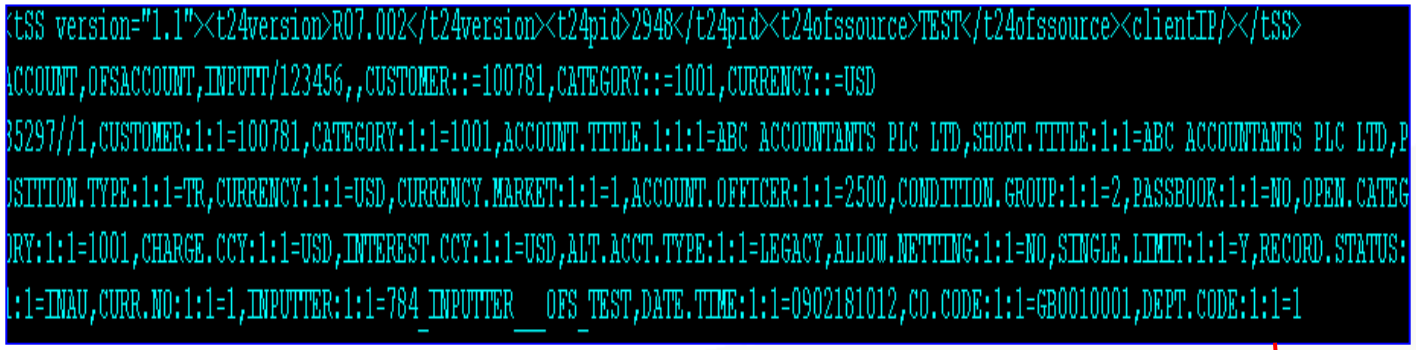
The Return message consists of two return codes

1 – Success

-1 – Failure

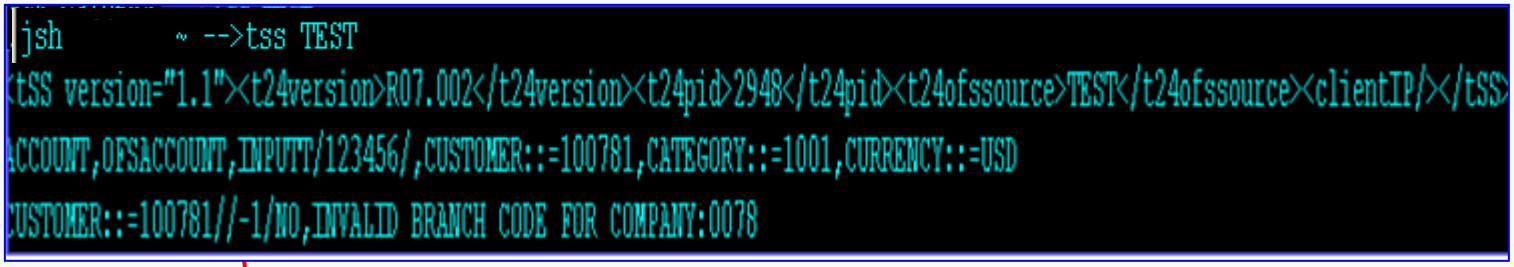
Sample return message

SUCCESS



success

FAILURE



failure

Workshop - I

Set up an OFS for ONLINE mode

Create a OFS.SOURCE record

Create a version for SECTOR

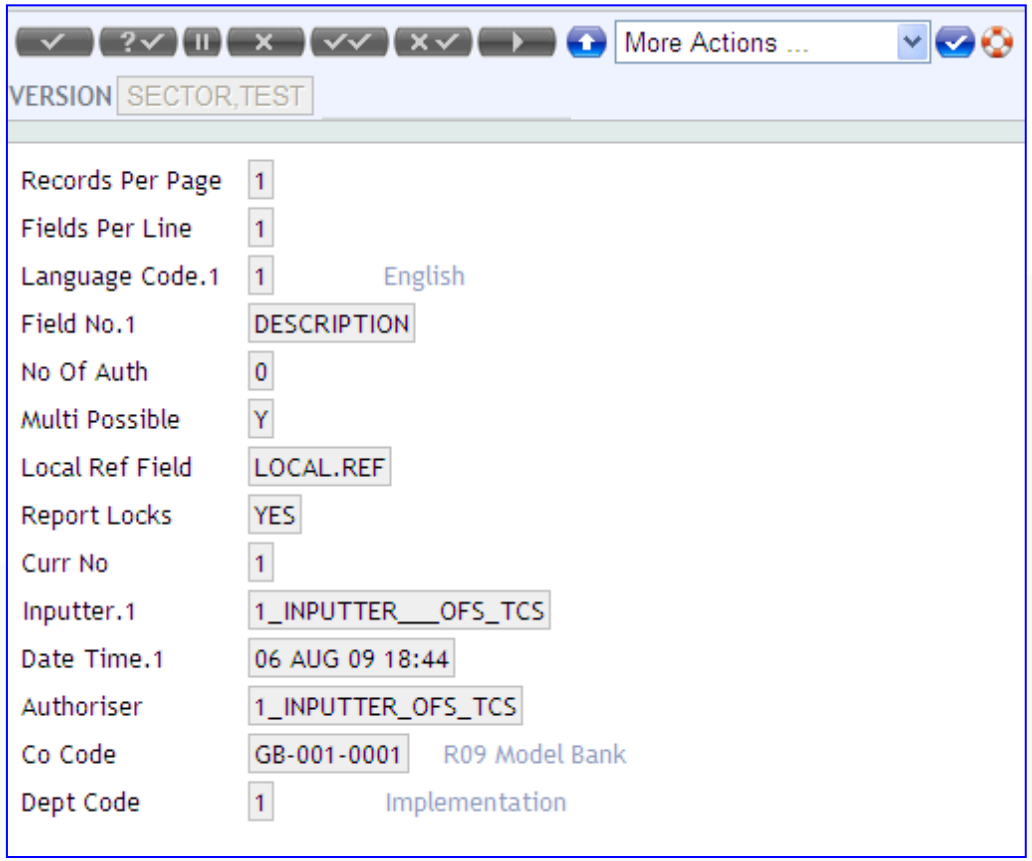
Update the DESCRIPTION field of the SECTOR through OFS

Solution – Step 1 : OFS ONLINE SETUP

```
    SOURCE.NAME..... TELNET
----------------------------------------
  1 DESCRIPTION..... TELNET OFS RECORD
  2 SOURCE.TYPE..... TELNET
  3.  1 LOGIN.ID..... t24demo
  4.  1 EB.PHANT.ID....
  5 MAX.CONNECTIONS... 10
  6 RESTRICT.LINK.....
  7 INITIAL.ROUTINE...
  8 CLOSE.ROUTINE.....
  9 IN.MSG.RTN........
 10 OUT.MSG.RTN.......
 11 MSG.PRE.RTN.......
 12 MSG.POST.RTN......
 13 LOG.FILE.DIR...... OFSLOG
 14 LOG.DETAIL.LEVEL.. FULL
 15 OFFLINE.QUEUE.....
 16 MAINT.MSG.DETS.... Y
```

```
 17 DET.PREFIX........ TEST
 18 IN.QUEUE.DIR......
 19 IN.QUEUE.NAME.....
 20 OUT.QUEUE.DIR.....
 21 OUT.QUEUE.NAME....
 22 QUEUE.INIT.RTN....
 23 QUEUE.CLOSE.RTN...
 24 SYNTAX.TYPE....... OFS
 25.  1 LOCAL.REF......
 26 GENERIC.USER...... GREAMELAIDLAW1
 27 IN.DIR.RTN........
 28 VERSION...........
 29 IB.USER.CHECK.....
 30 EOD.VALIDATE......
 31 FIELD.VAL.........
 32.  1 ATTRIBUTES.....
```

Solution – Step 2 : SECTOR-VERSION



Solution – Step 3 : OFS.CONNECTION.MANAGER

After setting up ‘OFS.SOURCE’ for Online updates, Initiate ‘OFS.CONNECTION.MANAGER’ from jsh prompt.

Syntax : OFS.CONNECTION.MANAGER <source name>

System prompts to enter OFS message

Enter OFS messages to be processed.

```
jsh jglobus ~ -->OFS.CONNECTION.MANAGER TELNET
<version>OFS VERSION 3.0K</version>
SECTOR,TEST/I/PROCESS,AUTH.1/123456,1000/,DESCRIPTION::= Financial,SHORT.NAME::=Financial
```

Solution - Return Message

After processing message , system will return a message from which we can identify that updates are successful or not.

In the above message, Transaction is updated successfully and Transaction details also returned.

```
1000/TEST0033500003/1,DESCRIPTION:1:1= Financial,SHORT.NAME:1:1=Financial,CURR.NO:1:1=4,INPUTTER:1:1=11_AUTH1___OFS_TELNET,DATE.TIME
:1:1=0603191628,AUTHORISER:1:1=11_AUTH1_OFS_TELNET,CO.CODE:1:1=US0010001,DEPT.CODE:1:1=1
```

EB.PHANTOM - FOR BATCH MODE

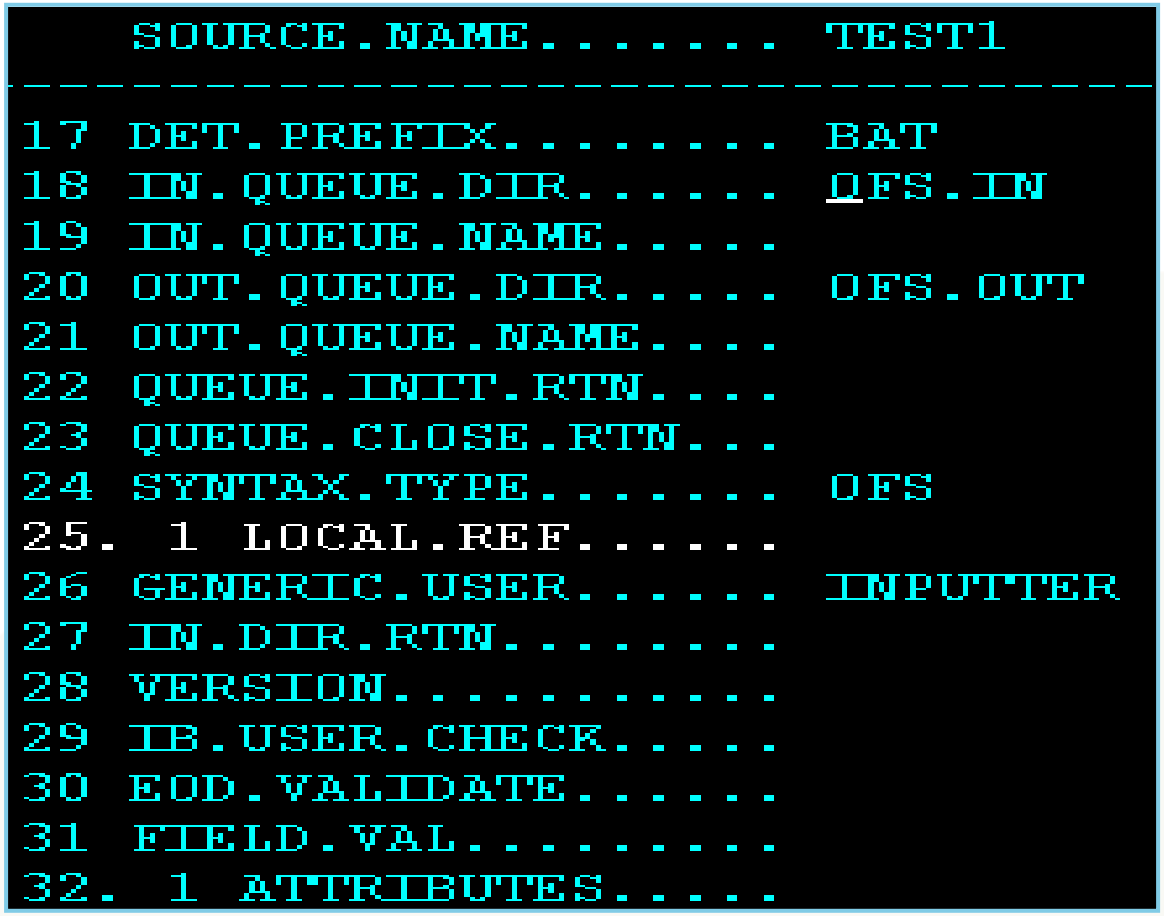
EB.PHANTOM to be set up when the OFS messages are to be processed using Batch mode

For OFS processing OFS.SOURCE and EB.PHANTOM has to be set up

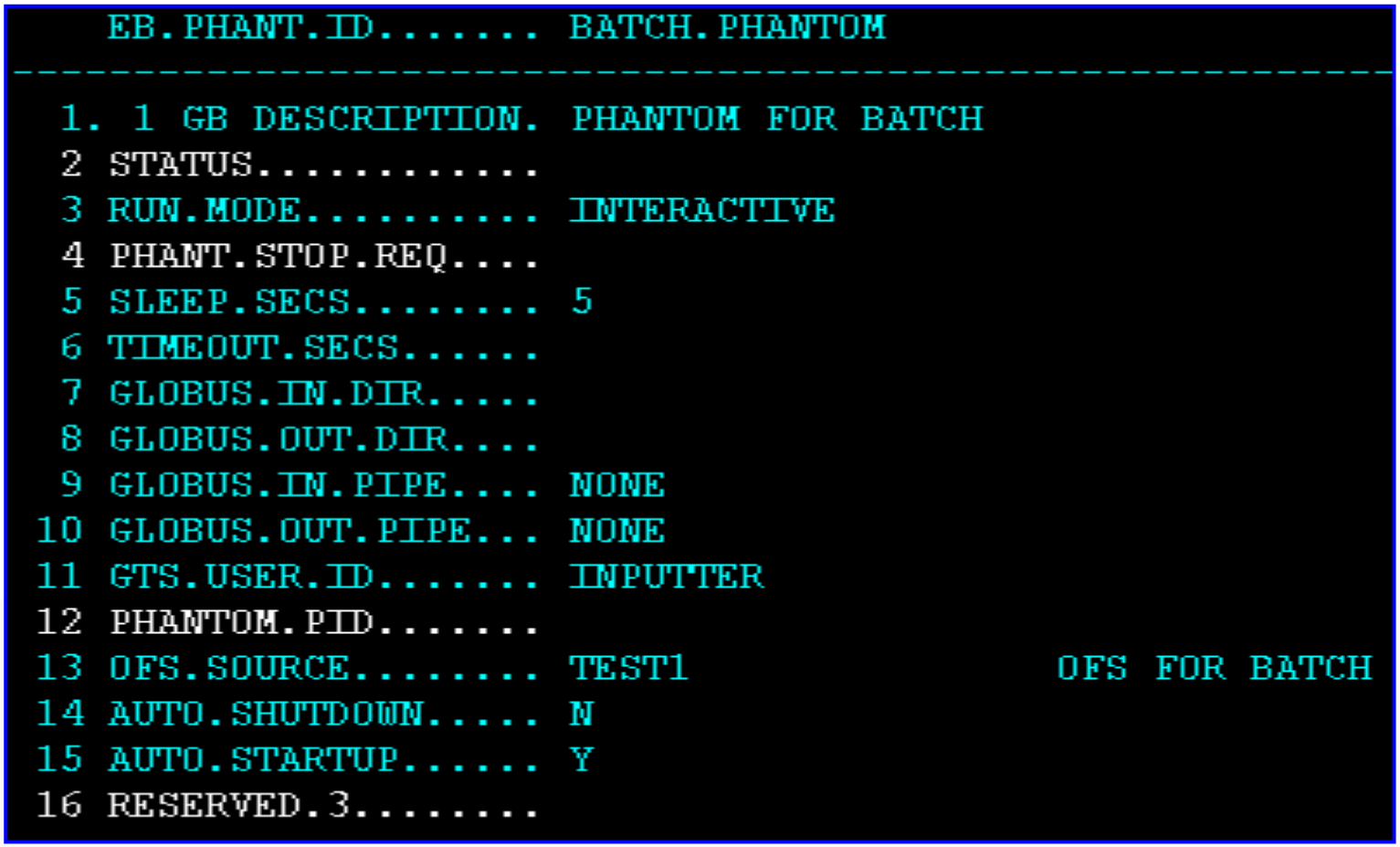
OFS.SOURCE – FOR BATCH MODE

```
    SOURCE.NAME.......  TEST1
-------------------------------------
 1  DESCRIPTION.......  OFS FOR BATCH
 2  SOURCE.TYPE.......  BATCH
 3.  1  LOGIN.ID.......
 4.  1  EB.PHANT.ID....  BATCH.PHANTOM
 5  MAX.CONNECTIONS...
 6  RESTRICT.LINK.....
 7  INITIAL.ROUTINE...
 8  CLOSE.ROUTINE.....
 9  IN.MSG.RTN........
10  OUT.MSG.RTN.......
11  MSG.PRE.RTN.......
12  MSG.POST.RTN......
13  LOG.FILE.DIR......  OFSLOG
14  LOG.DETAIL.LEVEL..  FULL
15  OFFLINE.QUEUE.....
16  MAINT.MSG.DETS....  Y
```

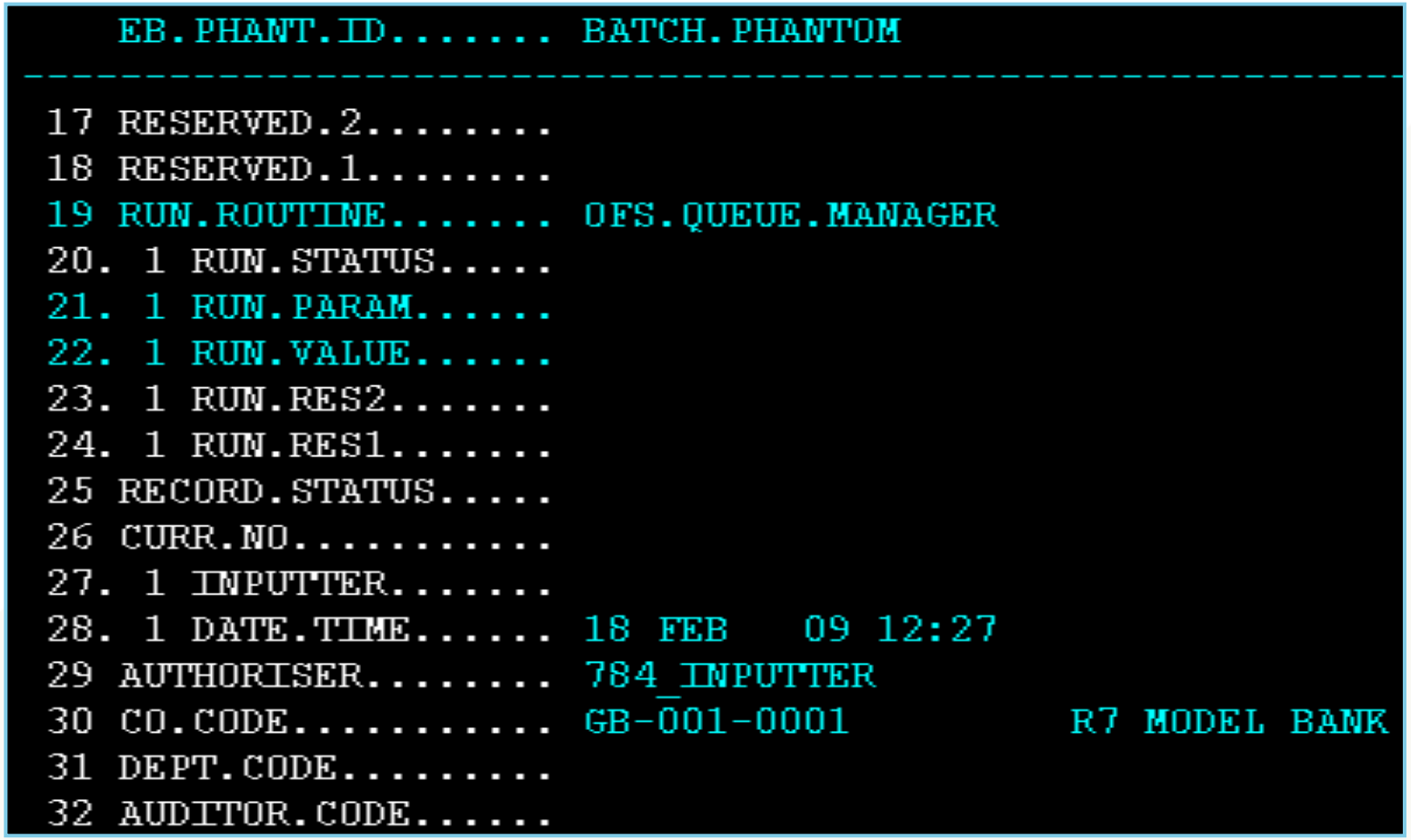
OFS.SOURCE – FOR BATCH MODE



EB.PHANTOM – FOR BATCH MODE



EB.PHANTOM – FOR BATCH MODE



BATCH RUNNING PROCESS

After setting up OFS.SOURCE and EB.PHANTOM for Batch

Input message is passed in INQUEUE dir of OFS.SOURCE

From shell prompt type

jsh>JED OFS.IN MSG1

```
0001 ACCOUNT,OFSACCOUNT,INPUTT/123456,,CUSTOMER::=100781,CATEGORY::=1002,CURRENCY::=USD_
```

EB.PHANTOM

Enter into Globus

Verify EB.PHANTOM Record

EB.PHANTOM V BATCH.PHANTOM

Check the out Queue Dir using

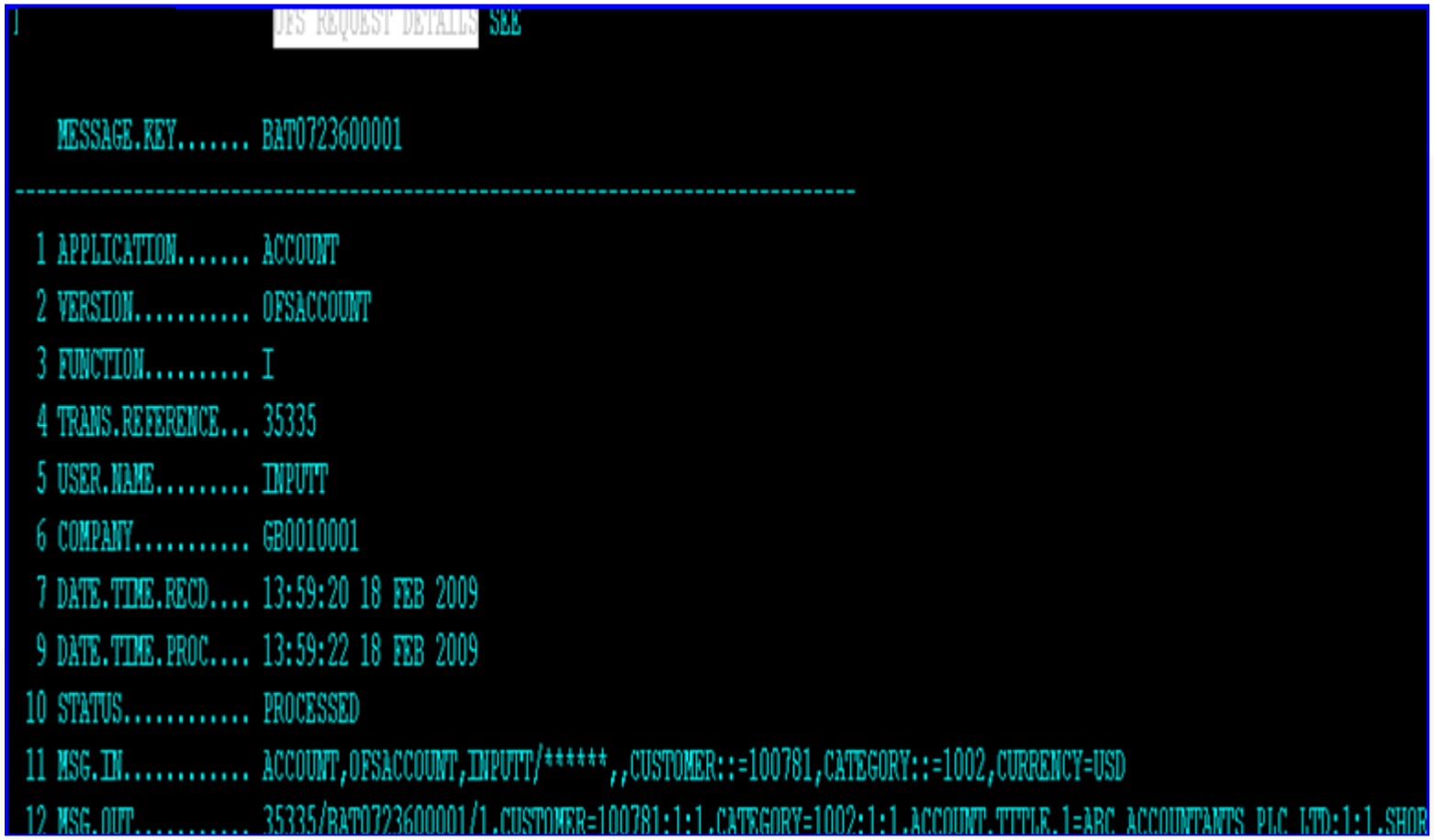
```
35335/BAT0723600001/1,CUSTOMER=100781:1:1,CATEGORY=1002:1:1,ACCOUNT.TITLE.=0902181359:1:1,CO.CODE=GB0010001:1:1,DEPT.CODE=1:1:1
```

OFS.REQUEST.DETAIL

OFS.REQUEST.DETAIL

- Is an application where all the details of the record will be stored
- The message key will be prefixed with the id which we have given in DET.PREFIX = BAT

OFS.REQUEST.DETAIL



Workshop - II

Set up OFS for Batch

Create EB.PHANTOM

Pass the OFS message in the INFILE for updating a sector version

Solution – Step 1: Setup BATCH

	SOURCE.NAME.....	BATCH.SOURCE
1	DESCRIPTION.....	OFS FOR BATCH
2	SOURCE.TYPE.....	BATCH
3.	1 LOGIN.ID.....	
4.	1 EB.PHANT.ID....	BATCH.PHANTOM
5	MAX.CONNECTIONS...	
6	RESTRICT.LINK.....	
7	INITIAL.ROUTINE...	
8	CLOSE.ROUTINE.....	
9	IN.MSG.RTN.....	
10	OUT.MSG.RTN.....	
11	MSG.PRE.RTN.....	
12	MSG.POST.RTN.....	
13	LOG.FILE.DIR.....	OFSLOG
14	LOG.DETAIL.LEVEL..	FULL
15	OFFLINE.QUEUE.....	
16	MAINT.MSG.DETS....	Y

	SOURCE.NAME.....	BATCH.SOURCE
17	DET.PREFIX.....	BAT
18	IN.QUEUE.DIR.....	OFS.IN
19	IN.QUEUE.NAME.....	
20	OUT.QUEUE.DIR.....	OFS.OUT
21	OUT.QUEUE.NAME....	
22	QUEUE.INIT.RTN....	
23	QUEUE.CLOSE.RTN...	
24	SYNTAX.TYPE.....	OFS
25.	1 LOCAL.REF.....	
26	GENERIC.USER.....	AUTH1
27	IN.DIR.RTN.....	
28	VERSION.....	
29	IB.USER.CHECK.....	
30	EOD.VALIDATE.....	
31	FIELD.VAL.....	
32.	1 ATTRIBUTES.....	

Solution – Step 2 : EB.PHANTOM

	EB.PHANT.ID.....	BATCH.PHANTOM
1.	1 GB DESCRIPTION.	PHANTOM FOR OFS BA
2	STATUS.....	CLOSED
3	RUN.MODE.....	INTERACTIVE
4	PHANT.STOP.REQ....	
5	SLEEP.SECS.....	5
6	TIMEOUT.SECS.....	
7	GLOBUS.IN.DIR.....	
8	GLOBUS.OUT.DIR....	
9	GLOBUS.IN.PIPE....	NONE
10	GLOBUS.OUT.PIPE...	NONE
11	GTS.USER.ID.....	AUTH1
12	PHANTOM.PID.....	
13	OFS.SOURCE.....	BATCH.SOURCE
14	AUTO.SHUTDOWN.....	N
15	AUTO.STARTUP.....	Y
16	RESERVED.3.....	

	EB.PHANT.ID.....	BATCH.PHANTOM
17	RESERVED.2.....	
18	RESERVED.1.....	
19	RUN.ROUTINE.....	OFS.QUEUE.MANAGER
20.	1 RUN.STATUS.....	
21.	1 RUN.PARAM.....	
22.	1 RUN.VALUE.....	
23.	1 RUN.RES2.....	
24.	1 RUN.RES1.....	
25	RECORD.STATUS.....	
26	CURR.NO.....	
27.	1 INPUTTER.....	
28.	1 DATE.TIME.....	19 MAR 06 17:33
29	AUTHORISER.....	11_AUTH2
30	CO.CODE.....	US-001-0001
31	DEPT.CODE.....	
32	AUDITOR.CODE.....	

Solution – Step 3: Create a Record

After setting OFS.SOURCE and EB.PHANTOM for Batch then pass the OFS message to be processed through the In Queue Directory specified in your OFS.SOURCE record.

Here In Queue Dir is OFS.IN

From the jbase prompt : JED OFS.IN <record>

Save the record

```
NEW *File OFS.IN , Record 'MSG1'                          Insert      17:35:23
Command->
0001 SECTOR,TEST/I/PROCESS,,1500/,DESCRIPTION::=Zenith,SHORT.NAME::=Lagos
------------------------------ End Of Record ------------------------------
```

Solution – Step 4: Verify EB.PHANTOM

Enter INTO GLOBUS/T24

From the Globus Application using ‘COMMITT’ the created EB.PHANTOM record (BATCH.PHANTOM) with Function **“V”**

Example : EB.PHANTOM V BATCH.PHANTOM

Check the Out Queue Dir using JED OFS.OUT MSG1

```
File OFS.OUT , Record ´MSG1´                                  Insert      17:50:44
Command->
0001 1500/BAT0033500001/1,DESCRIPTION:1:1=Zenith,SHORT.NAME:1:1=Lagos,CURR.NO:1
```

Indicates success

ENQUIRY – USING OFS

ENQUIRY CAN BE DONE USING OFS MESSAGE STRUCTURE
OPERATIONS should always be ENQUIRY.SELECT

TRANSACTION – ID SHOULD BE NAME OF THE ENQUIRY
MESSAGE DATA CAN BE SPECIFIED BY SELECTION FIELD

ENQUIRY - OFS

Enquiry can either be passed through online or Batch

Steps to be followed are the same as that of Transaction

Message structure of Enquiry

ENQUIRY.SELECT,,USERINFORMATION,ENQUIRYNAME,MESSAGEDATA

ENQUIRY - OFS

ENQUIRY.SELECT

- The ENQUIRY.SELECT is actually a TEMENOS T24 application that is used to run queries and return the data
- The first portion of an enquiry type request must always be ENQUIRY.SELECT

USER INFORMATION

- The user information portion of the message structure is same as that in the transaction type request

ENQUIRY - OFS

ENQUIRY.NAME

- The enquiry name portion of the message structure must contain the name of the TEMENOS T24 Enquiry that will be run
- The Enquiry name supplied here must be a valid TEMENOS T24 enquiry

ENQUIRY.DATA

- The message data portion of the enquiry message structure contains the selection criteria passed to the enquiry

ENQUIRY – OFS (SYNTAX)

Type the following command in shell prompt

```
jsh r      ~ -->tss TEST
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>308</t24pid><t24ofssource>TEST</t24ofssource><clientIP/></tSS>
```

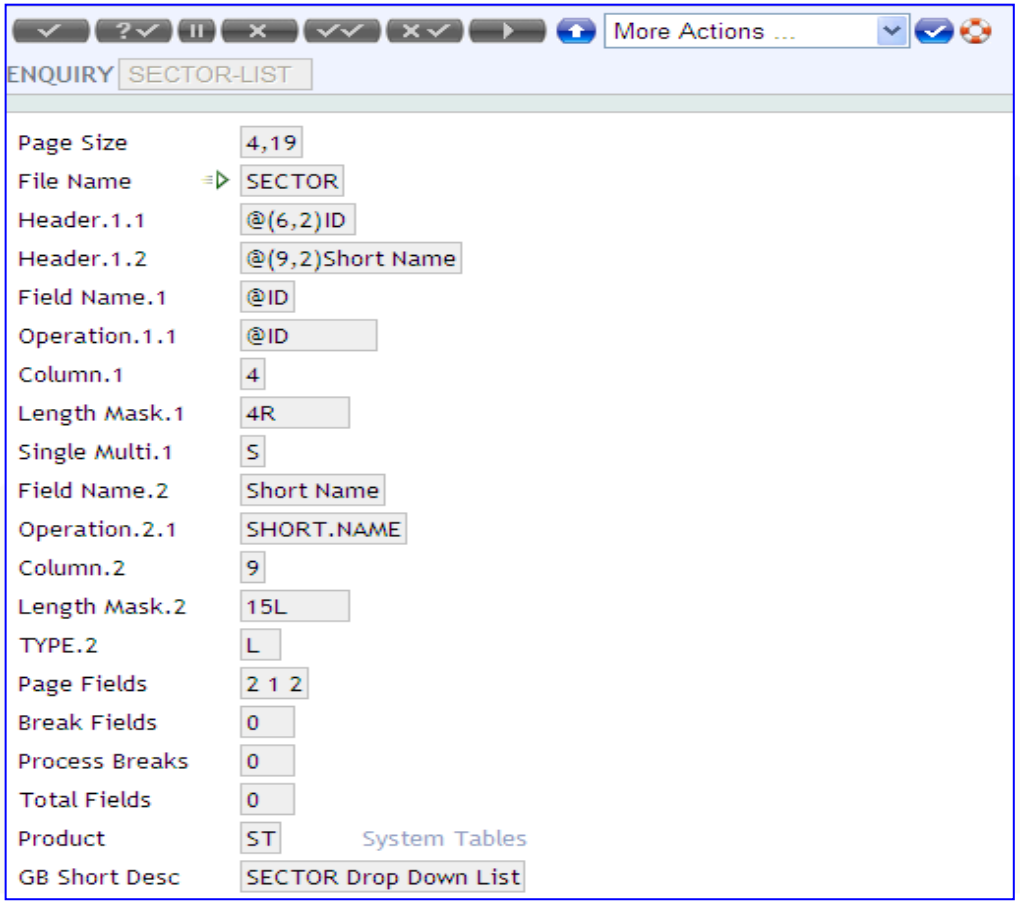
```
jsh        ~ -->tSS TEST
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>308</t24pid><t24ofssource>TEST</t24ofssource><clientIP/></tSS>
ENQUIRY.SELECT,,INPUTT/123456,CUS.1
,@ID::@ID/RESIDENCE::RESIDENCE/NATIONALITY::NATIONALITY/SECTOR::SECTOR/TARGET::TARGET/INDUSTRY::INDUSTRY,"    100112"   "NL"    "NL
"1501"  "   7"  "3200","    100113"    "GB"    "GB"    "1501"  "   7"  "3200","    100114"    "US"    "US"    "3501"  "   6"  "32
0","    100115" "US"    "US"    "3501"  "   6"  "3130","    100116"    "US"    "US"    "3501"  "   6"  "3200","    100117"    "CA
"CA"    "3501"  "   7"  "3200","    100130"    "US"    "US"    "2001"  "   7"  "2710","    100132"    "US"    "US"    "1001"  "
1"      "1800","    100139"    "US"    "US"    "3001"  "  30"  "3100","    100149"    "IN"    "IN"    "3501"  " 999"  "3900","
100200" "US"    "US"    "1001"  "   1"  "1800","    100205"    "BB"    "BB"    "1000"  " 999"  "1000","    100211"    "AU"    "AU
"1000"  "   1"  "1000","    100224"    "GB"    "GB"    "1001"  "   1"  "1000","    100225"    "GB"    "GB"    "1001"  "   1"  "10
0","    100226" "GB"    "GB"    "1001"  "   2"  "1000","    100227"    "GB"    "GB"    "1001"  "   2"  "1000","    100228"    "GB
"GB"    "1001"  "   2"  "1000","    100229"    "GB"    "GB"    "1001"  "   2"  "1000","    100231"    "US"    "US"    "3001"  " 9
9"      "3100","    100232"    "US"    "US"    "3001"  " 999"  "3100","    100236"    "US"    "US"    "1501"  "   7"  "3200","
100240" "US"    "US"    "3001"  " 999"  "3100","    100241"    "US"    "US"    "9000"  " 999"  "3100","    100242"    "US"    "US
"3501"  " 999"  "8200","    100243"    "US"    "US"    "3505"  " 999"  "8190","    100244"    "DE"    "DE"    "3503"  " 999"  "31
```

Workshop - III

Create a enquiry type request to find the list of SECTOR found in T24

Solution – Step 1 : Enquiry for sector

Enquiry for sector contains the following fields



Solution – Step 2 : OFS MESSAGE

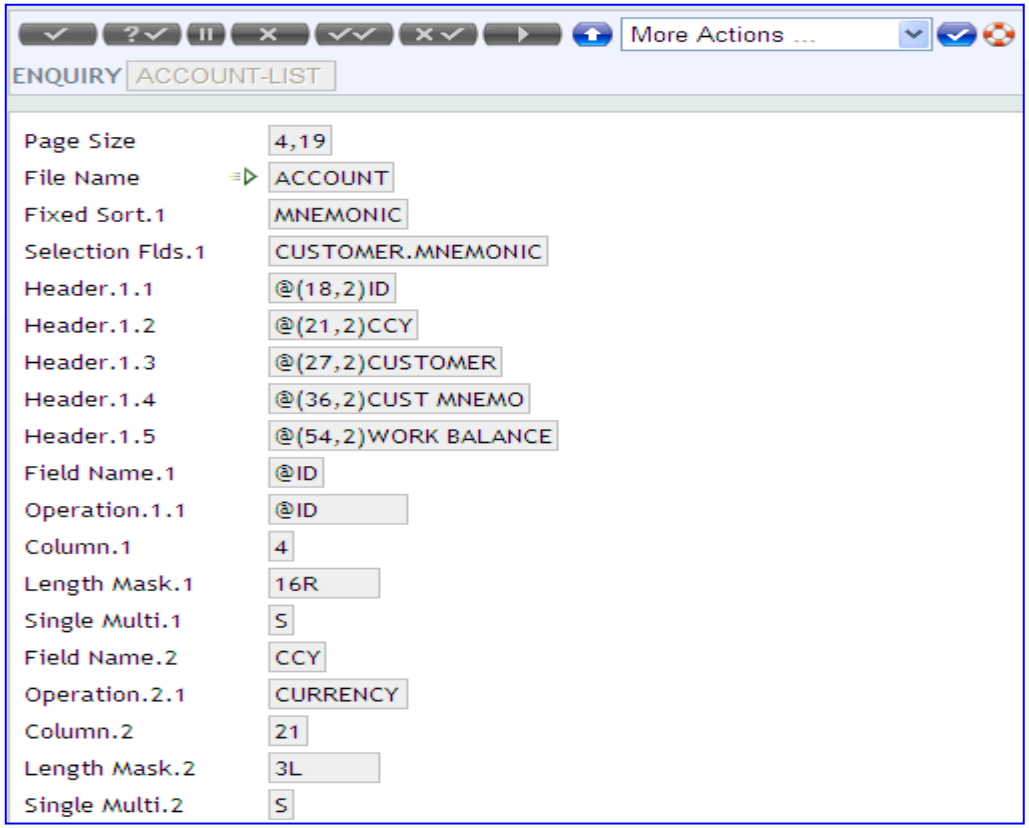
```
jsh :          ->tss TEST
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>4284</t24pid><t24ofssource>TEST1</t24ofssource><clientIP/></tSS>
ENQUIRY.SELECT,,STEN11/123456,SECTOR-LIST
,@ID::@ID/Short Name::Short Name," 111" "SENIOR CITIZEN "," 222"       "STUDENTS      "," 444"       "HOST          "," 555"   "
TOBA          "," 888" "Entrepreneur   ","1000"       "Individuals   ","1001"       "Individual    ","1002"       "Staff
   ","1005"    "JOINT         ","1201"       "WTR           ","1202"       "QTRGYRGTR     ","1499"       "Individuals   ","1
500"    "Brokers       ","1501"       "Broker        ","1599"       "Brokers       ","1600"       "Clrg Agents   ","1601"   "
Clrg Agents    ","1699" "Clrg Agents    ","1900"       "Oth Individuals","1999"       "Oth Individuals","2000"       "Corporate
   ","2001"    "Corporate     ","2002"       "Unincorp Busnss","2999"       "Corporate     ","3000"       "Banks         ","3
001"    "Banks         ","3002"       "CentralBanks  ","3003"       "Post Office   ","3005"       "Bank Branches ","3499"   "
Banks         ","3500" "FinInstitutions","3501"       "Financial Corps","3502"       "Ins Companies ","3503"       "Fin Advisor
s   ","3504"    "Mutual Funds   ","3505"       "Invstmnt Exchgs","3900"       "OthFinInstits ","3999"       "FinInstitutions","4
000"    "Public Sector ","4001"       "CentralGovt   ","4002"       "LocalGovt     ","4599"       "Public Sector ","4600"   "
Non-Prof Orgs  ","4601" "EduInstitutions","4999"       "Non-Prof Orgs ","5000"       "IntlOrgs      ","5001"       "UnitedNatio
ns  ","5002"    "UNESCO        ","5999"       "OthIntlOrgs   ","9000"       "Others        "
```

Workshop IV

Create a enquiry type request to find the Customer whose WORKING.BALANCE > 10000

Solution – Step 1 : Enquiry for Account

The enquiry Account-list contains the following fields for display.



Solution – Step 2 : OFS MESSAGE

```
jsh        ~ -->tss TEST
<tss version="1.1"><t24version>R07.002</t24version><t24pid>4284</t24pid><t24ofssource>TEST1</t24ofssource><clientIP/></tSS>
ENQUIRY.SELECT,,STEMI1/123456,ACCOUNT-LIST,WORKING.BALANCE>1000_
```

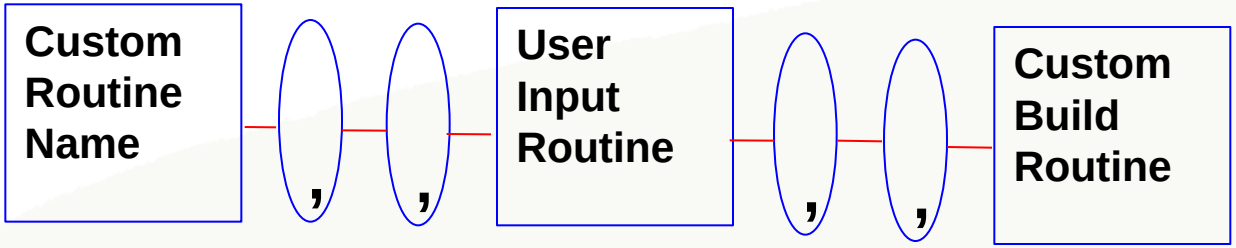
```
),000.00","         14281"   "EUR"   "    100313"    "    PEPSI"    "                ","         14297"      "GBP"   "
100313"       "    PEPSI"   "        -52,808.68","         14303"      "JPY"   "    100313"    "    PEPSI"    "
      ","         14265"   "USD"   "    100313"    "    PEPSI"    "      191,465.55","         28339"      "USD"   "
100313"       "    PEPSI"   "         -3,917.33","         29197"      "USD"   "    100313"    "    PEPSI"    "
-844.97","   USD119100001"   "USD"   "         "    "         "    "                ","   USD129050001"      "USD"   "
     "       "          "   "                ","         13021"      "CHF"   "    100271"    "   PEUGEOT"   "      2,39
9,000.00","         13037"   "EUR"   "    100271"    "  PEUGEOT"    "          -56.35","         13048"      "GBP"   "
100271"       "   PEUGEOT"   "                ","         13056"      "JPY"   "    100271"    "   PEUGEOT"   "
      ","         13013"   "USD"   "    100271"    "  PEUGEOT"    "   -1,056,367.74","         35599"      "USD"   "
100286"       "   PMORRIS"   "                ","         31321"      "USD"   "    100200"    "  PHILIPPS"   "         5
),804.69","         31372"   "GBP"   "    100200"    "PHILIPPS"    "      201,194.27","         13927"      "CHF"   "
100286"       "   PMORRIS"   "                ","         13935"      "EUR"   "    100286"    "   PMORRIS"   "
      ","         13943"   "GBP"   "    100286"    "  PMORRIS"    "                ","         13951"      "JPY"   "
100286"       "   PMORRIS"   "                ","         13919"      "USD"   "    100286"    "   PMORRIS"   "        79
9,148.91","         31337"   "USD"   "    100200"    "PHILIPPS"    "       -13,616.57","         12823"      "CHF"   "
100272"       "  PTNAULTF"   "                ","         12831"      "EUR"   "    100272"    "  PTNAULTF"   "
```

ROUTINE REQUEST – OFS

OFS messages can be passed to the routine

Data received from OFS can be formatted according to the user requirement and the value can be returned

Message format for calling ROUTINE



Message format for calling ROUTINE

CUSTOM ROUTINE NAME

- Name of the routine to be called by OFS
- Should be a valid subroutine
- Routine name should have a PGM.FILE entry

USER INFORMATION

- Valid User name and password

CUSTOM DATA

- String that is to be passed to the routine

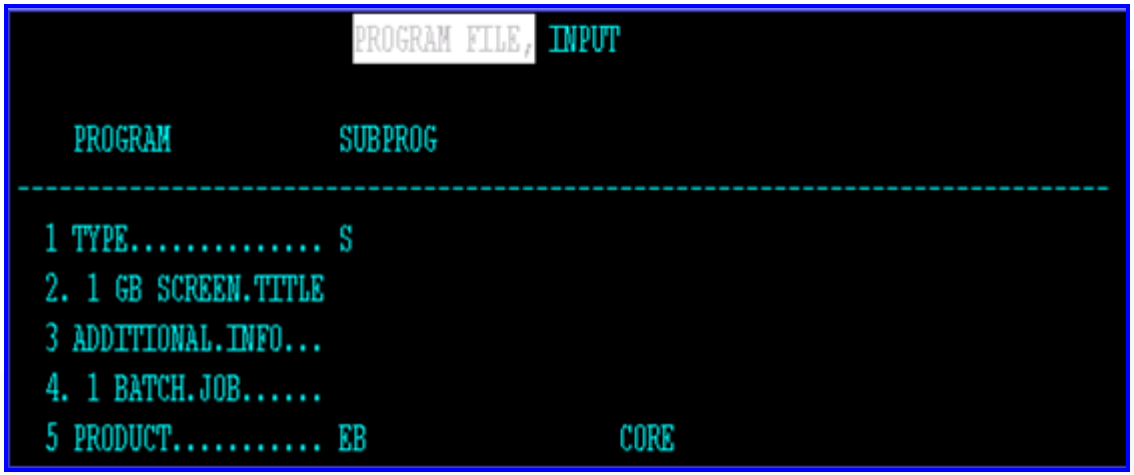
Program

The program will pass the OFS format message to T24. If the message is successfully passed then it will return a message as VALID ACCOUNT

SAMPLE PROGRAM FOR OFS ROUTINE CALL

```
001 SUBROUTINE SUBPROG(ARGIN)
002           $INSERT I_COMMON
003           $INSERT I_EQUATE
004           $INSERT I_F.ACCOUNT
005           Y.VAR1=ARGIN
006           Y.VAR2 = ''
007           Y.CUS.ID = ''
008           Y.BAL = ''
009           R.ACCT = ''
010           FN.ACCT='F.ACCOUNT'
011           F.ACCT=''
012           CALL OPF(FN.ACCT,F.ACCT)
013           CALL F.READ(FN.ACCT,Y.VAR1,R.ACCT,F.ACCT,F.ERR)
014           IF R.ACCT THEN
015 ARGIN='ACCOUNT,/I/PROCESS,INPUTT/123456/,':ARGIN:',SHORT.TITLE::=TESTING_
016 ARGIN = 'VALID ACCOUNT'
017           END
```

PGM.FILE


PROGRAM FILE, INPUT
PROGRAM
SUBPROG
1 TYPE..... S
2. 1 GB SCREEN.TITLE
3 ADDITIONAL.INFO...
4. 1 BATCH.JOB.....
5 PRODUCT..... EB
CORE

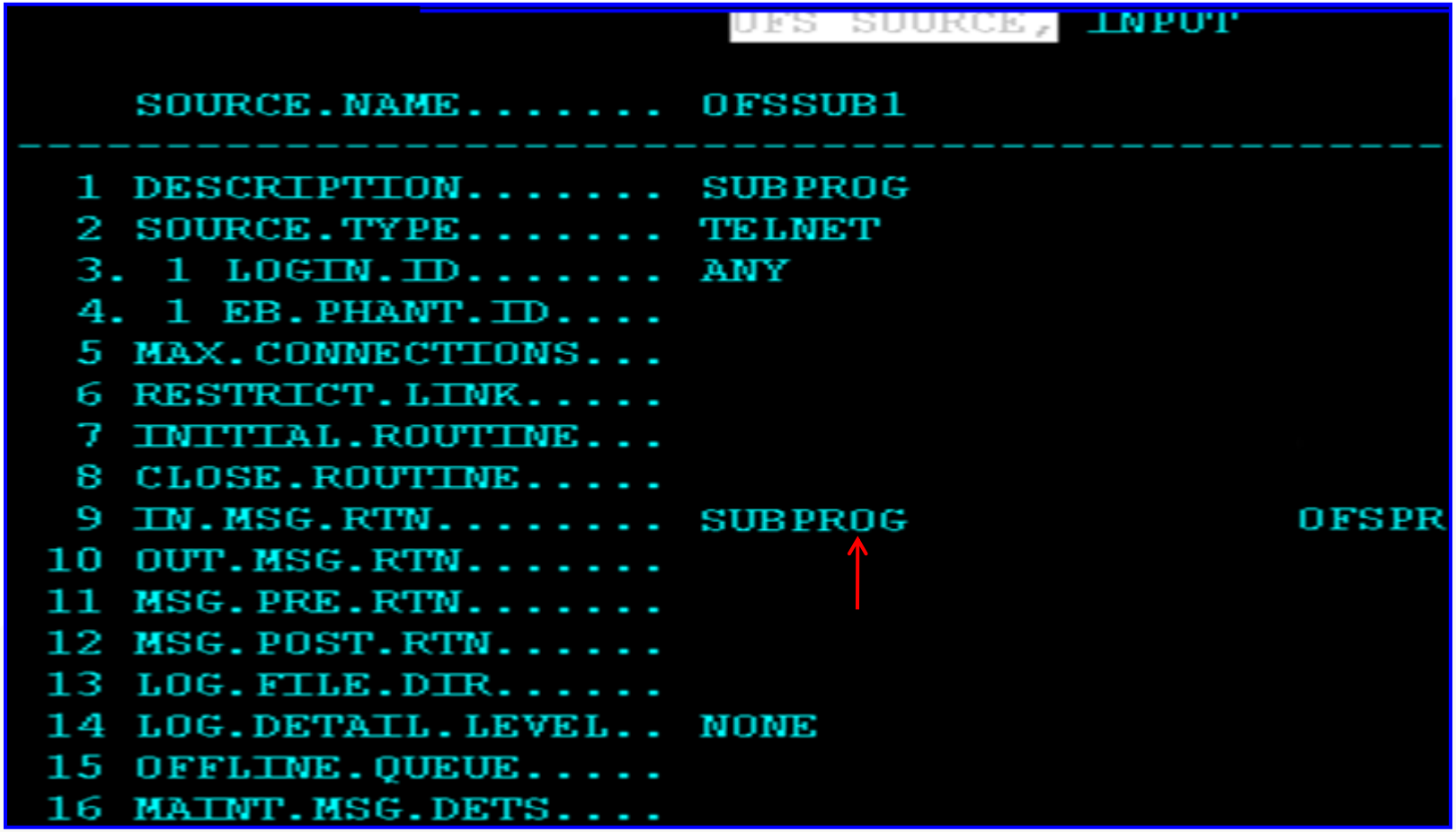
Steps for execution

Compile the subroutine SUBPROG

Create a OFS.SOURCE

Specify the name of the compiled subroutine in the IN.MSG.RTN

OFS.SOURCE FOR SUBROUTINE CALL



SAMPLE SCREEN OF OFS MESSAGE

After setting up the OFS

Pass the a/c no through OFS message

```
jsh         ~ -->tss TEST
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>3960</t24pid><t24ofssource>OFSSUB1</t24ofssource><clientIP/></tSS>
10014_
```

```
10014//1,CUSTOMER:1:1=100778,CATEGORY:1:1=1001,ACCOUNT.TITLE.1:1:1=TREASURY CUSTOMER 1,SHORT.TITLE:1:1=TESTING OFS,MNEMONIC:1:1=TRAI
01,POSITION.TYPE:1:1=TR,CURRENCY:1:1=AUD,CURRENCY.MARKET:1:1=1,ACCOUNT.OFFICER:1:1=35,OTHER.OFFICER:1:1=5,RECONCILE.ACCT:1:1=Y,CONDI
TION.GROUP:1:1=1,INACTIV.MARKER:1:1=Y,CAP.DATE.CHARGE:1:1=20070630,CAP.DATE.CHARGE:2:1=20070430,PASSBOOK:1:1=NO,OPENING.DATE:1:1=200
60417,OPEN.CATEGORY:1:1=1001,CHARGE.CCY:1:1=AUD,INTEREST.CCY:1:1=AUD,ALT.ACCT.TYPE:1:1=LEGACY,ALLOW.NETTING:1:1=NO,ACC.DEB.LIMIT:1:1
=20070709,RECORD.STATUS:1:1=INAU,CURR.NO:1:1=2,INPUTTER:1:1=780_INPUTTER__OFS_OFSSUB1,DATE.TIME:1:1=0902191018,CO.CODE:1:1=GB001000
1,DEPT.CODE:1:1=1
```

Workshop - V

Write a Subroutine to count the number of characters received in the input string and return back a string “The length of Request message is:” followed by the actual count characters in the request message. If the request message is empty then a string “The length of Request message is: EMPTY” must be returned..

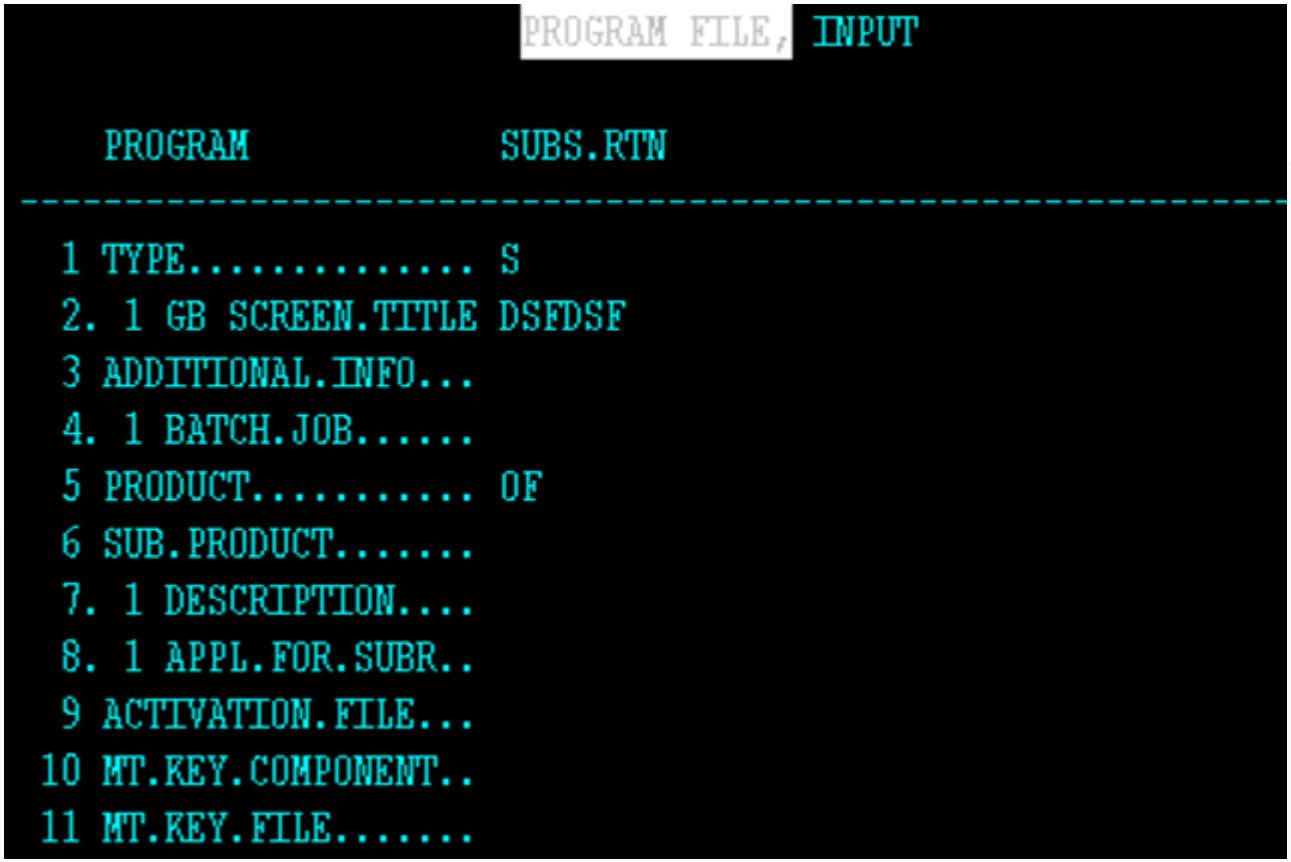
Workshop - V

Now, use the OFS Online mode you created in the previous workshop to upload your transaction type messages and see the results. If you were to receive any errors, check the request message to see if everything is correct about the syntax. Also check if the setup is correct

Solution – Step 1 : Write the Routine

```
*File THEME.BP , Record 'SUBS.RTN'                          Insert      17:14:53
Command->
0001 *-----------------------------------------------------------------------
0002 * <Rating>0</Rating>
0003 *-----------------------------------------------------------------------
0004 SUBROUTINE SUBS.RTN(Y.REQUEST,Y.RESPONSE)
0005 *IF THE REQUEST MSG DOES NOT CONTAIN ANY
0006 *DATA THEN DID NOT RECIEVE ANYTHING WOULD
0007 *BE DISPLAYED ELSE LENGTH OF THE DATA STORED
0008 * IN Y.REQUEST IS DISPLAYED
0009 IF Y.REQUEST NE "" THEN
0010 Y.RESPONSE = "THE RETURN MESSAGE IS": "  ":LEN(Y.REQUEST)
0011 END ELSE
0012 Y.RESPONSE = "DID NOT RECIEVE ANYTHING"
0013 END
0014 RETURN
------------------------------ End Of Record ------------------------------
```

Solution – Step 2 : PGM.FILE



Solution – Step 3 : Output

```
jsh        . -->TSS TELNET
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>1096</t24pid><t24ofssource>TELNET</t24ofssource><clientIP/></tSS>
SUBS.RTN,,INPUTT/123456,,THESYS
THE RETURN MESSAGE IS  6
```

```
jsh        . -->TSS TELNET
<tSS version="1.1"><t24version>R07.002</t24version><t24pid>1096</t24pid><t24ofssource>TELNET</t24ofssource><clientIP/></tSS>
SUBS.RTN,,INPUTT/123456,,
DID NOT RECIEVE ANYTHING
-
```

I_GTS.COMMON – OFS Common variables

The I_GTS.COMMON contains the common variables of the OFS module

Inserting this in the programs makes them available for use during the OFS processing

A subroutine attached in MSG.PRE.RTN field of OFS.SOURCE can make use of the common variables of the OFS module by inserting the I_GTS.COMMON file

OFS Common variables

GTSACTIVE	Flag to indicate if OFS is used for the message being processed
OFS$SOURCE.ID	ID of the OFS.SOURCE record that is currently being used
OFS$SOURCE.REC	Dynamic array containing the OFS.SOURCE record that is currently being used
OFS$GLOBMAN.ACTIVE	Flag to indicate if OFS Globus mode (OFS.GLOBUS.MANAGER) is used for the message being processed

DATA CAPTURE- Introduction

Data capture is an Account Based application which is used to pass entries to any type of account within T24

Data capture record which generates one accounting entry,

- E.g.: To debit one customer a/c and credit P/L a/c (it may be commission on Demand Draft)

Feature of Data Capture

A Data Capture record generates one accounting entry – either a debit or a credit only

While in other applications a single transaction may create multiple accounting entries

DC items are grouped in batches and each batch must have balancing debits and credits in local currency

Batch - Introduction

Combination of debit and credit entries are called ‘Batch’

Batch must have at least two entries i.e. one debit and one credit Batch – Groups into which items are separated for control purposes

- E.g.: Single debit to a Nostro a/c, balanced by many credits to varying client a/c.

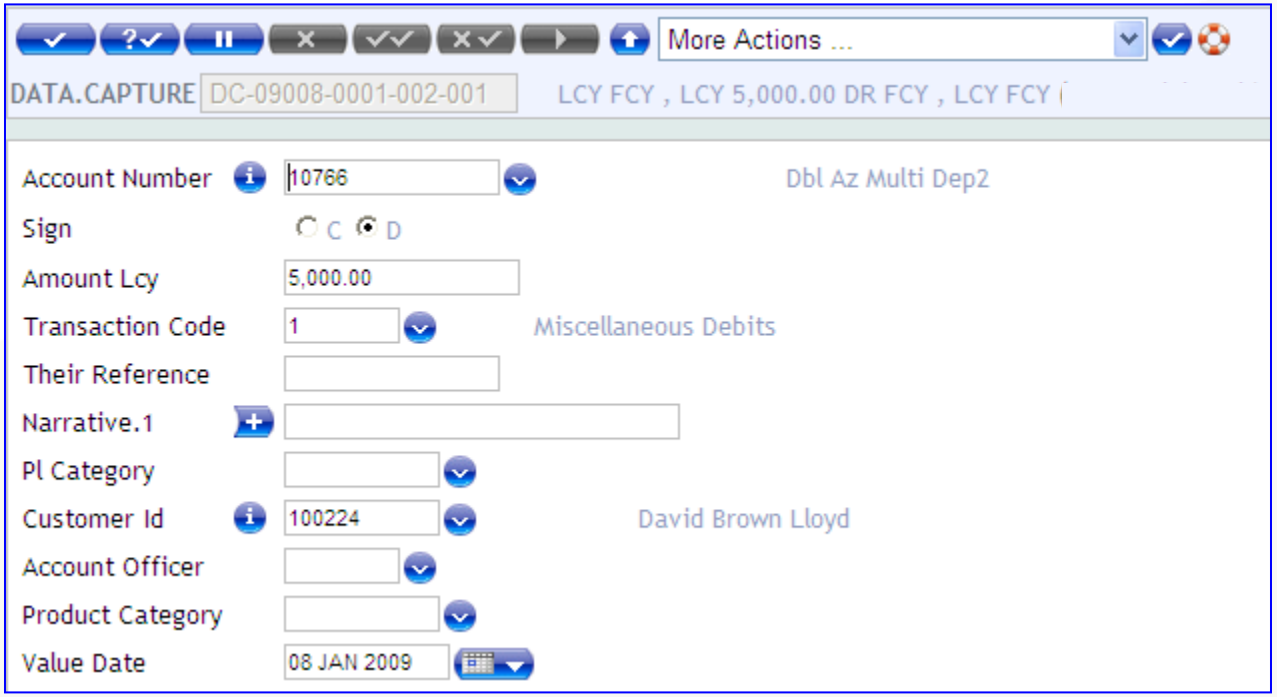
Batch - Introduction

Data capture applies most of its accounting entries online

Only where a batch has not been fully authorized, or does not balance, will any entries to be generated during end of the day

OFS – DATA CAPTURE STEPS INVOLVED

Open “DC” application and pass the Debit leg for a particular a/c



ID OF DATA CAPTURE

Id is a 10 digit reference number used in DATA.CAPTURE record which generates one accounting entry,

The 10 digit reference nuumber is comprised of Department code; Batch number; and Item number

The Department code is derived from the DEPT.ACCT.OFFICER system table, the reference is as follows:

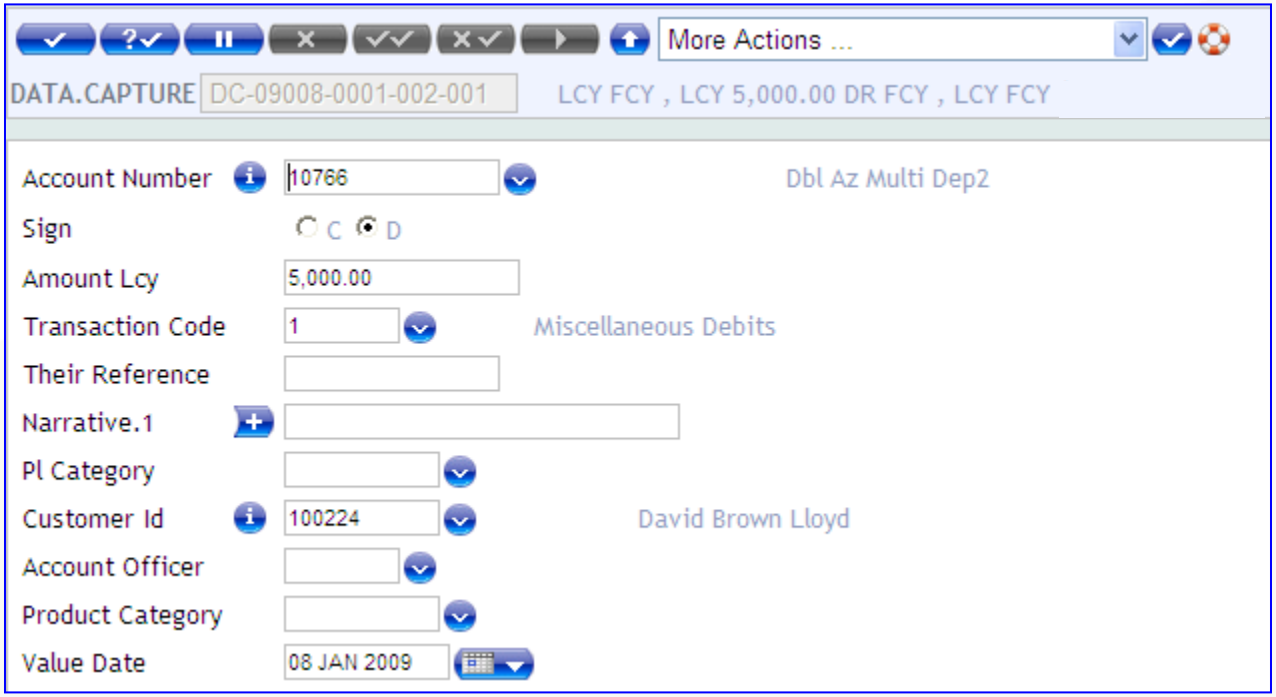
Department code: Value 0001-9999

Batch Number : Value 001-999

Item Number : Value 001-998

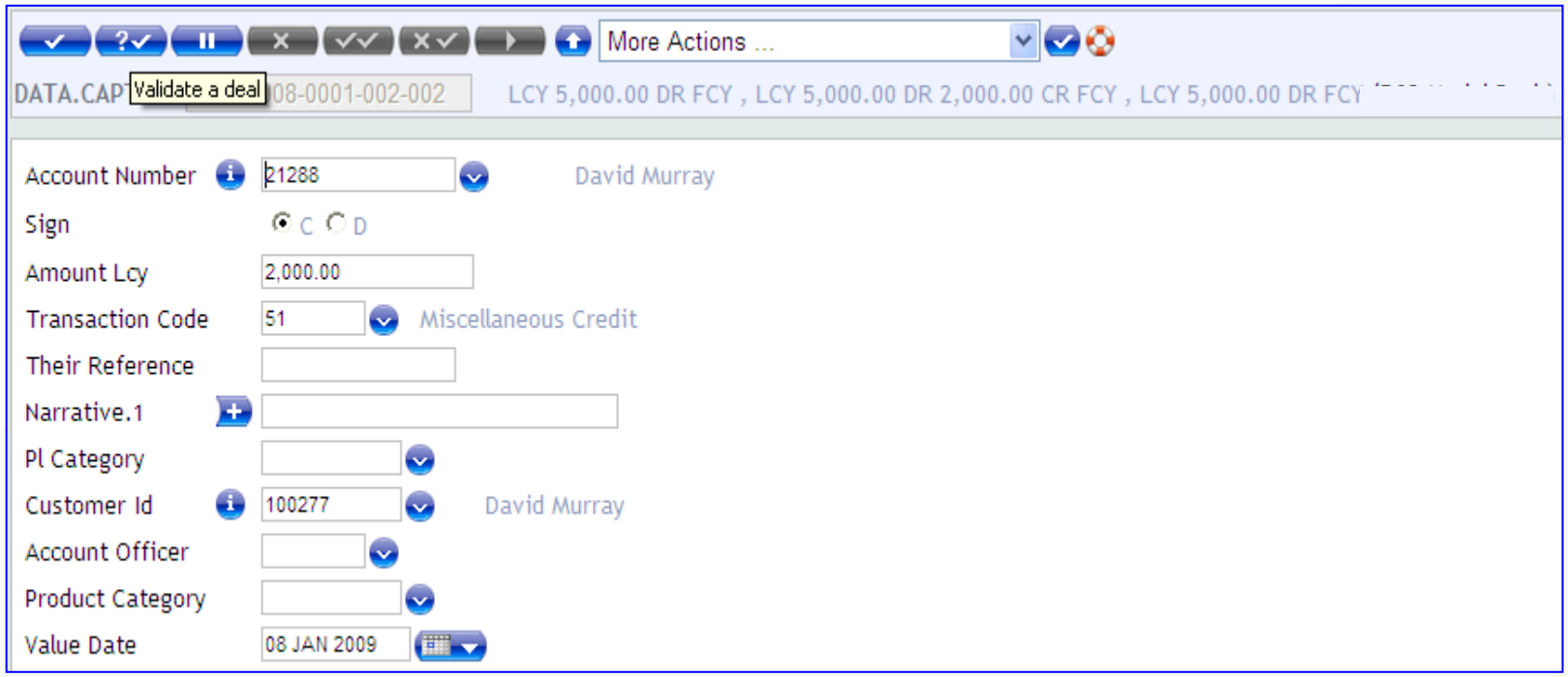
OFS – DATA CAPTURE STEPS INVOLVED

Open “DC” application and pass the Debit leg for a particular a/c

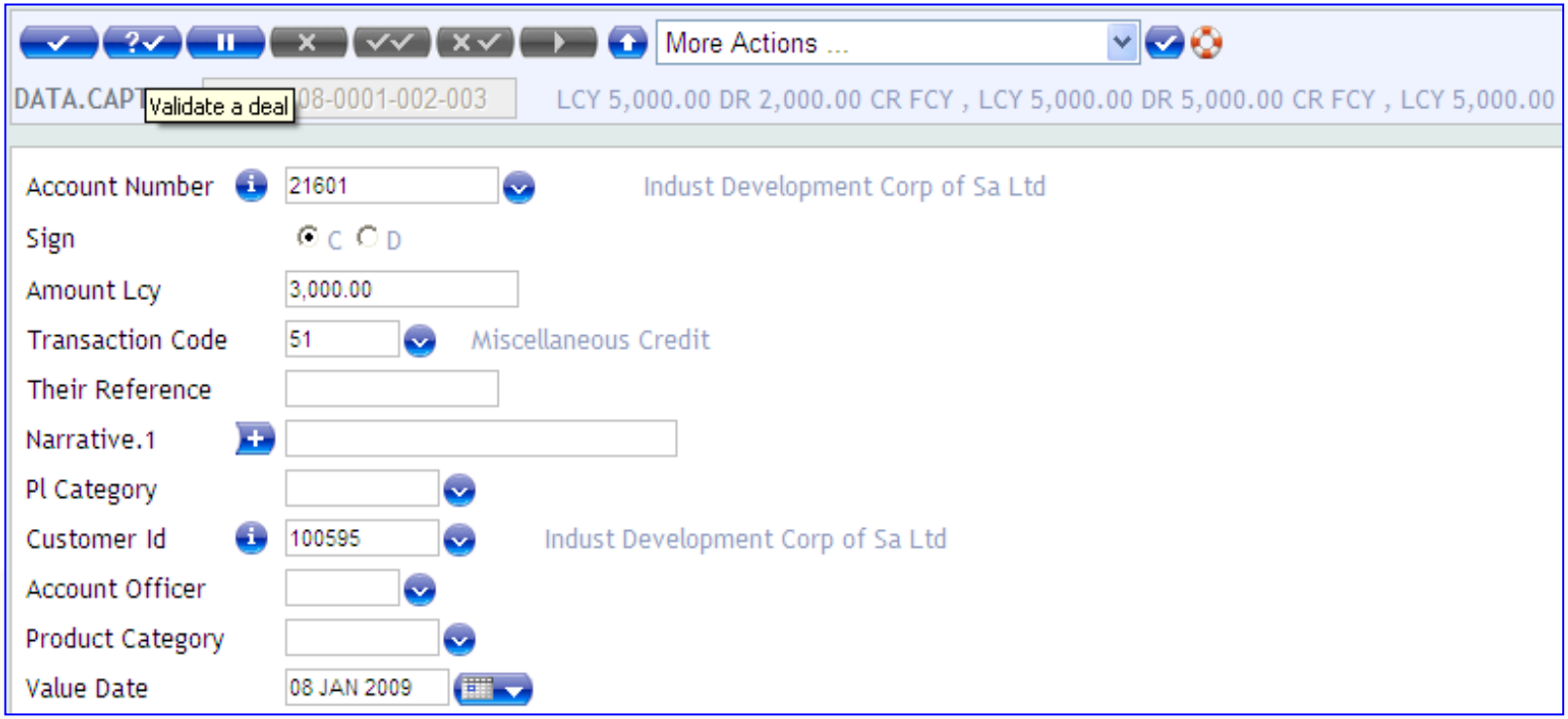


CREDIT LEG FOR 1ST A/C

After passing the debit leg, pass the credit leg for multiple accounts



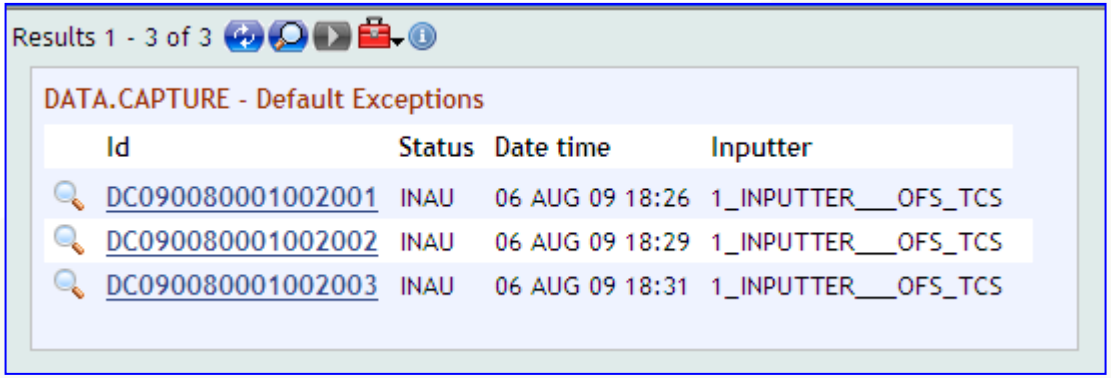
CREDIT LEG FOR SECOND A/C



DATA CAPTURE

After passing both the debit and credit legs

The Batch is now available in INAU



DATA CAPTURE

Now OFS Message is used to authorize the Batch

To pass an OFS message to authorize or Delete a DC Batch, the Transaction ID part of the message must be ‘DC Department Code : DC Batch Code ALL’

Assuming 2 DC legs DC072360001017001 & DC072360001017002, DC072360001017003 need to be authorized, then the Transaction ID part of the OFS message must be ‘DC072360001017 ALL’ and the Function would be A or D as appropriate

SAMPLE SCREEN - DC

From the shell prompt pass the following OFS format

```
jsh       ~ -->tSS TELNET
<tSS version="1.1"><t24version>R09.000</t24version><t24pid>331976</t24pid><t24ofssource>TELNET</t24ofssource><clientIP/></tSS>
DC,/A,INPUTT/123456,DC090080001002 ALL
DC090080001002 ALL//-1/NO,@ID:1:1=EB.RTN.SAME.NAME.AUTHORISER/INPUTTER,@ID:1:1=EB.RTN.SAME.NAME.AUTHORISER/INPUTTER,@ID:1:1=EB.RTN.S
AME.NAME.AUTHORISER/INPUTTER
```

Workshop - VI

Create a file with following details – Debit Account Number, Debit Currency, Debit Amount, Credit Currency and Credit Account Number, separated by a comma

Write a Subroutine to read the details from the file created in the above step and create FUNDS.TRANSFER messages in OFS format

Like in the previous exercise the version, user name and password can be put in a file and read

Solution – Step 1: Creating a file

```
File RENU.BP , Record 'MSG.IN'                          Insert     12:13:46
Command->
0001 14637,EUR,14613.00,14613,USD
------------------------------ End Of Record -------------------------------
```

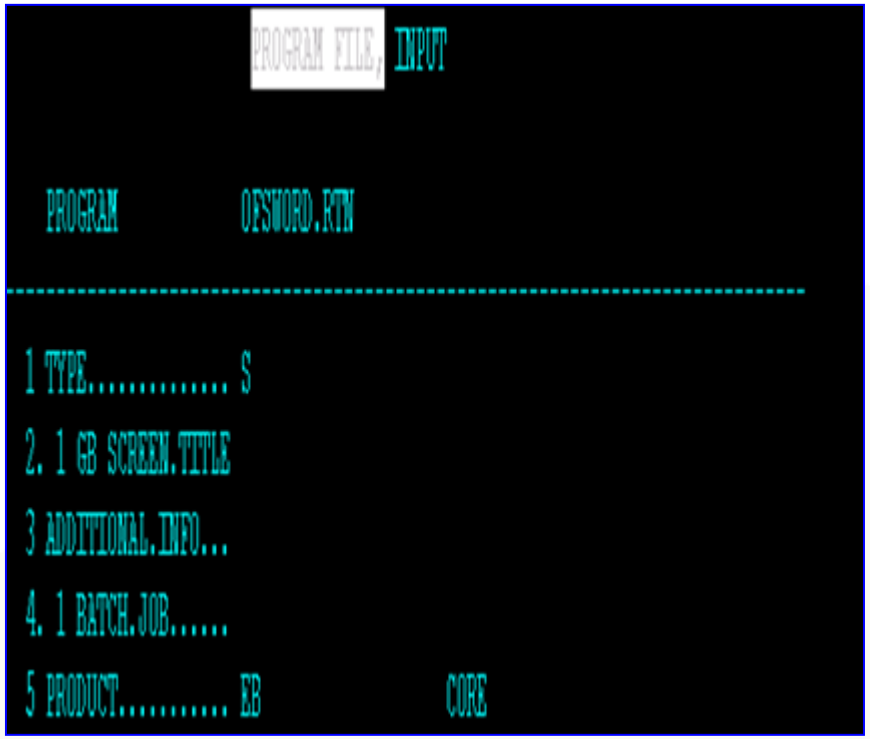
Solution – Step 2: Routine

```
*File RENU.BP , Record 'OFSWOR.RTN'                          Insert      17:10:15
Command->
0001 *-----------------------------------------------------------------------
0002 * <Rating>-10</Rating>
0003 *-----------------------------------------------------------------------
0004      SUBROUTINE OFSWOR.RTN
0005      $INSERT I_COMMON
0006      $INSERT I_EQUATE
0007      $INSERT I_F.FUNDS.TRANSFER
0008
0009      Y.DRAC=R.NEW(FT.DEBIT.ACCT.NO)
0010      Y.DRCU=R.NEW(FT.DEBIT.CURRENCY)
0011      Y.DRAMT=R.NEW(FT.DEBIT.AMOUNT)
0012      Y.CRAC= R.NEW(FT.CREDIT.ACCT.NO)
0013      Y.CRCU= R.NEW(FT.CREDIT.CURRENCY)
0014      Y.MSG=""
0015      Y.MSG=Y.DRAC:',':Y.DRCU:',':Y.DRAMT:',':Y.CRAC:',':Y.CRCU:
0016 * SEQUENTIAL FILE CONTAINS THE RECORD IS ASSIGNED TO SEQ.FILE.NAME
0017      SEQ.FILE.NAME='RENU.BP'
0018 *RECORDNAME WHERE THE ACTUAL DATA IS PRESENT
0019      RECORD.NAME='MSG.IN'
0020 *OPENING THE SEQUENTIAL FILE AND ASSIGNING A PTR
0021 * TO THAT. IF FILE IS NOT PRESENT ERROR MSG IS GIVEN
0022      OPENSEQ SEQ.FILE.NAME,RECORD.NAME TO F.PTR ELSE
```

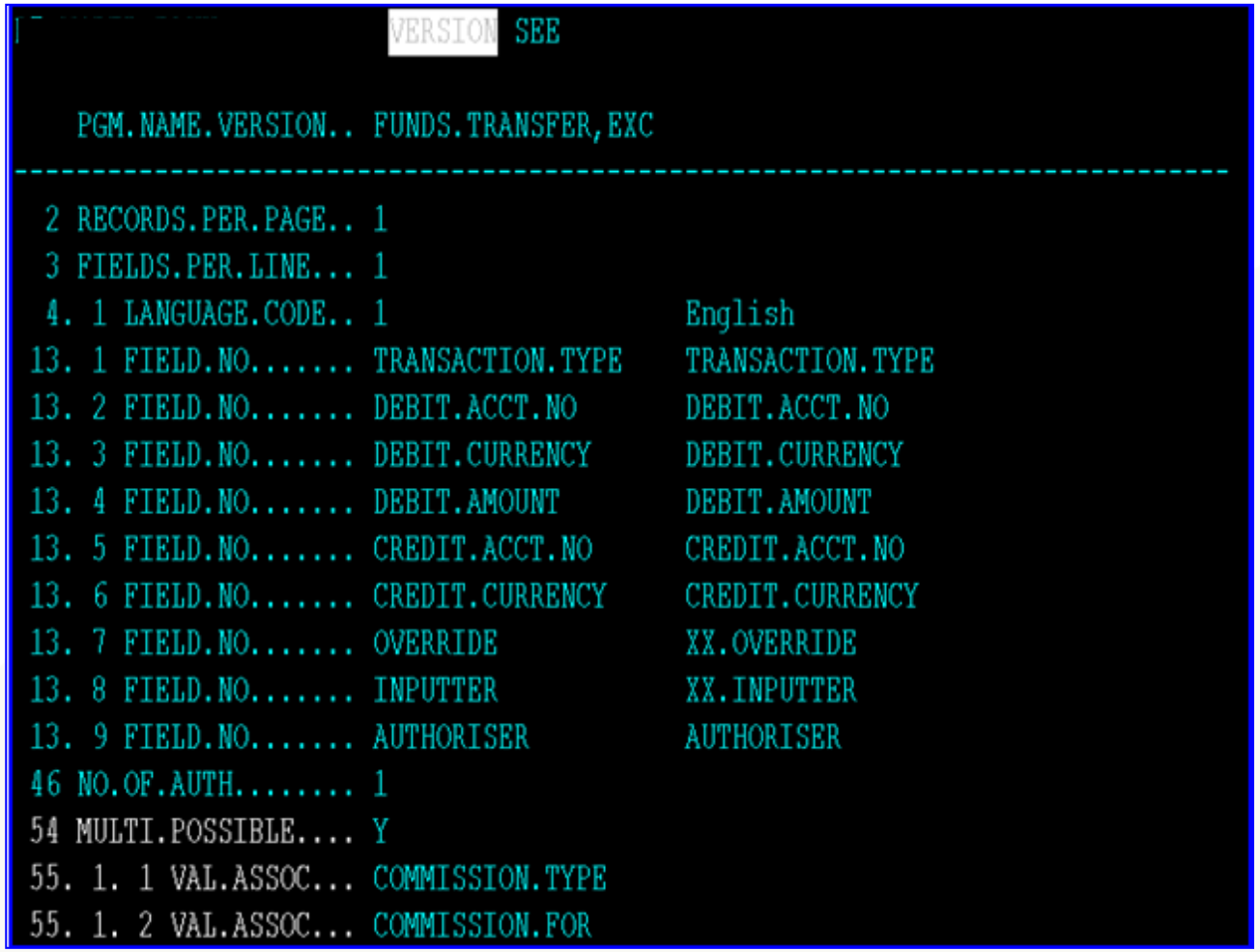
Solution – Step 2 : Routine (Contd)

```
0023              E="FILE NOT FOUND"
0024              CALL STORE.END.ERROR
0025         END
0026  *WRITING THE MSG TO THE TO THE PTR
0027  *IF ELSE DISPLAY MSG UNABLE TO WRITE
0028         WRITESEQ Y.MSG TO F.PTR ELSE
0029              E="UNABLE TO WRITE"
0030              CALL STORE.END.ERROR
0031         END
0032         CLOSESEQ F.PTR
0033         RETURN
```

Solution – Step 3 : PGM.FILE



Solution – Step 4 : Create Version



Solution – Output

```
File RENU.BP , Record 'MSG.OUT'                          Insert      12:14:15
Command->
0001 FT,EXC/I/PROCESS,SDEEPA/I/123456,FT0732300799/ ,DEBIT.ACCT.NO::=14637,DEBI
------------------------------ End Of Record -------------------------------
```

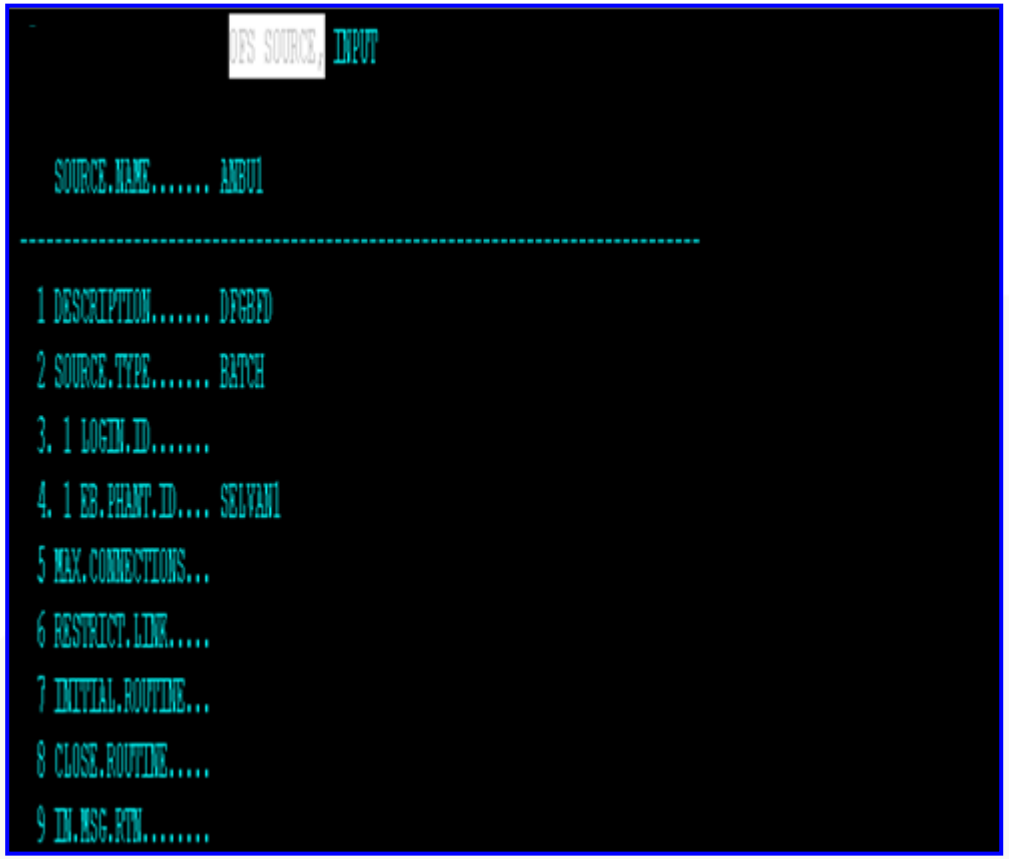
Workshop VII

Create OFS for Batch Mode

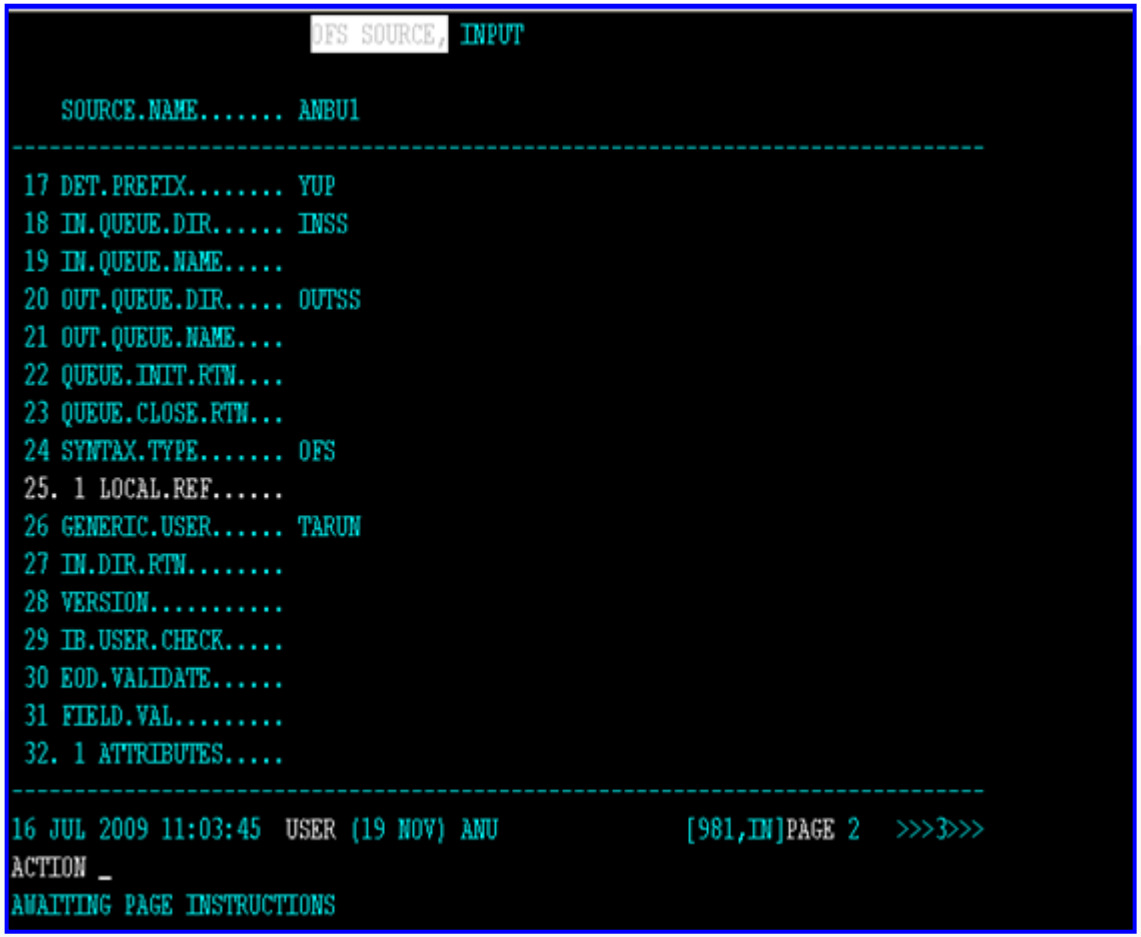
Then subroutine should write the formatted OFS messages to the in directory defined in this OFS.SOURCE record

If you were to receive any errors, check the request message to see if everything is correct about the syntax

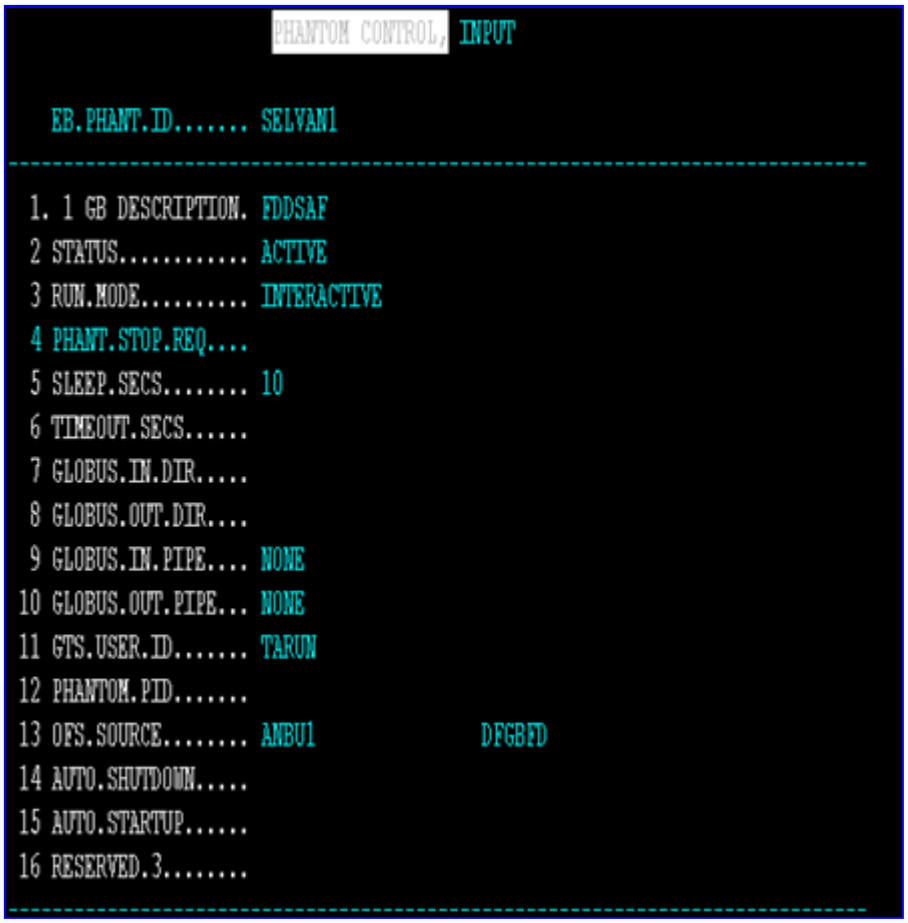
Solution – Step 1: OFS.SOURCE



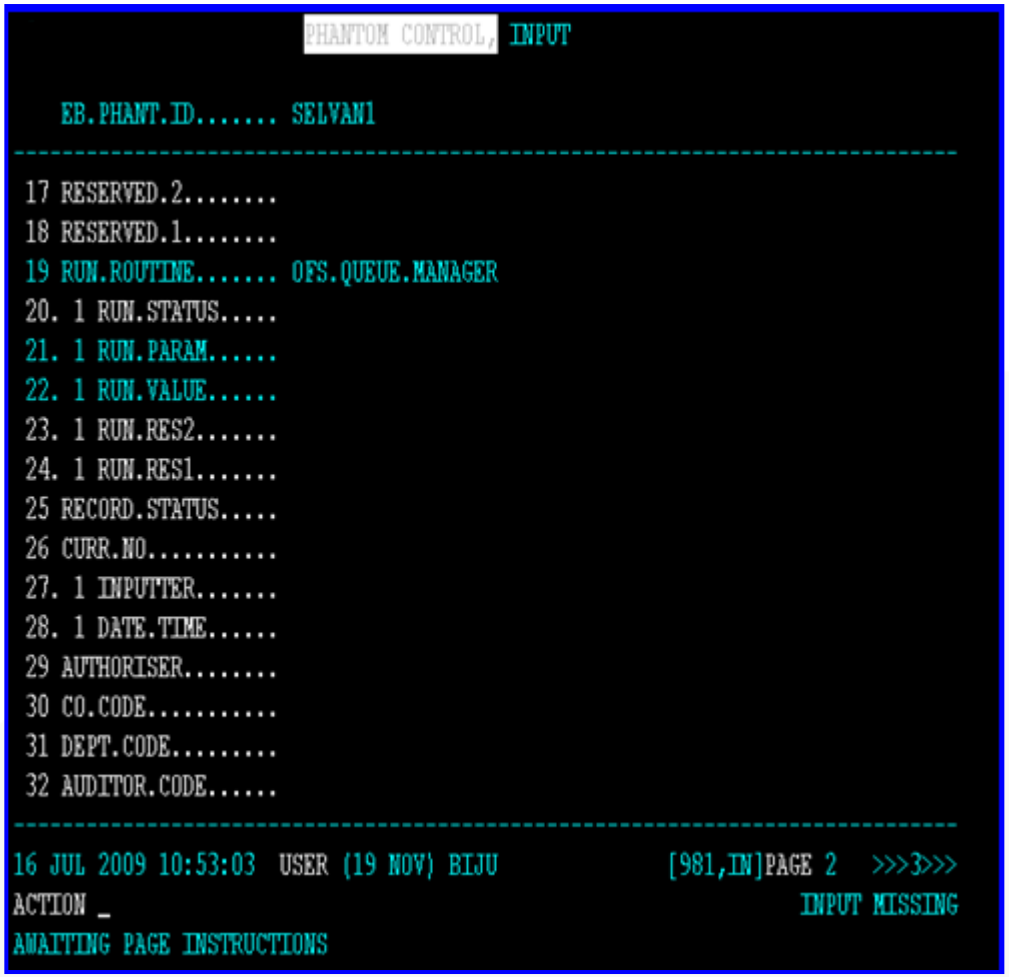
Solution – Step 1: OFS.SOURCE



Solution – Step 2 : EB.PHANTOM



Solution – Step 2 : EB.PHANTOM

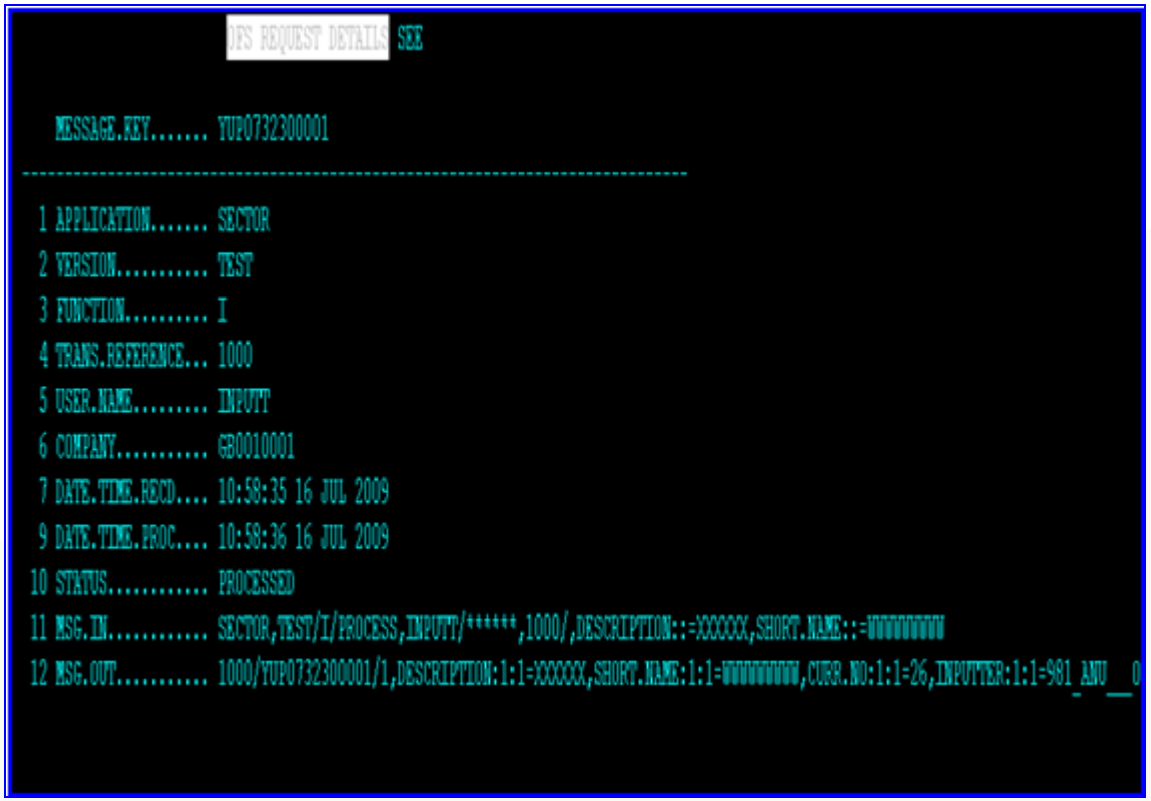


Solution – Step 3 : Input IN File

```
File INSS , Record 'MSG1'                                  Insert      10:57:15
Command->
0001 SECTOR,TEST/I/PROCESS,INPUTT/123456,1000/,DESCRIPTION::=XXXXXX,SHORT.NAME:
------------------------------ End Of Record ------------------------------
```

```
File OUTSS , Record 'MSG1'                                 Insert      10:59:26
Command->
0001 1000/YUP0732300001/1,DESCRIPTION:1:1=XXXXXX,SHORT.NAME:1:1=WWWWWWWWW,CURR.
------------------------------ End Of Record ------------------------------
```

Solution – OFS.REQUEST.DETAILS



ERROR HANDLING

Error messages related to the OFS module could be broadly classified in to three categories

- Message related
- System related
- File related

Message related

These error messages are encountered with the format of request messages

To overcome this ofs messages have to analyzed and corrected before we pass the message

Message related

APPLICATION / OPERATION MISSING

- This error message is encountered when the request message is either empty or does not contain the name of the application / subroutine / Enquiry select (in case of enquiry messages), in the first portion of the request message
- To overcome this, supply the required name in the first portion of the request message

Message related

NO SIGN ON NAME SUPPLIED DURING SIGN ON PROCESS

- This error message is encountered when the request message does not contain the TEMENOS T24 user sign on name / password in the User information portion of the request message
- To overcome this, supply a valid TEMENOS T24 user name and password in the User information portion of the request message

SYSTEM RELATED ERROR MESSAGE

These error messages happen when the server is offline or when the restriction in the application used in the transaction

The following is the list of common system related error messages

<Message ID>/<Transaction ID>/-3/OFFLINE

This error message is encountered when the TEMENOS T24 Server is in offline status at the time the request message is received by it

In this stage inputs / updates to the database will not be allowed

File I/O related errors

The errors are encountered when there are problems related to files

- File cannot be opened
  - This error message when the open operation on the file is failed
  - To avoid this ensure that whether permission to access the file is given

File I/O related errors

Missing Standard Selection on the Application

- This error message is displayed when the Standard selection entry for this application is not done
- To overcome this ensure whether standard selection record for this application is available
- Else create SS for this application

NOINPUT Fields

In normal T24 application a field in a application will be defined as NOINPUT field

If values to such fields are passed through OFS we will get these errors

To overcome this ensure that the field is enabled as INPUT field

Summary

Functionality of OFS.

Architecture of OFS.

Message Structure of OFS.

Components of OFS.

Passing Message through OFS.

To display enquiry data using OFS.

OFS message to access Routine.

Data capture using OFS.

Capgemini

CONSULTING.TECHNOLOGY.OUTSOURCING

www.capgemini.com/financialservices

TOGETHER. FREE YOUR ENERGIES

COLLABORATIVE BUSINESS EXPERIENCE