Query Management System

PROJECT REPORT

Dissertation Submitted in partial fulfilment of the requirements for the Degree of

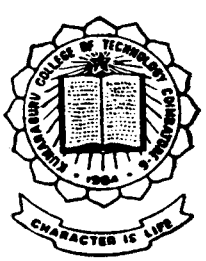
MASTER OF COMPUTER APPLICATIONS

of the Bharathiar University

By

ANIL K. JANS

Reg. No. 9438MO188



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Kumaraguru College of Technology

COIMBATORE - 641 006.

JUNE 1997

CERTIFICATE

This is to certify that this project work entitled

" QUERY MANAGEMENT SYSTEM "

submitted to Kumaraguru College of Technology, Coimbatore (affiliated to Bharathiar University) in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications is a record of original work done by ***Mr. ANIL .K. JANS, Reg. No. 9438MO188*** *during his period of study in the Department of Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore under my supervision and guidance and this project work has not formed the basis for the award of any Degree/Diploma/Associateship/Fellowship or similar title to any candidate of any University.*

Professor and Head

Staff in-charge

Submitted for University Examination held on / /1997

Internal Examiner

External Examiner

DECLARATION

I here by declare that this project work entitled

"QUERY MANAGEMENT SYSTEM"

submitted to Kumaraguru College Of Technology, Coimbatore (Affiliated to Bharathiar University) is a record of original work done by me under the supervision and guidance of ***Mr. D. RAMESH*** *M.E M.I.S.T.E. lecturer, Department of computer science and engg, Kumaraguru College Of Technology, and that his project work has not formed the basis for the award of any Degree / Diploma / Associateship / Fellowship or similar title to any candidate of any university.*

Place : Coimbatore;

Date:

Signature of the candidate

(ANIL .K. JANS)

Counter signed by

Staff in-charge

Mr. D. RAMESH *M.E M.I.S.T.E.*
Senior Lecturer
Department of Comp. Science & Engg.
Kumaraguru College Of Technology.
Coimbatore.

WIPRO SYSTEMS

A Part of Infotech Group
and A Division of WIPRO LIMITED

Business Office : 40/1A, Lavelle Road, Bangalore - 560 001. India
Tel. : 91-80-2215010 / 2215014, Fax : 91-80-2271954 / 2271579

APPROVAL

of the project for MCA degree by

Anil . K . Jans

This is to certify that Anil .K. Jans has completed his project titled 'Query Management System' with Wipro Systems, Madivala, Bangalore in partial fulfillment of the requirements for the degree of Master of Computer Applications .

The project was completed during the period January 1997 - June 1997. His work was found satisfactory in all regards and we recommend that appropriate credit to be given to his work.

The project report has been reviewed by us and has been found to be satisfactory regarding the content and is ready to be submitted to the university.

Date : 14/5/97

R. Balasubramaniam,
Technical Manager,
Wipro Systems,
Madivala,
Bangalore - 560 068

Acknowledgment

I wish to express my heartfelt thanks to our Principal Dr. S. Subramaniam, for his encouragement and counseling. I also thank Prof. P. Shanmugham, Head of the Department, Department Of Computer Science & Engineering, for his valuable guidance.

I express my gratitude to Mr. D. Ramesh, Lecturer, Department Of Computer Science & Engineering , for providing me constant support and encouragement for successfully completing this project.

There are no adequate words to express my sincere gratitude to Mr. B.R. Nagaraj, Facilities Manager, Wipro Systems, for kindly recommending me for the project work at Wipro Systems.

I express my sincere thanks to Mr. R. Balasubramaniam, Tech Manager, Wipro Systems, for providing me this opportunity to do my project in Wipro Systems. I also express sincere thanks to Mr. Johnson B., Associate Consultant, for giving me this opportunity to work under his supervision and has been a constant source of guidance and support throughout my project.

Let me admit that I really enjoyed carrying out my project work in the friendly and informal atmosphere of this group. I thank whole heartedly all my team members and friends in this organization for their co-operation and timely help.

SYNOPSIS

Query management system (Q.M.S) has been designed at WIPRO SYSTEMS, Bangalore. The QMS is a tool to assist in the communication of problems found during the development of a software product. It is an ISPF dialog. It is a communication tool typically used between Tester and Developer. Q.M.S provides facilities for answering queries and keeping records of all the details of queries and answers for future cross reference.

Q.M.S has been implemented in the IBM mainframe ES/9000. The operating system is MVS/ESA. Languages used are CLIST, VS-COBOL and JCL. VSAM files are used to store information. ISPF (Interactive System Productivity Facility) functions are used for creating a user friendly and interactive user interface. Operating environment of this package is TSO (Time Sharing Option)

This tool has been developed keeping in mind the need of the user. It is completely menu driven and data validation procedures have been included where ever necessary to avoid the user entering irrelevant data. In addition to this a tutorial is provided to guide the novice user.

Contents

CONTENTS

Introduction

Chapter 1

Introduction

1.1 Organization Profile

Wipro's Information Technology business has a 15 year track record of continuos growth. Wipro Ltd. comprises Wipro Infotech Group, Wipro Consumer and Lighting Group, Wipro Fluid Power and Wipro Biomed. The other units forming a part of Wipro corporation are Wipro Finance, under whose umbrella are Wipro Factors and Wipro Securities, and three joint venture companies - Wipro GE Medical Systems, Wipro BT and Wipro Acer.

Wipro Systems Division under Wipro IT Group is engaged in the development of software and export of the same. Wipro Systems Division is the second largest software exporter with customers who are the world leading technology providers, system integrators and large end users. The list includes names like GE, AT & T, Northern Telecom, Allied Signal, Tandom, Xerox and Seagate. Wipro Systems Division is specialized in client/server development, communications and networking technology in addition to solve problems like Year 2000. Wipro Systems have excellent infrastructure that provides offshore development with multiple 64KB high speed data communication links. These global development centers functions as virtual extension of customer's development environment. Wipro Systems Divisions has been accredited for ISO9000.

Wipro Infotech group is the only Information Technology company in India which has maintained continuos growth on revenues and profits since its inception.

Wipro BT is offering electronic mail and other value added services.

Wipro GE Medical Systems is a joint venture with GE Medical Systems - Asia which is the Asia - Pacific regions business activity focus. This alliance is meant to support a wide range of diagnostic medical imaging and therapy systems.

Wipro Finance is engaged in serving customers with capital resources, expertise and ideas to meet their financial needs.

Wipro Computers & Peripherals Division provides high quality Wipro Acer range desktops and servers.

Wipro Solutions & Systems Integration Division offers products and services to build private local area and wide area networks.

Wipro Support Division maintains all products, does network audit and facilities management of networks.

With so much people - clients, professionals, users etc. involved, covering a large area of the globe, spontaneous growth in pace with emerging new Information Technologies, Wipro Limited is still striving to 'keep people together'.

1.2 Overview of Query Management System

QMS is a tool for recording and tracking problems encountered during the development stages of a software. It is a ISPF dialog. It records necessary key data plus a free format description of problem and its solution. It is a communication tool typically used between testers and developers.

This tool contains three main modules viz; Functions, Reports and Support. Module Functions contains sub modules which support query manipulation. Support contains functions for setting access rights, maintaining user information and project information.

A number of reports are produced for management/project leaders use. Module Reports contains options for producing reports.

In addition to these one help module is included, Which contains two options viz; Tutorial and Help index. The tutorial provides you with an overview of the QMS and gives you an introduction about the main features of the tool.

All the main functions are written in VS COBOL. When the user enters his option in a selection panel corresponding function is called. In COBOL programs, ISPF functions are used to display input and output screens and for displaying messages. On-line help is provided with each and every screen. You can invoke on-line help by entering key for the help menu, usually <F1> key, or by typing 'help' in the command prompt.

QMS runs in an ISPF environment and follows the standard ISPF conventions for Screen layouts and function keys.

A CLIST program is first invoked to start QMS. All the temporary dataset needed by the application is dynamically allocated in this program. After the execution it will be automatically deleted.

Operating Environment

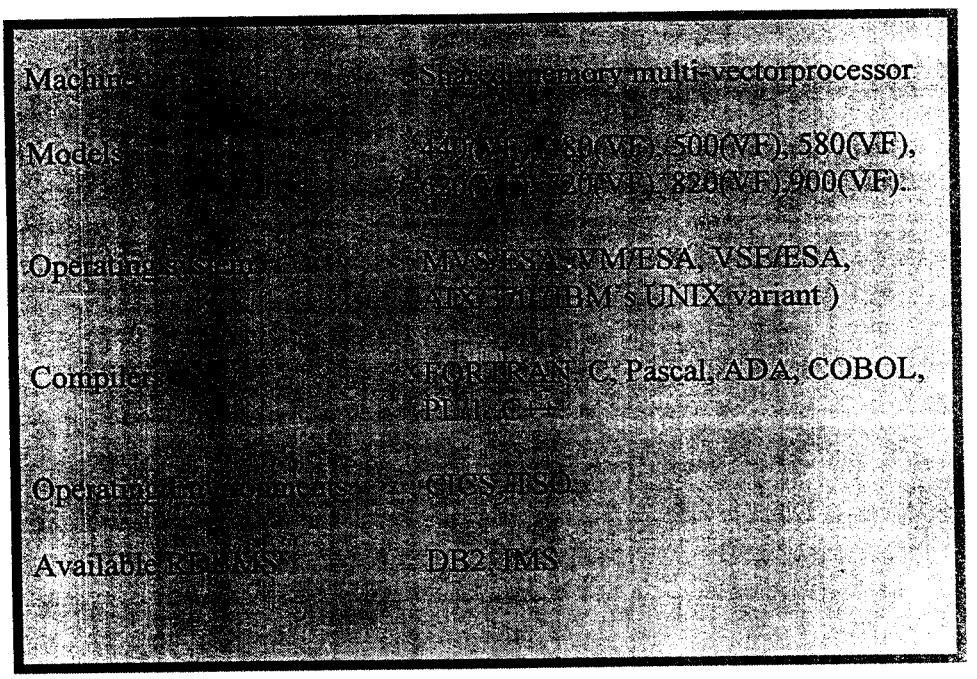
Hardware used	: IBM Mainframe ES/9000
Operating System	: MVS/ESA
Operating Environment	: TSO (Time Sharing Option)
Languages	: VS-COBOL, CLIST, JCL
Files	: VSAM KSDS
Other Tools	: ISPF

System concept

Chapter 2
System concept

2.1 The IBM ES/9000

The IBM ES/9000 is one of the most popular mainframes, used all over the globe. IT has shared-memory multivector processor which has a clock cycle of 7ns. It is a 64-bit machine with 1 GB of main memory and 8 GB of expanded memory.



All the peak performance numbers quoted above are for the maximal configured system, that is all processors are equipped with a vector facility (VF). Vfs may be installed or left out as required by the user. Enhancing the normal processor with a VF was an idea that already was implemented in the ES/9000 predecessor, the IBM 3090. Instead of cache that has to be shared among processors, each processor has its on cache of 4MB. In addition the memory hierarchy has increased by another layer by adding separate high speed buffers of 128 KB each for data and instructions. This should compact the problems that some times were experienced in the 3090 system with data starvation of the functional units. As in the 3090 systems, in the ES/9000 system the relative data bandwidth is low : 8 bytes/cycle, i.e., one 64-bit operand or result can be loaded or stored per cycle.

An interesting enhancement for the high-end 820 and 900 models is that their Vfs are able to produce 4 results per cycle which, for suitable code, should lead to a very respectable 563 Mflop/s per processor.

Only FORTRAN can be used for parallel processing. Parallel processing is accomplished by (a)symmetric processing with process synchronization via memory. There are no hardware provisions like communication registers which makes synchronization rather costly. The system is aware of which processor have a VF and which have not and act accordingly without any required action of the user.

We can have 1-8 processors in a single machine, operating in a band width of 889 MB/s which gives lot of strength to the giant machine. Normally the operating system will span across the processors. Even operating environments like CICS can span across the processors. On the same machine we can also have different instance of same operating environment. We can also have different operating environments at the same time.

2.2 Multiple Virtual Storage (MVS)

MVS is IBM's flagship operating system that runs on its largest computers. Its is a high-throughput product that sacrifices user-friendliness for speed. MVS can manage extremely large data and totally different kinds of programs running all at the same time. It can handle real-time, teleprocessing, program development, program testing batch updating and reporting, hierarchical and relational database management systems, personnel programming and reporting tasks and electronic mail. MVS gives each task the level of service it needs and gets as much as useful work done as possible. If you have MVS you'll be able to run all the product IBM sells and all the ones that are planned.

Lets us now see how MVS starts. Loading the operating system is called IPL (Initial Program Load). IPL actually loads a small part of the operating system into memory. This small part then loads the rest of the operating system. The IPL process starts the first job on the system; the Master Schedular. Then IPL says " Sayonara" and the Master Schedular takes over and directs all the work from then on.

The Master Schedular starts the System timer, system logging, and recording and JES2 or JES3 , depending on which you have installed. It then tells MVS to start the initiators running.

The initiators are programs that have one purpose : to find a job to start running and then go backstage. These active little programs look for the next important job that needs to be run, load it into memory and then go out for a soda. When the job finishes, they come back in and clean up after the program, take care for the files and flushing out memory for next program. Then they look for another job to start , and it begins all over again.

The initiator can start up three kinds of jobs.

1. JCL job - That's what we are doing.

2. Started task - Started from the console with a command. uses JCL stored on system library, higher privileges than any job.

3. TSO session - uses JCL stored in the system library to start an interactive session.

When some one submits JCL to MVS in order to get some useful work done, a series of things happens. The JCL is checked for syntax and interpreted. Any procedure library JCL that is called for is found and incorporated into the JCL that was submitted. JCL statements that refer to the file are processed . Files are found or created and connections to the program is established. The program called for in the JCL is located on the program library, loaded into memory, and started up. The

program opens the files, and MVS completes the connection to the program, prepares the files for reading or writing . If the program doesn't violates any MVS's rules, it reaches the end of its processing and returns control to MVS. The initiator comes back in and prepares for the next program or job.

JOB MANAGEMENT

A job is a combination of related requests for system services. Each job runs one or more programs, each accessing one or more data files. A job typically corresponds to one person's need to have a specific work request fulfilled- for example, to produce a report or to update a file. MVS manages a job as a whole. It insulates a job from all others so that one job cannot directly affect any other. Each job has its on memory, which no other job can look at. Each job is timed separately and is given its own priority. All the jobs printed output is printed together if desired. A job can "own" a file while its running. MVS will enforce ownership and keeps other jobs from owning the same file. MVS, the operator, and the TSO user can all cancel a job while its running.

PROGRAM MANAGEMENT

When you ask MVS to run a program, it must find the program on a disk library and is must find enough memory for the program to run in . It will pass up to 100 characters of information to the program if you ask it to, and it will start the program executing at its first instruction. It will time the program, making sure it doesn't use the CPU too long, and monitor the program to make sure that it doesn't violate any rules, such as trying to look at some other job's memory or trying to add two persons name together. If the program does violate some rule, MVS print out the contents of memory in what is called a dump for the programmer to pull hair over.

DATA MANAGEMENT

The operating system can locate data and see to it that it gets into the program that requests it. It performs thousands of tedious tasks necessary to move data from its place on the tape or disk into the program. It places magnetically encoded labels on the data so it can later verify that it indeed has the right data, and can keep track of all the data in the entire computer center by means of its all-encompassing catalog.

TYPES OF DATA FILES

Data files differ in their internal structure. The types are as follows

1 Ordinary, called sequential or physical sequential datasets by IBM

2 Libraries called PDS's or partitioned datasets by IBM

3 VSAM, modern types of file structures created and managed by a special utility called IDCAMS

THE CATALOG

A typical computer center has thousands of tape reels or cartridges and hundreds of disks packs. Each disk pack contains thousands of files. Just so you won't have to go searching for a needle in a haystack, MVS provides a sophisticated method of locating the exact file you want - the CATALOG. The catalog records the name and location of every dataset in the computer . The system catalog tells MVS how to find each dataset indicating exactly which volume the dataset is on. A volume is a reel or cartridge of tape or a disk pack.

2.3 Job Control Language

Introduction to JCL

Job control language is used to request the resources that the job needs to run the system. The main functions of JCL is as follows.

1. Introduces job to the system
2. Provides accounting parameters
3. Requests for the resources
4. Identifies the programs to be executed
5. Defines the scheduling information
6. Deferred execution (It can be executed as soon as it is SUBMITTED, or later)
7. No user intervention during running
8. Statically known resource requirements
9. Generally used for much time consuming jobs
10. Syntax checking & conversion Reader
11. Allocation and execution Initiator
12. Job purge

Code the JCL in a sequential or member of a Partition Data Set. To submit the JCL give following commands:

- On command line:
  ===> TSO SUBMIT Dataset name

- Inside editor:
  ===> SUBMIT

- From ready prompt:
  Ready
  SUBMIT Dataset name

Use System Development and Search Facility of ISPF

===> ST Shows the status of the job
===> H Shows the jobs in the JES2 held output queue
===> DA Displays all active jobs
===> I Shows the jobs in the JES2 input queue
===> O Shows the jobs in the JES2 output queue

General format of JCL

- A JCL statement consists of one or more 80 byte records
- Each record is in the form of an 80 column punched-card image
- All JCL statements begin with // in the first two columns (except the delimiter or JES2 control statements which begin with /*)
- All columns from 1 to 71 can be used for coding JCL statement
- Position 72 is used to embed comment continuation
- Positions from 73 to 80 are used for numbering purposes
- Each JCL statement is logically divided in to five fields

JCL statements have the following general syntax

identifier name operation parameter comments

Identifier

Identifiers used in JCL statements are the following:

// JCL statement
/* delimiter
//* comment

For example:

//myjob job trnact1,trng001,class=a my job stmt ; Indicates to the system that a statement is a JCL statement

Name

Identifies the particular statement. Other statements or system can refer a statement by its name. The name must begin in column 3.
Syntax rules:

- 1 to 8 alphanumeric or national ($, #, @)
- The first character must be alphabetic or national
- Name must be followed by at least one blank

Operation

Specifies the type of the statement and must be preceded by and followed by at least one blank.

Parameter

is a list of positional and/or keyword parameters separated by commas. Positional parameters must be coded in order; keyword parameters can be coded in

any order. Parameters are separated by commas. This field must be preceded and followed by one or more blanks. Blanks are not allowed between parameters. The parameter field must begin before column 16 and may not extend past column 71: however, the field may be continued on to another statement by stopping after a comma and coding // in columns 1 and 2 of the next statement followed by at least one blank and the rest of the parameters.

Comments

are optional if operand are coded. Comments must start past column 17 if operand(s) are not coded

Job Statement

- Marks the beginning of the job
- Tells the system how to process the job
- Defines job & job related information
- Null statement or another job statement or end of file marks the end of a job

Example:

```
//jobname   JOB       accounting-info,programmer-name,
//          CLASS=jobclass,TYPRUN=option,
//          MSGCLASS=output-class,MSGLEVEL=(x,y),
//          NOTIFY=userid
```

Assigns the job to a JES input class. The specified class must have been defined to JES; if omitted JES uses the installation default. System assigns no meaning to an input class, it is the installation which differentiates the jobs depending on their nature .

For example,

```
//testjob job trnac01,trng01,class=a
```

MSGLEVEL=(x,y) Controls the listing of the job log. The first parameter controls which statement will be included in the job log.

0 - Print only job statement
1 - Print all JCL & JES statements including all statements encountered in the procedure
2 - Print only Submitted JCL & JES statements

The second parameter controls which message will be included in the job log

0 - If normal termination print only JCL messages. If abnormal termination print all messages
1 - All messages are printed regardless of how job terminates

TYPERUN = option requests special job processing. When SCAN is given JCL is scanned for syntax errors, but will not be executed.
For example:

```
//testjob job trnac01,trng01,class=a,msgclass=a,
//        MSGLEVEL=(1,1),typrun=scan
```

NOTIFY = userid Informs specified TSO user when the job terminates. If the job terminates while the user was logged off, the message will appear when the user logs on.
Example:

```
//testjob job trnac01,trng01,class=a,msgclass=h,
//        MSGLEVEL=(1,1),typrun=scan,
//        notify=trng02
```

EXEC Statement

- Marks the beginning of a step
- Identifies the program or procedure to be executed
- Maximum 255 steps are allowed in a job
- Another exec statement or a job statement or a null statement or end-of-file marks the end of the step

Example:

```
//stepname  exec pgm=programe,parm='parameters'
```

Positional parameter

Pgm = progname Identifies the program to be executed in a step. The specified program must be a member of system / private / temporary library.
Example:

```
//stepone EXEC PGM=IFOX001
```

Positional parameter Identifies the procedure to be called and executed. The procedure can be CATALOGED or INSTREAM procedure. You can omit PROC and code only procedure name.
For example:

```
//steptwo EXEC PROC=gddmcomp
```

Or

```
//steptwo EXEC gddmcomp
```

Parm=parameters provide a way to pass the variable information to the processing program executed by the step. Length of the parameter passed must not exceed 100 characters.

- Information is included in the apostrophes or brackets
- If more than one subparameter are coded, separate them by commas
- If the PARM is continued to next line, it must be enclosed in parenthesis

For example:

```
//cobrun exec prg=myprog,parm='one,kkk,llll,xxxx'
Linkage section:
01 PARM
     05  plength  pic s9(4) comp
     05  pcontnt  pic x(100)
```

DD Statement

Describes a Data Set and specifies the input and output resources needed by the Program. A DD statement is required for each Data Set that is to be processed in each step. The name of the DD statement is coded in the program. When the Data Set is opened for processing the name is used to locate proper DD statement. From the Data name specified on DD statement, system locates the physical file

The format of the DD statement can be one of the following:

```
//ddname DD*
//ddname DD DATA,DLM=xx
//ddanme DD DUMMY
//ddname DD SYSOUT=(calass,forms)
//ddname DD DSN=dataset-name,
// DCB=(BLKSIZE=b,DEN=D,LRECL=N,RECFM=r,TRTCH=t),
// DISP= (Status, Normal_termination, Abnormal_termination) ,
// LABEL=(file,label-type,Processing-type)'
// SPACE=(Type,(primary,secondary,directory-blocks),RLSE),
// UNIT=device,
// VOL=SER=name
```

Each DD statement should have a unique DDNAME within a step. If a duplicate DDname is encountered in a step, it is ignored.

DSN=DataSetName

- Identifies the name of the Data Set to be created or retrieved
- A maximum of 44 characters can be coded for qualified name
- Maximum of 8 periods
- Maximum of 8 characters between two periods

For example:

```
//inpdat1 DD DSN=wipro.trnggrp.cobol.source
For temporary Data Set
//tmpdata DD DSN=&&tmpfile
```

DISP= (Status, Normal_termination, Abnormal_termination)

- Describe the status of a Data Set (to be created/ retrieved)
- Describes how to dispose of the Data Set when step terminates

Status = NEW/OLD/SHR/MOD

Normal-termination = KEEP/DELETE/ PASS/CATLG/UNCATLG

Abnormal-termination = KEEP/DELETE/CATLG/UNCATLG

NEW	New Data Set is to be created - Exclusive control
OLD	Exists before this step - Exclusive control
MOD	If Data Set exists before this step data will be appended. If it does not exist it will be created - Exclusive control
SHR	Exists before this step - Can share with other job
KEEP	Will be kept
PASS	Is to be passed for use by subsequent steps in the same job
DELETE	Will be deleted
CATLG	Creates the catalog entry
UNCATLG	Deletes the catalog entry

2.4 Introduction to TSO/E

TSO/E is a facility of the MVS Operating System that allows users to interactively share processor time and resources.

TSO /E is a tool to

- Access MVS resources/facilities;
- Develop and Maintain Programs;
- Communicate with other users.

TSO/E is a job running under MVS

- Once started, TSO/E will be waiting for users to Logon
- Creates separate address spaces for each user
- Loads terminal monitor program (tmp) into each address space
- Provides services for other applications running under TSO/E

Line mode

- Basic default mode on user logon
- Similar to prompts provided in other OS (UNIX $, DOS >)
- Denoted by READY

ISPF / PDF

- Interactive System Productivity Facility (ISPF)
- Program Development Facility (PDF)
- Menu driven application under TSO/E
- Uses TSO services implicitly

Issuing commands

Command name followed by one or more operands

Two ways of issuing commands

- Enter command name and let system prompt you
- Enter command name and parameters Command aliases Reentering commands

Logging on to the System

TSO/E logon session

- Initiated by terminal input (logon CMD)
- Logon process
- Interacting with TSO/E
- Logging off

Used to identify user to system and request use of its resources

Syntax

Logon [user_id]

Userid

Maximum 7 characters

First character : alphabet

Rest : combination of (0-9) and (a-z)

Logon Processing

- RACF checking for userid / password authorization
- Creation of address space
- Initiation of TMP
- Ready prompt

Exceptions During Logon

- Userid in use
- Reconnect facility
- Password restrictions

Sign off to end the terminal session (logoff CMD)

System releases the userid

Command syntax

Logoff

Relogon

DATASET OPERATIONS

A Dataset is an unit of information that can be stored and retrieved.

Types

- Sequential Dataset
- Partitioned Dataset - members
- VSAM Dataset

Dataset name consist of one or more parts (qualifier) separated by `.`

Each qualifier must begin with alphabet. Can consist of alphabet, numbers and special characters. It can have a maximum length of 8 characters. Maximum length of the name is 44 characters. Member names are specified within brackets.

For example:
Sysusr9.source.cobol(mem1)

Dataset names consist of three level qualifiers:

- First qualifier - user_id/project id
- Second qualifier - user defined
- Third qualifier - implies certain characteristics

For example:

Asm, cobol, data, load

Name of the Dataset

Space parameters - primary / secondary

Dataset organizations

- Physical Sequential (PS)
- Physical Ordered (PO)
- Directory blocks (DIR)
- Logical RECord Length (LRECL)
- RECord Format (RecFm)
- Device Type (Unit)
- Volume Id (Volume)

Allocate Command Syntax

Allocate / Alloc Dataset (Dsname)
New
Volume (Volume_Id)
Space (Qty , Inc)
Block/Avblock / Tracks / Cylinders
Dsorg (Po/Ps)
Dir (N)
Unit (Type)
Blksize (Value)
Lrecl (Value)
Recfm (Recfm)
Like (Dsname)
Using (Attrlist)

Delete one or more Datasets

Delete a member of a partitioned Dataset

Uncatalog a Dataset

Delete / del <dsname list >

For example,

delete 'userid.cobol.data' 'userid.text'

Renaming Datasets (Rename CMD)

- Change the name of one or more cataloged Datasets
- Member of a partitioned Dataset
- Wildcards are allowed

Rename Command Syntax

Rename / ren <old-name> <new-name>

For example,

rename 'userid.*.data' 'userid.*.text'
rename *.data *.text

Printing Datasets (PRINTDS CMD)

- Sequential/partition Datasets
- Complete/part of Dataset

PRINTDS Command Format

PRINTDS / PR Dataset (Dataset name list)
columns (startcol , endcol)
lines (startln / endln)
copies (number)
class/output (outclass)

Help facility

Retrieves the following information:
Functionality of command
Command syntax
Command operands
Subcommands
Messages
Help command syntax

2.5 Introduction to CLIST

The CLIST language enables you to work more efficiently with TSO/E. You can write programs, called CLISTs, that perform given tasks or group of tasks. From then on you can simply invoke the CLISTs to do those tasks.

The term CLIST(pronounced as "sea list") is the short form of COMMAND LIST, because the most basic CLISTs are list of TSO/E commands. When you invoke such a CLIST, it issues TSO commands in sequence.

Besides issuing TSO/E commands, CLISTs can perform more complex programming tasks. The CLIST language includes the programming tools you need to write extensive, structured applications. CLISTs can perform any number of complex tasks, from displaying a series of full screen panels to managing programs written in other languages.

The CLIST language is an interpretive language . Like programs in other higher level interpretive languages, CLISTs are easy to write and test. You don't have to compile and link them . To test a CLIST, you execute it, correct any errors, and re-execute it .

Features of CLIST language

The CLIST language provides a wide range of programming functions. Its features include :

- An extensive set of arithmetic and logical operators for processing numeric data.
- String handling functions for processing character data.
- CLIST statements that let you structure your programs, perform I/O, define and modify variables , and handle errors and attention interrupts.

Categories Of CLISTs

A CLIST can prefer a wide range of tasks . Following are the three general categories of CLISTs :

- CLISTs that perform routine tasks
- CLISTs that are structured application
- CLISTs That Manages applications written in other languages

As a user in TSO/E, you probably can perform certain tasks on a regular basis. These tasks may involve entering TSO/E commands to check the status of data sets, to allocate data sets for n particular programs, and to print files.

You can write CLISTs that significantly reduce the amount of time that you have to spend on these routine tasks. By grouping together in a CLISTs the instructions required to complete a task., you can reduce time, number of keystrokes, and errors involved in performing the task; thus you can increase the productivity. Such a CLIST can consists of TSO/E commands only, or a combination of TSO/E commands, JCL statements, or CLIST statements.

Example of CLIST that contains TSO/E commands :

```
allocate file(ABC) dataset(name1)
allocate file(DEF) dataset(name2)
call (prog1)
free file(ABC DEF)
```

The CLIST in the example issues TSO/E commands to allocate files for a program, call the program , and free the files when the program is finished . When ever you want to perform these related tasks, you can simply execute the CLIST instead of retyping the commands. If the tasks require input from the user, you can obtain the input in a CLIST using the CLIST statements or TSO/E commands to prompt the user for the input.

CLIST That Are Structured Applications

The CLIST language includes the basic tools you need to write complete, structured applications . Any CLIST can invoke another CLIST, which is refereed to as nested CLIST . CLISTs can also contain separate routines called subprocedures. Nested CLISTs and subprocedures let you separate your CLISTs into logical subunits and put common functions in a single location. Specific CLIST statements let you :

- Define common data for subprocedures and nested CLISTs
- Restrict data to certain subprocedures or nested CLISTs
- Pass specific data to a subprocedure or nested CLIST.

For interactive applications CLISTs can issue commands of Interactive System Productivity Facility (ISPF) to display full screen panels. Conversely, ISPF panels can invoke CLISTs, based on the input that the user types on the panel. When the user changes a value on a panel , the changes applies to the value in the CLIST that displayed the panel . With ISPF CLISTs can manage extensive panel-driven dialogs.

CLISTs That Manage Applications Written In Other Languages.

You might have access to applications that are written in other programming languages. However, the interfaces to these applications might not be easy to use or remember. Rather than write new applications, you can write CLISTs that provide easy-to-use interfaces between the user and such applications.

A CLIST can send messages to, and receive messages from, the terminal to determine what the users wants to do. Then, based on the information, the CLIST can set up the environment and issue commands required to invoke the program that performs the requested tasks.

2.6 ISPF(Interactive System Productivity Facility)

The ISPF (Interactive System Productivity) is an IBM product designed to improve productivity by simplifying and standardizing the process of communicating with an end-user in a variety of operating environment. This process the dialog, is effected through full screen displays.

An ISPF dialog is composed of any number of elements arranged in various combinations from the following set of components:

- FUNCTIONS
- PANELS
- MESSAGES
- TABLES
- FILE SKELETONS

Functions

A function is nothing more than a program or a command language procedure designed to control the logic of a given task with in a user dialog.

Panels

Panels are the full screen displays that the end user sees. They can be structured in a variety of ways depending on their purpose. Generally there are four type of panels.

- Selection panels
- Data entry panels
- Table display panels
- Tutorial panels

ISPF communicate with user through a series of pre defined panels. Panels may require a response and that response determines the next panel to be displayed or the function to be performed. Responses are case-insensitive, upper lower or mixture of the two is valid.

Selection panels shows the list of options representing the different paths available to the user. After the user indicate the desired selection, and depending on the complexity of the application, the next display may be another lower-level selection panel, or it may lead directly into a data or table display panel.

Data entry panels are panels through which data is entered or retrieved from the system.

Since displaying a table involves a repetitive process for each of its entries (rows), a table display panel is a specialized version of a data display panel.

Tutorial panels are designed to provide assistance to the end-user at any level of an application. These panels normally contains nothing more than text in a manner similar to a page in a book.

Panel format

All panels are formed to fit on a 24-line x 80 character screen, with the first three lines formatted as follows:

Messages

AS a user proceeds through dialog, it is important to keep that user informed of any errors detected or of certain action taken. This information is normally relayed through messages that overlay certain predetermined section of the screen.

Tables

Tables represent a two dimensional matrix of data where the rows are similar to logical records on a file, and the columns are similar to data fields with in a record. Although tables resides in disk to process them they must be in core.

One may access the various rows of table in a variety of ways:

- By key field(s)
- By data fields
- By row number

File skeletons

File skeletons are pre formatted files where the basic structure of the desired output is established, but the variable data has to be inserted on demand. This process is called *file tailoring*. File tailoring can be used to construct job streams, create output reports, or produce any file that may be used for some other process.

ISPF Services

Generally, ISPF provides services to perform many complex tasks such as:

- Displaying screen services and messages
- Building and maintaining tables
- Generating output files
- Defining and controlling symbolic variables
- Interfacing with programs such as EDIT and BROWSE
- Controlling operational modes

The ability to perform all these tasks with simple ISPF service calls enables an application developer to concentrate on the overall purpose of the without excessive concern for how each elementary detail is actually carried out. This is one of the reasons why ISPF is such a good programming tool as well as a great productivity aid.

2.7 Basic concept of VSAM

Introduction

The data management services of an operating system help in the storing, cataloging, organizing, and retrieving of data from magnetic media. the different modes of such storage are organizations are :

- Physical sequential
- Partitioned
- Indexed sequential
- Direct

Different access methods are used in the storage and retrieval of information from these storage organizations. Examples of such access methods are:

- Basic sequential access methods(BSAM)
- Queued sequential access method (QSAM)
- Partitioned access methods(PAM)
- Basic direct access method(BDAM)
- Queued indexed sequential access method(QSAM)

There are other highly evolved and sophisticated organizations which are managed by access services methods(AMS) of MVS/XA, MVS/ESA, VSE/ESA. These access methods used to manipulate information in these organizations is called the virtual storage access method or, as its most commonly known ,VSAM.

VSAM is the IBM's latest and most advanced access method. It makes use of the virtual storage of MVS and VSE, hence the name *virtual storage access method.* VSAM is an high performance access method used in MVS, MVS/XA, MVS/ESA/VSE/ESA operating systems. VSAM software resides in virtual storage along with the program that needs its services for the manipulation of data on the *direct access storage device(DASD).*

Access Method Services

Access method services (AMS) is a program that helps you to allocate, maintain, and delete catalogs and data sets. It currently consists of one utility program called IDCAMS. This utility program is used in a job step just like other utility program of OS/VS.

IDCAMS is a multipurpose utility that can be used for the following functions:

- Allocating, maintaining and deleting catalogs.
- Allocating, maintaining and deleting VSAM data sets.
- Reorganizing and printing data sets.
- Cataloging non-VSAM data sets and generation data groups.

Advantages and Drawbacks compared to other Access methods

1. VSAM is superior to other access methods in the following respects.

2. The retrieval of records is faster because of an efficiently organized index . The index is small because of key compression algorithm used to store and retrieve its records.

3. Imbedded free space makes the insertion of records easy, and data sets therefore requires less organization. however , when this feature is used , data sets require more disk space.

4. The deletion of VSAM records, unlike that in ISAM, means that are physically deleted, thus allowing the reclaiming of the free space within the data set.

5. Records are accessed randomly by key or by address and can also be accessed sequentially at the same time.

6. VSAM data sets can be shared concurrently by partitions, regions, address space, and systems. The type and level of sharing can be controlled through AMS and JCL.

7. In MVS Access Methods Services commands can be executed as *time sharing option* (TSO) commands.

8. In *customer information control system* (CICS) , the VSAM pool of buffers, control block, channel programs can be shared by many VSAM data sets.

9. JCL for VSAM data sets are much simpler than for the other file structures .

10. VSAM provides data security through password protection of data set at different levels such as read and write.

11. VSAM provides the ability to physically distribute data sets over various volumes based on key ranges.

IDCAMS is a multipurpose utility that can be used for the following functions:

- Allocating, maintaining and deleting catalogs.
- Allocating, maintaining and deleting VSAM data sets.
- Reorganizing and printing data sets.
- Cataloging non-VSAM data sets and generation data groups.

Advantages and Drawbacks compared to other Access methods

1. VSAM is superior to other access methods in the following respects.

2. The retrieval of records is faster because of an efficiently organized index . The index is small because of key compression algorithm used to store and retrieve its records.

3. Imbedded free space makes the insertion of records easy, and data sets therefore requires less organization. however , when this feature is used , data sets require more disk space.

4. The deletion of VSAM records, unlike that in ISAM, means that are physically deleted, thus allowing the reclaiming of the free space within the data set.

5. Records are accessed randomly by key or by address and can also be accessed sequentially at the same time.

6. VSAM data sets can be shared concurrently by partitions, regions, address space, and systems. The type and level of sharing can be controlled through AMS and JCL.

7. In MVS Access Methods Services commands can be executed as *time sharing option* (TSO) commands.

8. In *customer information control system (* CICS) , the VSAM pool of buffers, control block, channel programs can be shared by many VSAM data sets.

9. JCL for VSAM data sets are much simpler than for the other file structures .

10. VSAM provides data security through password protection of data set at different levels such as read and write.

11. VSAM provides the ability to physically distribute data sets over various volumes based on key ranges.

12. VSAM catalogs and data sets are portable between operating systems (MVS TO VSE/ESA and viceversa).

13. VSAM data sets are device independent.

Some of the major drawbacks of VSAM

1. To take advantage of the partial self re-organization capabilities of VSAM data sets , free space must deliberately be left. This results in increased disk space requirements However, free space is only left for data sets requiring record adds deletes and changes. For data sets that are used as read only purpose, no free spaces are left.

2. Except for read only data sets, the integrity of VSAM data sets in cross system and cross region sharing must be controlled by the user. Data integrity must be a prime consideration in the initial design of application that will be shared across systems or regions.

Operating systems supporting VSAM

IBM provides three major operating systems for its mainframes. VSAM is supported by all three. However, there are significant differences in the implementation of VSAM .

DOS/VS : VSAM is supported on DOS/VS and DOS/VS(E) , and VSE/ESA systems. VSE/VSAM is much more powerful than its DOS/VS versions and has simplified JCL statements.

OS/VS: VSAM is supported on OS/VSI, MVS, MVS/XA and MVS/ESA operating systems. An enhanced version of VSAM, called *data facility product* (DFP), is now been used at many MVS installations.

VM/ESA : In a VM environment , CMS/VSAM support is based on VSE/VSAM

Types of VSAM data sets

VSAM data sets may have one of three possible types of organizations. they are :

1. *Key sequenced data set (KSDS)*
2. *Entry sequenced data set (ESDS)*
3. *Relative record data sets(RRDS)*
4. *Linear data set (LDS)*

2.8 COBOL

COBOL is the most wide spread commercial applications language in use today. The name COBOL is an abbreviation for Common Business oriented Language. As a business oriented language COBOL is designed specifically for commercial applications that can operate on a large volume of data. Although commercial languages such as COBOL are best suited for processing large volumes of data, they are less suitable for handling scientific problems where complex calculations are required.

COBOL is a common programming language, meaning that COBOL compilers are available for most computers. The same COBOL can be compiled and run on a variety of different machines, such as an IBM ES/9000 and a VAX-6410, or a PC with only minor variations.

The universality of COBOL allows computer users greater flexibility than they would have with many other languages. A company is free to acquire different brands of computers while using a single programming language. Similarly, conversion from one model computer to a more advanced or newer one presents no great problem as long as there is a COBOL compiler for each model. Since the language is so widely used, computers of future generation will undoubtedly support COBOL.

In summary, the meaning of the name COBOL suggests two of its basic advantages. It is common to most computers, and is business oriented. There are additional reasons why it is such a popular language. COBOL is an English-like language. All instructions can be coded using English words rather than complex codes. Because users are frequently able to understand English-like instructions of COBOL, it is considered as a user-friendly language; this means that its not overly technical like other languages. Users who rely on computer output but have no computer expertise may be able to understand the logic and instructions in a COBOL program.

COBOL was developed in 1959 by a group called CODASYL Committee. This committee included representatives from academia, user groups, and computer manufactures. The ultimate objective of this committee was to develop a standard business-oriented language for which all major manufactures would provide compilers.

In 1968, the first American National Standards(ANS) version of COBOL was developed and approved. Beginning in 1968, all major computer manufactures and software suppliers provided compilers that adhered to COBOL language formats specified in this ANS version of COBOL. In 1974, a second version of ANS COBOL was developed to make the language more efficient and standardized. The 1985 version of ANS COBOL is now widely used; it goes beyond the pervious

versions in increasing the versatility and the structure of the language. The next version of COBOL will be produced during this decade and is currently referred to as COBOL 9x.

System study and design

Chapter 3
System study & Design

3.1 System Study

Introduction

A successful software undergoes a software development life cycle (SDLC) consist of the Phases: System study, System design, Testing, Implementation and Maintenance. During the above phases, the team members in the project may come across various problems. Query Management System (QMS) is a tool to assist them in the communication of these problems. QMS record and track problems and their solutions. It ensures essential key data is recorded and validated for each problem raised.

In QMS by default each problem is known as a **query**. This is simply the term used for the recording of the problem and its associated details.

When a team member discovers a problem during the testing of a piece of software, he then need to communicate the details of the problem to the team leader. This is done by means of raising a query. The team leader investigate the query and may provide the solution for it, or direct the query to a person who is in a better position to answer it. QMS facilitate these functions.

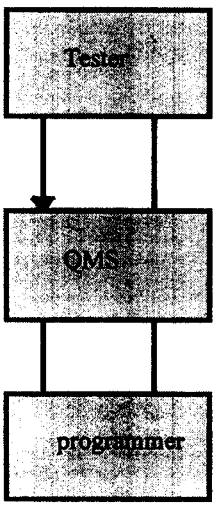
In a big organisation like WIPRO SYSTEMS, where a number of project will be going on at the same time, members in various projects are allowed to raise the queries in their respective projects. An employee can have access right to more than one project. For this QMS maintains records of access rights.

Query life cycle

Once a query is recorded in the QMS it passes through following stages before completion.

- Team member records the query in QMS
- Problem is assigned to team leader or another team member
- Team leader responds to the problem with a solution
- Solution is implemented and tested
- The team member signs off the problem

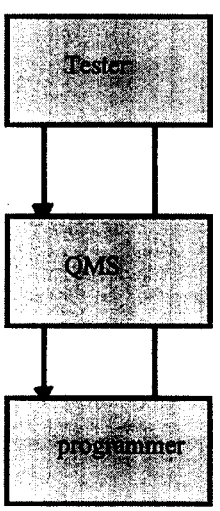
Following diagrams explain the main processes in Query Management System



Tester finds a problem

The problem is recorded in QMS

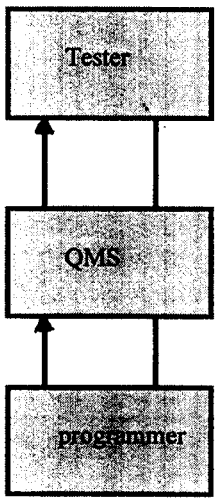
Figure 1.



Tester Assigns the problem to a programmer

QMS send notifiers to the programmer

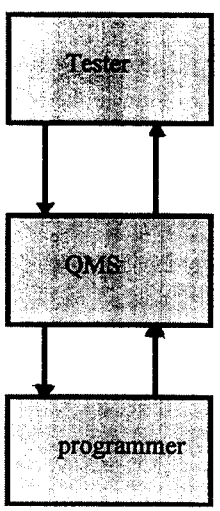
Figure 2.



QMS send notifier to tester

Programmer reads the problem and records the solution in QMS

Figure 3.



Tester implements the solution and fix the problem

Programmer and tester communicate between each other and finally problem is signed off.

Figure 4.

Main functionalities

QMS performs the following main functions

Raise query

When a programmer finds a problem. He can record his problem in QMS. QMS should allow the programmer to enter all the details of the query viz. raised by, date of raising, severity and priority of the query, brief description and an unformatted description of the query. Priority is to indicate the urgency of the problem. Severity is to indicate the business impact of the problem. Brief description indicate the development stage of the project and name of the module. Detailed description can range from few lines to several pages.

Update query

Query can be updated only by the raiser of the query. But after it is assigned, even the raiser can not update the query.

Assign/reassign the query

After a query is raised, the raiser can assign the query to any of the team members. After assigning the query the raiser will not be able to modify the query. If a query is assigned to a person, he can answer the query or it can be reassigned to some other person. A query is allowed to be forwarded to members in the same project. QMS must keep forward-to information and date of forwarding.

A raiser can not forward query to himself. Again a query can not be assigned to a person more than once.

Answer a query

QMS should provide a facility to answer raised queries. Answer and all the related information viz. answered-by, date of answering, and effort should be recorded. Answer consist of two parts, a brief description and detailed description. Like problem details, answer details also can extend from a few lines to several pages.

Updating answer

Like query details answer also can be updated. The right for updating the answer rests with the answeree. Ones the answer is confirmed, it can not be modified.

Fixing a query

The query can be fixed by the raiser of the query or the answeree. This indicate that the solution is successfully implemented and tested.

Closing a query

After successfully implementing the solution, the query can be signed off. This is the final stage. Only the raiser of a query is permitted to close it.

Supporting functions

QMS should provide options for maintaining the information about the users and project. Only the users with supervisor privilege are allowed to access these information.

A wide range of reports regarding queries, users and project are to be produced

When key events like assigning a query or answering a query occurs, notifiers should be send to appropriate users.

3.2 System design

3.2.1 Panel design

In QMS Project ISPF panels are used to create user interface. Panels are also used for presenting various reports and tutorial. For the screen layout of these panels refer Appendix-A. This tool has on-line help facility built into it. Help can be invoked by Pressing the <PF1> key. The help screen relating to the displayed panel will be shown.

When a panel is displayed, entering the END command or Pressing the <PF3> will cancel the current display. The control will return to the previous menu. Similarly RETURN command (<PF4>) causes the immediate return to the primary option menu of the application. Jump function allows you to go directly to any valid option from the primary menu. To use the jump function, enter the option in the command field of any panel, Preceded by an equal sign and followed by a blank
For Example:
Command = = => =3.1
takes you directly to the first suboption of option 3 of the primary option menu.

3.2.2 File Design

Six VSAM KSDS files are used to store all the information for this project. In order to suit with the multi-user environment, all the files are created with share option (2,3). It gives write integrity.

Purpose of each file are explained below. To get a more detailed picture, please refer Appendix-B.

File Query-tab

This file is used to store information about query. Each query in the file can be identified using its project-id, release-no and query-no. So the primary key for this file is composed of these three fields.

A status field is used to store the current status of the query. User can not modify this field. Program check the content of this field to validate the user operations on query. For example, the user is allowed to modify the query only if the status field is 'O' . When an opened query is first forwarded, the status field must be changed from 'O' to 'F' .

Following are the possible values of the status field used in this program:

value	explanation
O	Query is open
F	Query is forwarded
W	Waiting for answering to complete
A	Query is answered
X	Query is fixed
C	Query is closed

Other fields and its explanation are given in the Appendix-B.

Querydtls-Tab

This file is used to store the details of the query and its answer. User is first allowed to enter the details into a scratch file. After getting all the details into this file, it is copied line-by-line to the Querydtls-tab. Each line will have a particular identifier. This key is composed of fields project-id, release-no, query-no, type and line-no. Type field contains values 'Q' or 'A' which indicate query and answer correspondingly.

Project-Tab

This file is used to store the project information. Fields in the project are Project-id, Release-no, Project-description and query-count. Field query-count is used to store number of existing queries in the project. When creating a new project this field is initialised to zero. Length of this field is four. This limit the number of queries in each project to a maximum of 9999.

User-tab

User details are stored in this file. User-id is the primary key for this file.

Accright-tab

This file is used to store the access rights of the users. User-id, Project-id and Release-no are the fields in this file.

Forward-tab

Purpose of this file is to store the forwarding information. When a query is assigned, Query-no, Project-id and release-no, assigned-to and date-of-forwarding are stored in this file.

3.2.3 Application design

The initial panel of QMS is used to accept the Project-id, and release number. User's TSO user-id will be taken from the system variables. Before going to the next panel user's access right is checked. Only a valid user will be able to proceed to the next panel, Main menu..

In QMS all the functions are grouped under four main options viz.; Functions, Reports Support and Help. These main options contain sub options. Purpose of each option is explained below:

Main option functions

Selecting this option will display a sub menu, which contains following options.

Option raise query

This option allows the user to raise a new query. Project id and release no of the project is displayed on the panel. User has to enter a brief description of the query, priority and severity. The first four letters of the description should be SPEC, CODE or MISC, which indicate the subject of the query.

SPEC - Specification
CODE - Coding
MISC - Miscellaneous

After checking the access rights of the user, he will be informed of the query number in which the query is raised. this number is automatically raised by the system. Maximum number of queries allowed in a project is 9999. Date-of-raising and raised-by information are taken from the system variables. After the validation process the user is allowed to enter query-details. For this purpose a scratch file is opened in output mode and the user is allowed to edit this file. After editing is over, the content of this file is copied to querydtls-tab. ISPF edit macro is called for editing the scratch file.

Assign query

This option is to assign or reassign a query. User has to enter query-number and the Assigned-to field. Assigned to should be TSO user-id. If the status of the query is 'O' only the raiser is allowed to forward this query. If it is 'F' all the users to whom the query is forwarded are permitted to reassign the query. Access rights of the assigned-to and duplicate forwarding are checked before a query is assigned.

When a query is successfully assigned, notifiers are send to the assigned user-id. This is possible by calling a CLIST program. TSO send command is used for this purpose.

Answer query

This option allows the user to answer an existing query. User has to enter query number and a brief description of the answer. Rights of the user and the status of the query is checked. If the query-status is 'F' and query-number, user-id entry is present in the Forward-tab, user is allowed to answer the query. I.e. answered-by and date-of-answering are stored in the query-tab. The status of the query is changed to 'w'. After performing these operations, the user is allowed to enter the answer details in the same manner by which query-details are entered . Answer details are also stored in the querydtls-tab.

After entering the answer-details, the user is provided with a screen where he can enter the confirmation of the answer and effort (time taken in answering the query). If the confirmation is 'yes', query-status will be changed to 'A' and effort is written to the query-tab.

Update query details

A query raiser can update his query using this option. Query number is accepted from the user and the corresponding query is read from the query-tab. If the query is raised by the user and the current query status is 'O' , then the user is allowed to update the query. All the query details of the particular query are read from the querydtls-tab and is written to a scratch file. This file is then called using the edit macros. After necessary modification by the user, it is then written back to the querydtls-tab.

Update Answer details

This option allows the user to update the query-answer. Here also, the query number is the input from the user. Query-status and answered-by field are checked. If the query is answered by the user and the present query status is 'W', user is allowed to update answer-details.

After updating the answer-details, the user is provided with a screen where he can enter the confirmation of the answer and effort (time taken in answering the query). If the confirmation is 'yes', query-status will be changed to 'A' and effort is written to the query-tab.

Fix a query

Using this option, user can fix a query by entering the query number. Before fixing the query, query-status and fields; raised-by and answered-by in the file query-tab are checked. If the query-status is 'A' and user-id of the fixer is matched with any of the above mentioned fields, the query is fixed. This is indicated by changing the query status to 'X'.

Close a query

This option allows the user to close a query. Only the raiser is allowed to close a query. So after accepting the query number from the user, query-status and raised by field are checked. If the query-status is 'x' and user-id is equal to raised-by , the query-status is changed to 'C'.

Browse query

This option enables the user to browse through all the details of the query including the forwarding information. Query-no is accepted from the user. Query-tab, Querydtls-tab and forward-tab are opened and information for the accepted query number is written to a sequential file (scratch-file). ISPF browse macro is called and this allows the user to scroll through the contents of the file.

List processing

This option list all the existing queries in the project. User can select the queries displayed in the list by entering search arguments. A screen will be provided, where the user can enter the criteria for displaying the queries. All the fields in the file query-tab will displayed. User can enter his argument for each field. There is no need to enter values for each and every field. Leaving a field free will display all the queries for that field. Symbol '*' can be used as a generic search argument. I.e. entering a value abc* will display all the field starting with letters 'abc'.

To implement the above mentioned process, All the queries in the current project are copied into an ISPF table. ISPF functions are used to display the selected rows.

Following line commands are possible when the list is displayed:

AS - To assign/Reassign a query

AN - To answer a query

BR - To browse query details

CQ - To close a query

FQ - To fix a query

UQ - To update query answer

UA - To update query answer

Main option report

This option will display a sub menu with following options:

1. Display/Print queries
2. Display/Print project entries
3. Display/Print User list

Before printing the above reports, user can specify the format of printing, like number of line in a page, line spacing, margin and number of copies.

Main option support

This option display a sub menu which contains six options. Only user with supervisory right can access these options.

Modify Project

Using this option projects can be created, deleted or updated. User have to enter the project-id and release-no. If the project exist, all its details are displayed on a panel. Otherwise blank fields will be displayed. Type of change, I.e. New, Update or Delete is accepted from the user and corresponding operations are performed.

If the type of change is Delete, query-tab, querydtls-tab, forward-tab and accright-tab are opened and all the rows having the same project-id and release-no are deleted from the files.

Update user

This option allows the super user to maintain the user information. Entries in the file can be updated, deleted or new entries can be created. Deleting a user will delete the access rights of that user form the file accright-tab.

Set access rights

Access rights of the user can be set using this option. Project-id, release-no and user-id are the inputs. After validation of the fields, they are written to accright-tab. Before writing to the file duplicity of the entry is checked.

List project

Like option 'List processing', this option allows the super user to list all the project and the number of existing queries in each project. There is option to display selected rows. When the list is displayed, line commands DL-delete or UP-update can be performed.

List users

In this option, like other listing options, user can enter search arguments for displaying a user list. ISPF table functions are used to display the list. Line commands UP-update and DL-delete are allowed.

List access rights

Selecting this option will display a list of access rights according to the users choice. He can mention project or user-id. DL-delete is the only line command available with this option.

System Implementation and testing

Chapter 4
System implementation and Testing

4.1 System implementation

The completion of the design process initiate the next step of development , implementation and testing. After the design is completed the VSAM files are created first. All the files are created with share option (3,4). ISPF panels are created next. Two panels Libraries are used in this program

Self023.qms.panels contains all select and display panel

self023.qms.help Contains all the tutorial and help panels

Panel naming conventions used in this program is explained below:

Each panel has a panel id of length eight, starts with the letter 'Q'. The last letter is used to indicate the type of the panel. 'P' indicate a selection or display panel. 'H' or 'T' indicate that the panel is of type help. The other letters indicate the option , it belongs to. For example; panel id 'Q121000P' will be displayed when the user select the first option in the submenu 'Reports' (second option in the main menu). If more than one panel is to be displayed in an option, the second last character is used to indicate the order of display.

For example:

Q131001P is the first panel in the option =1.3.1;

Q131002P is the second panel in the option =1.3.1

ISPF message definitions are stored in the library self023.qms.msgs and they are grouped in three members, QMSM10, QMSM11 and QMSM13.

Other libraries used in this program are:

Self023.qms.loadlib - Application language library

Self023.qms.clist - command library

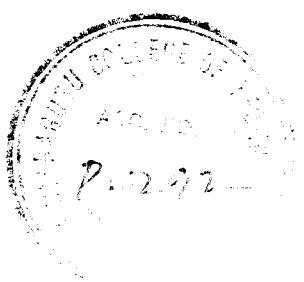
self023.qms.tables - Table input and output library

4.2 System testing

The application is tested with sample data before implementing it at the user site. In the testing process the main issues taken into consideration are

- Data validation and security.
- User friendliness
- Process validation
- Data integrity
- Preparing the feedback information on the testing process

After completing the testing process, the application is implemented at the user site and the users are given training on how to use the system. If, after testing there are some changes found to be made then they are incorporated into the application before implementation



Conclusion and scope for future expansion

Chapter 5
Conclusion & Scope for future Enhancement

5.1 Conclusion

The system has been developed and the objectives were achieved with test and real data. The Query Management System is an On-line system. The system provides near total solution to the Problems faced in the recording of the queries.

The system is currently under implementation in the user Environment. The system has undergone in-house testing with the help of random and real data. To conclude the system is highly flexible and User-friendly.

5.5 Future Enhancements

The package is developed with all the current requirements being incorporated into it . In future the system can be enhanced to provide more functionalities when required. It can be improved by providing statistical reports like bar charts and Pie charts. The changes can be made easily because of the flexible structure used in the design of the package.

Benefits

Chapter 6

Benefits

As a project trainee at Wipro Systems , I have benefited a lot. I have been provided with many facilities during the time of my project. The major facilities are:

- Unrestricted work-time. I could use the resources round the clock. No restrictions of working hours were imposed for my project work.

- In-house Training - Being a trainee , I could utilise the unique skills of the experienced employees at Wipro.

- Access to Library books and Software.

- Access to World Wide Web - This enabled me to have access to lot of information available on the WWW and learn new skills and additional knowledge on the state-of-the-art technology.

Above all I had experienced a great time being, a trainee in an IT firm which is currently one of leading provider of software for managing the information

These facilities helped me understand and have sound knowledge of state-of-the-art technology. Being in Wipro I have gained a lot of skills and information. I certainly believe that this experience will help me lift my career.

Summary

chapter 7

Summary

The project has been a great experience for me. During my project work I learnt the ES/9000 concepts and technology. The project work helped me to learn Operating System MVS/ESA, Operating environment TSO, and also languages like CLIST, JCL, VS COBOL, and ISPF/PDF. I was also able to learn the VSAM concepts. The guidance given by the technical professionals at Wipro helped me to learn and understand the concepts easily and apply them at times of need. I believe that all that I have learnt at Wipro will definitely help me to built a strong career in IT industry.

Appendix - A

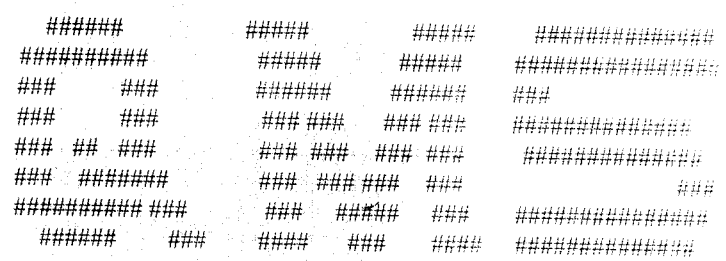
Screens

QUERY MANAGEMENT SYSTEM

DATE : 97/04/30
TIME : 10: 45
USER ID: SELF023

Enter Project id and Release no below

PROJECT ID : GADA
RELEASE NO : 3.33



WIPRO SYSTEMS
BANGALORE

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

DATE : 97/04/30
TIME : 11: 47
USER ID: SELF02

MAIN MENU

OPTIONS = = =>

1. FUNCTIONS	To get the sub menu functions
2. REPORTS	To get the sub menu reports
3. SUPPORTS	To get the sub menu support
4. EXIT	To exit from the application
5. HELP	To get the tutorials

Press < PF3 > to return

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

SUPPORT

OPTION===>

1. MODIFY PROJECT ENTRIES
2. MODIFY USER ENTRIES
3. SET ACCESS RIGHTS
4. LIST PROJECTS
5. LIST USERS
6. LIST ACCESS RIGHTS

Press <f3> to return to Previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

DATE :
TIME :

REPORTS

OPTION===>

1. DISPLAY/PRINT QUERIES
2. DISPLAY/PRINT PROJECT
3. DISPLAY/PRINT USER LIST

Press <PF3> to return to previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

DATE : 97/04/28
TIME : 12:50
USER ID: SELF023

FUNCTIONS

OPTION===>

1. RAISE QUERY	Raise a new query
2. ASSIGN/REASSIGN QUERY	For assign/reassign a query
3. ANSWER A QUERY	To answer an existing query
4. UPDATE QUERY DETAILS	For updating an existing query
5: UPDATE ANSWER DETAILS	For updating query answer
6. FIX A QUERY	For fixing a query
7. CLOSE A QUERY	For closing a query
8. CHANGE PROJECT	To change the existing project
9. LIST PROCESSING	To list queries
10. BROWSE QUERY	To browse query details

Press <Pf3> to return

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

RAISE A NEW QUERY

COMMAND===>

PROJECT ID : GADA DATE : 97/ 04/23

RELEASE NO : 3.33 RAISED BY : SELF023

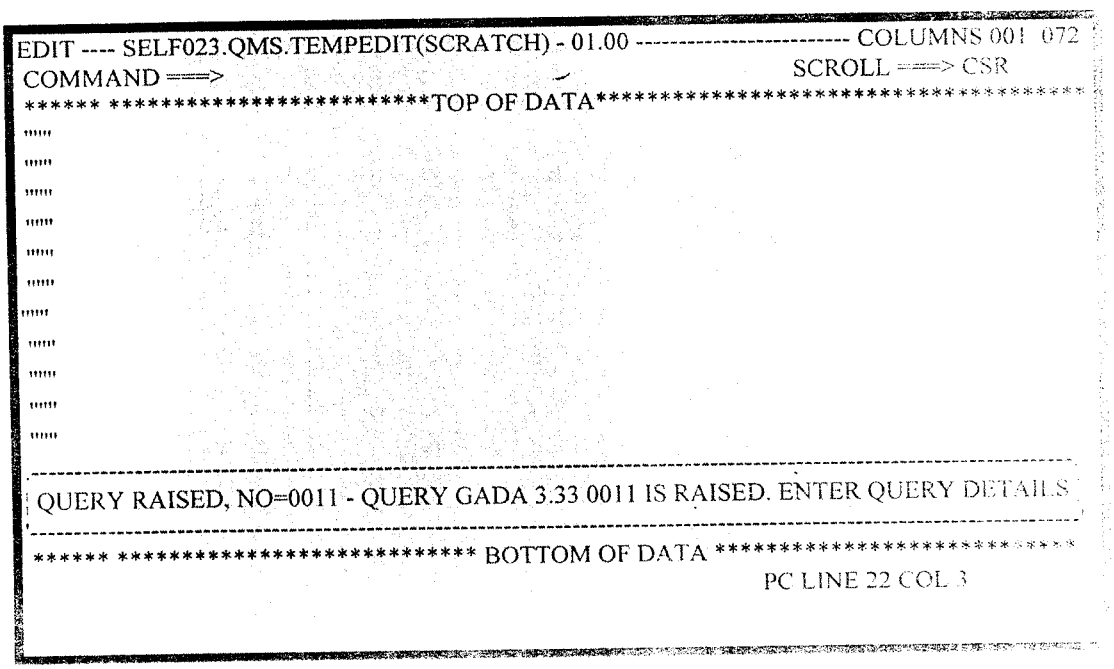
Please enter the following fields

QUERY DESCRIPTION : SPEC PROC CV1011 First four letters should be SPEC or CODE or MISC

SEVERITY : 1 Numeric 1:High 3:Low

PRIORITY : 2 Numeric 1:High 3:Low

Press <ENTER> to raise the query
press <PF3> to return to previous menu


EDIT ---- SELF023.QMS.TEMPEDIT(SCRATCH) - 01.00 ---------- COLUMNS 001 072
COMMAND ===>
SCROLL ===> CSR
****** **********TOP OF DATA**********
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
QUERY RAISED, NO=0011 - QUERY GADA 3.33 0011 IS RAISED. ENTER QUERY DETAILS
****** ********** BOTTOM OF DATA **********
PC LINE 22 COL 3

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

ASSIGN/REASSIGN A QUERY

COMMAND===>

PROJECT ID : GADA DATE : 97/04/29

RELEASE NO : 3.33 USER ID: SELF023

Please enter the following fields

QUERY NUMBER : 0231 Numeric value

FORWARD TO : SELF023 User id

Press <ENTER> to Assign query
Press <PF3> to return to the previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

ANSWER A QUERY

COMMAND===>

PROJECT ID : GADA DATE : 97/ 05/01

RELEASE NO : 3.33 ANSWERED BY: SELF025

Please enter the following field

QUERY NO : 0025 (Numeric)

ANSWER DESCRIPTION: USE COPY BOOK NO CPY-0034

Press <ENTER> to raise the query
press <PF3> to return to previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

UPDATE ANSWER DETAILS

COMMAND===>

PROJECT ID : GADA DATE : 97/05/04

RELEASE NO : 3.33 USER ID :SELF023

Please enter the following field

QUERY NUMBER : 1342 Numeric value

Press <ENTER> to continue
press <PF3> to return to the previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

CONFIRM ANSWER

COMMAND===>

PROJECT ID : GADA DATE : 97/05/05

RELEASE NO : 3.33 USER ID: SELF023

QUERY NUMBER :

Please enter the following fields

EFFORT : 00:35 HH:MM Format

CONFIRM ANSWER : YES Enter YES or NO

Press <ENTER> to confirm answer
press <PF3> to return to the previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

FIX A QUERY

COMMAND===>

PROJECT ID : GADA DATE : 97/04/30

RELEASE NO : 3.33 FIXED BY : SELF023

Please enter the following field

QUERY NO : 0456 Numeric

Press <ENTER> to FIX the query
press <PF3> to return to previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

CHANGE PROJECT

COMMAND===>

PROJECT ID : GADA

DATE : 97/04/29

RELEASE NO : 3.33

USER ID : SELF023

PROJECT DETAILS : GALAXY DATA CONVERSION

Please enter the following fields

PROJECT ID : SECD

RELEASE NO : 3.11

Press <ENTER> to change project
Press <PF3> to return to previous menu

WIPRO SYSTEMS QUERY MANAGEMENT SYSTEM

LIST PROCESSING

COMMAND===>

PROJECT ID : DADA
RELEASE NO : 3.33

DATE : 97/04/25
USER ID : SELF023

Please enter the criteria for displaying queries BLANK for selecting all
Use * as generic search argument

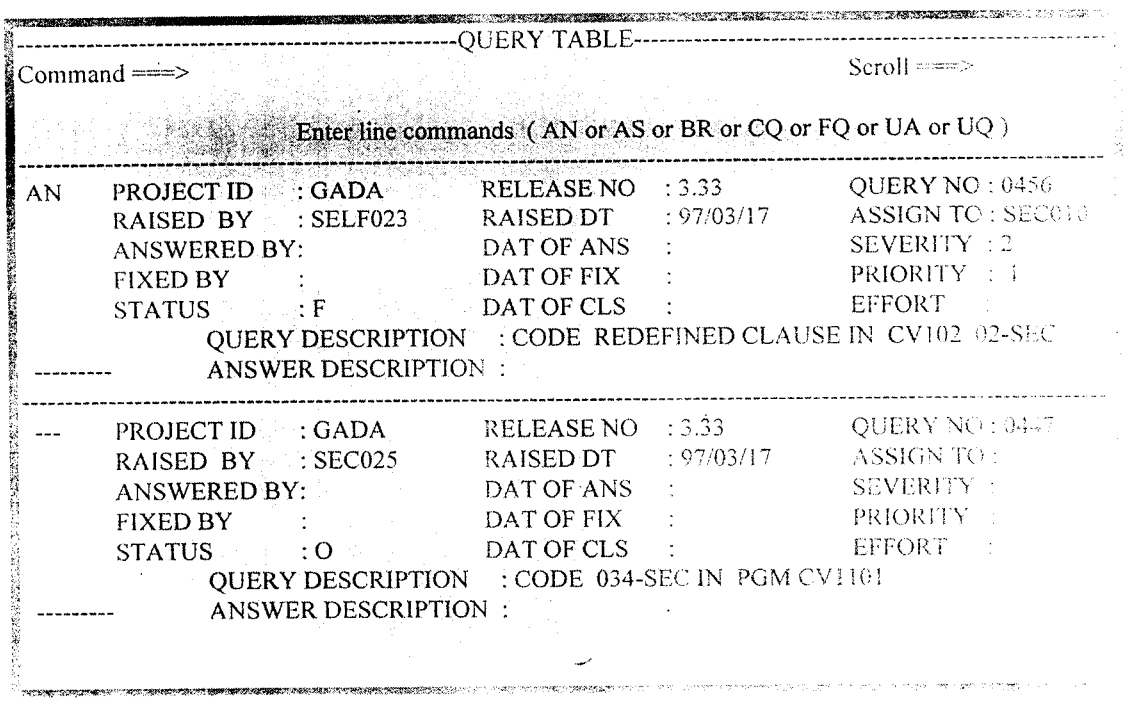
QUERY DES : SPEC*
RAISED BY : SELF023
ASSIGN TO :
PRIORITY :
FIXED BY :

SPEC or CODE or MISC
ANSWERED BY :
SEVERITY :
STATUS :

DATE OF RAISING	FROM : 97/03/01	TO	:	YY/MM/DD format
DATE OF ANSWERING	FROM :	TO	:	YY/MM/DD format
DATE OF FIXING	FROM :	TO	:	YY/MM/DD format
DATE OF CLOSING	FROM :	TO	:	YY/MM/DD format

Following line commands are available when the list is displayed

UQ Update query	AS Assign query	CQ Close query	BR Browse query
UA Update Answer	AN answer query	FQ Fix query	



----------QUERY TABLE----------

Command ===> Scroll ===>

Enter line commands (AN or AS or BR or CQ or FQ or UA or UQ)

AN PROJECT ID : GADA RELEASE NO : 3.33 QUERY NO : 0456
RAISED BY : SELF023 RAISED DT : 97/03/17 ASSIGN TO : SEC010
ANSWERED BY: DAT OF ANS : SEVERITY : 2
FIXED BY : DAT OF FIX : PRIORITY : 1
STATUS : F DAT OF CLS : EFFORT :
QUERY DESCRIPTION : CODE REDEFINED CLAUSE IN CV102 02-SEC
--------- ANSWER DESCRIPTION :

--- PROJECT ID : GADA RELEASE NO : 3.33 QUERY NO : 0457
RAISED BY : SEC025 RAISED DT : 97/03/17 ASSIGN TO :
ANSWERED BY: DAT OF ANS : SEVERITY :
FIXED BY : DAT OF FIX : PRIORITY :
STATUS : O DAT OF CLS : EFFORT :
QUERY DESCRIPTION : CODE 034-SEC IN PGM CV1101
--------- ANSWER DESCRIPTION :

WIPRO SYSTEMS

MODIFY USER ENTRIES

COMMAND===>

ENTER USERID BELOW:

USER ID : (First three letters should be character)

Press ENTER to display user record
Enter <PF3> to return to previous menu

WIPRO SYSTEMS

MODIFY USER ENTRIES

COMMAND ===>

USER ID : SEC025

TYPE OF CHANGE ===> NEW (NEW, UPDATE, OR DELETE)

USER NAME	: R.K KOTHARI
USER ADDRESS	: BOX NO 234512 BOMBAY
TELEPHONE NO	: 0124567235
EXTENSION NO	: 0345
EMAIL ADDRESS	: RKKOTH@WIPSYS.SOFT.NET

Confirm modification : YES (Enter YES or NO)

Appendix-B

Files

Appendix -B

File Descriptions

FILE NAME : PROJ-TAB

DESCRIPTION:

PROJ-TAB is used to store information about projects

Organization is VSAM KSDS

Primary key is PROJECT-NAME

Alternative key : NONE

Number of fields = 4

Total record length = 42

Record type is FIXED

FIELD DESCRIPTION:

PROJECT-NAME		PROJECT-DETAILS	QUERY-COUNT
PROJECT-ID	RELEASE-NO		

FIELD NAME	FIELD LENGTH	TYPE
PROJECT-ID	4	ALPHA NUMERIC
RELEASE-NO	4	ALPHANUMERIC
PROJECT-DETAILS	30	ALPHA NUMERIC
QUERY-COUNT	4	NUMERIC

FILE NAME : USER-TAB

DESCRIPTION:

USER-TAB is used to store information about USERS

Organization is VSAM KSDS

Primary key is USER-ID

Alternative key NONE

Number of fields = 6

Total record length = 86

Record type is FIXED

FIELD DESCRIPTION:

USER-ID	USER-NAME	PHONE-NO	EXT-NO	USER-ADD	EMAIL-ADD

FIELD NAME	FIELD LENGTH	TYPE
USER-ID	7	ALPHA NUMERIC
USER-NAME	20	ALPHA NUMERIC
PHONE-NO	10	NUMERIC
EXT-NO	4	NUMERIC
USER-ADD	20	ALPHA NUMERIC
EMAIL-ADD	25	ALPHA NUMERIC

FILE NAME : FORWARD-TAB

DESCRIPTION:

FORWARD-TAB is used to store query forwarding information

Organization is VSAM KSDS

Primary key is FORWARD-KEY

Alternate key : NONE

Number of fields = 5

Total record length = 29

Record type is FIXED

FIELD DESCRIPTION:

FORWARD-KEY			FORWARD-TO	FORWARD-DATE
PROJECT-ID	RELEASE-NO	QUERY-NO		

FIELD NAME	FIELD LENGTH	TYPE
PROJECT-ID	4	ALPHA NUMERIC
RELEASE-NO	4	ALPHA NUMERIC
QUERY-NO	4	NUMERIC
FORWARD-TO	7	ALPHA NUMERIC
FORWARD-DATE	10	ALPHA NUMERIC

FILE NAME : ACCRIGHT-TAB

DESCRIPTION:

ACCRIGHT-TAB is used to store the access right informations

Organization is VSAM KSDS

Primary key is ACCRIGHT-KEY

Alternate key : NONE

Number of fields = 3

Total record length = 15

Record type is FIXED

FIELD DESCRIPTION:

ACCRIGHT-KEY		
USER-ID	PROJECT-ID	RELEASE-NO

FIELD NAME	FIELD LENGTH	TYPE
USER-ID	7	ALPHA NUMERIC
PROJECT-ID	4	ALPHA NUMERIC
RELEASE-NO	4	ALPHA NUMERIC

FILE NAME : QUERY-TAB

DESCRIPTION:

QUERY-TAB is used to store query informations

Organization is VSAM KSDS

Primary key is QUERY-KEY

Alternate key : NONE

Number of fields = 12

Total record length = 138

Record type is FIXED

FIELD DESCRIPTION:

QUERY-KEY			RAISED-BY	DATE-OF-RAISING	ASSIGNED-TO	ANSWERED-BY
PROJECT-ID	RELEASE-NO	QUERY-NO				

DATE-OF-ANSWERING	FIXED-BY	DATA-OF-FIXING	DATE-OF-CLOSING	SEVERITY	STATUS	PRIORITY

EFFORT	QUERY-DES	ANSWER-DES

FIELD NAME	FIELD LENGTH	TYPE
PROJECT-ID	4	ALPHA NUMERIC
RELEASE-NO	4	ALPHA NUMERIC
QUERY-NO	4	NUMERIC
RAISED-BY	7	ALPHA NUMERIC
DATE -OF-RAISING	10	ALPHA NUMERIC
ASSIGNED-TO	7	ALPHA NUMERIC
ANSWERED-BY	7	ALPHA NUMERIC

DATE-OF-ANSWERING	10	ALPHA NUMERIC
FIXED-BY	7	ALPHA NUMERIC
DATE-OF-FIXING	10	ALPHA NUMERIC
DATE-OF-CLOSING	10	ALPHA NUMERIC
SEVERITY	1	NUMERIC
QRY-STATUS	1	CHARACTER
PRIORITY	1	NUMERIC
EFFORT	5	ALPHA NUMERIC
QUERY-DES	25	ALPHA NUMERIC
ANSWER-DES	25	ALPHA NUMERIC

FILE NAME : QUERYDTLS-TAB

DESCRIPTION:

QUERYDTLS-TAB is used to store query details and answer details

Organization is VSAM KSDS

Primary key is QUERYDTLS-KEY

Alternate key : NONE

Number of fields = 6

Total record length = 97

Record type is FIXED

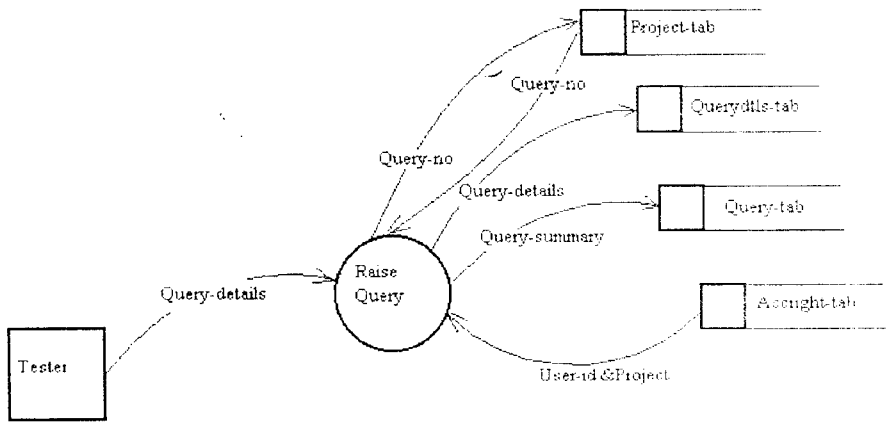
FIELD DESCRIPTION:

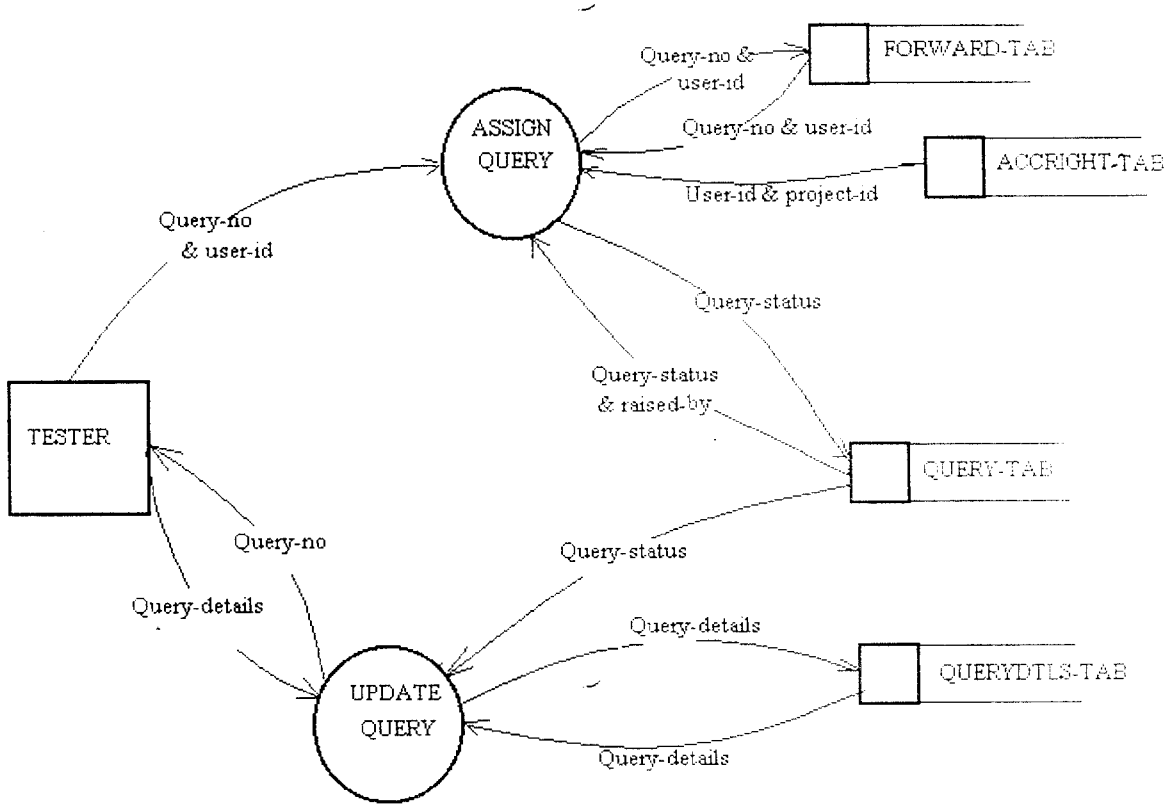
QUERYDTLS-KEY					QUERY-DETAILS
PROJECT-ID	RELEASE-NO	QUERY-NO	QUERY-TYPE	LINE-COUNT	

FIELD NAME	FIELD LENGTH	TYPE
PROJECT-ID	4	ALPHA NUMERIC
RELEASE-NO	4	ALPHA NUMERIC
QUERY-NO	4	NUMERIC
QUERY TYPE	1	CHARACTER
LINE-COUNT	4	NUMERIC
QUERY-DETAILS	80	ALPHA NUMERIC

Appendix - C

Data flow diagrams


Project-tab
Query-no
Querydtls-tab
Query-no
Query-details
Query-tab
Query-summary
Raise
Query
Query-details
Tester
User-id &Project


FORWARD-TAB
Query-no &
user-id
ASSIGN
QUERY
Query-no & user-id
ACCRIGHT-TAB
User-id & project-id
Query-no
& user-id
Query-status
Query-status
& raised-by
TESTER
QUERY-TAB
Query-no
Query-status
Query-details
Query-details
QUERYDTLS-TAB
UPDATE
QUERY
Query-details

Appendix -D

Reports

BIBLIOGRAPHY

IBM MANUALS

IBM TSO Extension Version 2
CLISTS

ISPF Dialog Management Guide & Reference
reference SC34-4266-1;

ISPF Dialog Management Examples;
reference SC34-4313-0;

ISPF/PDF Guide & Reference ;
reference SC34-4258-1;

ISPF/PDF Edit and Edit Macros;
reference SC34-4253-1;

ISPF/PDF Library Management Facility;
reference SC34-4260-1;

VSAM

VSAM Concepts, Programming and Design
Jay Ranade , Hirday Ranade Macgraw-Hill Inc.

VS COBOL

Structured COBOL Programming (Seventh Edition)
Nancy Stern & Robert A. Stern. John Wiley &
Sons Inc.

System Analysis And Design

Introduction to System Analysis and Design
Lee, Galgotia publications, 1982

System Analysis and Design
Elis .M. Awad, Galgotia Publications, 1989.