

Utilidades MVS (I)

Entradilla:

Este articulo comienza un breve tema dedicado a unas herramientas que por su sencillez y potencia permiten realizar gran numero de funciones a nivel del sistema. Son las típicas utilidades del sistema operativo.

1.- Introducción

Evidentemente un tema tan amplio como el de las utilidades requiere mucho mas que un articulo para poder describir cada una de ellas, razón por la cual solo se han elegido aquellas que pueden ser mas representativas dentro de cada época, y que a su vez son de gran UTILIDAD para cualquier desarrollador de un Centro de Desarrollo/Proceso de Datos (CPD)

Con el fin de enmarcar históricamente el origen de las utilidades, se incluye a continuación una definición del termino SISTEMA OPERATIVO, ya que las utilidades no son mas que programas incluidos o que acompañan a éste. No tienen una función determinada dentro de cualquier aplicativo, pero en cambio son usadas en la preparación y ejecución de cualquier trabajo para editar, listar, borrar, etc., cualquier fichero.

En cualquier caso, hay que tener en cuenta que los sistemas operativos surgieron acompañando al Hardware para el que se escribieron con el fin de hacer mas manejables las maquinas. No obstante, cabe recordar que el S.O. es el soporte lógico del hardware, y en consecuencia, como todo informatico conoce bien, un mismo hardware puede soportar distintos S.O.

Definición: Sistema operativo es un conjunto de programas realizados, generalmente por el fabricante del equipo y cuya misión es aumentar el rendimiento total del equipo de proceso de datos.

Este rendimiento total se ve influido por los siguientes factores:

- Rendimiento específico: volumen total de trabajo realizado por el sistema en un determinado periodo de tiempo (**THROUGHPUT**)
- Tiempo de respuesta: intervalo de tiempo que transcurre desde que un usuario somete un elemento de trabajo al ordenador, y el momento en el que recibe los resultados.
- Disponibilidad: Es el grado en que un sistema de proceso de datos esta preparado cuando se necesita.

2.- Utilidades

Bajo este calificativo, se encuentran aquellos programas que están realizados o no por el fabricante del S.O. y que tienen por misión liberar al usuario de codificar rutinas y códigos para la realización de funciones de uso común, tales como:

- mantener y manipular datos del sistema
- reorganizar, cambiar, comparar o manipular datos y/o ficheros de usuario

Las utilidades no se diferencian en nada de cualquier programa de aplicación, a no ser por su finalidad, y pueden ejecutarse como cualquier otro, bien mediante:

- un procedimiento de comandos o CLIST.
- a través de una trabajo batch preparando un JCL.
- Desde cualquier aplicación mediante una llamada al ejecutable correspondiente.

Para la exposición que sigue se va a seguir un JCL que agrupa todas las utilidades que se comentan en el articulo. No obstante, se verán ejemplos aislados con las otras formas de invocarlas.

Entorno de la Utilidad

Dentro de un JCL, los programas de utilidad necesitan para su control dos tipos de sentencias:

- Sentencias propias de JCL para la asignación de recursos; y
- Sentencias de control de la Utilidad para especificar a ésta lo que debe ejecutar.

Esquema de paso de JCL para ejecutar una UTILIDAD

```
//nom_paso EXEC PGM=nom_utilidad
//STEPLIB DD DSN=SYS1.LINKLIB,DISP=SHR
//SYSPRINT DD ...
//SYSIN DD *
 fichas de control de la utilidad
/*
//Ddname1 DD ...
//Ddname2 DD ...
//Ddnamen DD ...
```

Figura 1

La **figura 1** muestra el formato general de cualquier paso de JCL que ejecute un programa de utilidad:

Donde la Ddname **SYSPRINT** define el destino en el que se escribirán los mensajes devueltos por la utilidad; y

La Ddname **SYSIN** contiene las fichas de control de la utilidad, es decir, contiene los comandos que debe ejecutar la utilidad.

El formato de las fichas de control es el siguiente:

COMANDO operadores comentario

debiendo resaltarse que cada elemento se encuentra delimitado por blancos, y los distintos operadores separados por comas sin dejar espacios. En el caso de que los distintos parámetros u operadores no quepan en una única línea, deben terminar en coma, dejar al menos un blanco, y escribir un carácter distinto de blanco en la columna 72, considerándose el contenido de la siguiente línea como continuación de los operadores de la anterior.

Así mismo, es de reseñar que todas las utilidades del MVS tienen como característica común el hecho de ubicarse en una librería de uso general, cuyo nombre suele ser **SYS1.LINKLIB**. Por esta razón debe especificarse la ficha DD que tiene por etiqueta (Ddname) **STEPLIB** para poder incorporar al entorno (alocar) el ejecutable correspondiente.

3.- IEFBR14

Esta utilidad tiene por particularidad que su fuente solo contiene como instrucción ejecutable: 'FIN DE PROGRAMA'. Es decir, en sí mismo no hace nada, pero su ejecución influye en los ficheros asociados en base al parámetro **DISP** que acompaña a cada una de las fichas DD asociadas a dicho paso.

Como ya conoce el lector, este parámetro especifica mediante 3 subparámetros, la disposición del fichero al comenzar el paso, y la disposición en la que debe quedar si termina bien, y si termina mal, tal y como muestra la **figura 2**.

| Formato del parametro DISP de la ficha DD | |
|---|---|
| | DISP=(anterior,bien,mal) |
| donde | |
| ANTERIOR puede tomar los valores: | |
| NEW | si no existía antes |
| OLD | si ya existía antes. En este caso los nuevos datos sustituyen a los existentes. |
| MOD | Ya existe, y los nuevos datos se apendizaran a los existentes. |
| SHR | Es un fichero compartido. |
| BIEN y MAL pueden tomar los valores: | |
| DELETE | si se desea que se borre. |
| KEEP | que se mantenga. |
| CATLG | que se incluya en el catalogo. |

Figura 2

Los nombres dados a las fichas DD (de ahí el nombre de Ddname) no son significativos en si mismos, incluso pueden concatenarse especificando una única Ddname.

| Ejemplo de la Utilidad: IEFBR14 | |
|---------------------------------|---------------------------------|
| //***** | |
| //* | BORRAR SALIDA |
| //* | ----- |
| //EJEMPLO3 | EXEC PGM=IEFBR14 |
| //STEPLIB | DD DSN=SYS1.LINKLIB,DISP=SHR |
| //SORTIN | DD DSN=&USER..ESPECIAL.SORT, |
| // | DISP=(MOD,DELETE,DELETE), |
| // | UNIT=(SYSDA,3),DATACLAS=DFBXP, |
| // | LRECL=40,SPACE=(CYL,(4,1),RLSE) |

Figura 3

La **figura 3** muestra un ejemplo de uso de esta utilidad. Como ya se verá al comentar el JCL completo que acompaña al artículo, el siguiente paso al mostrado, tiene definido el fichero de salida como nuevo para cuando comience el paso, por eso en este paso, si existe el fichero cuyo DSN se especifica en la ficha DD de nombre **SORTIN**, lo borra, pues el parámetro **DISP=(MOD,DELETE,DELETE)** especifica que cuando comience la ejecución de este paso, si no existe el fichero, debe crearle, razón por la que se tienen que especificar los parámetros necesarios para crear el fichero; y si ya existía, los datos de salida (que no hay, pues el programa solo ejecuta un fin de programa) se apendizarían al fichero por tener situación **MOD**. Cuando termina la ejecución del programa, elige la segunda situación si termina bien, y la tercera si termina mal. Como no hace nada, debe terminar bien, y precisamente se especifica que le borre con el

parámetro **DELETE**; y, si termina mal, se especifica que le borre también pues se ha especificado el parámetro **DELETE** en la tercera posición, por lo que en cualquier caso, cuando aloque (asigne al entorno del programa), el fichero, es cierto que no existe, y por tanto debe tener disposición **NEW**.

4.-IEBGENER

Este programa de utilidad, junto con el anterior, son las utilidades estándar mas usadas en cualquier instalación.

Básicamente, este programa se limita a copiar el fichero de entrada definido con la Ddname **SYSUT1** sobre otro de salida definido con la Ddname **SYSUT2**, pudiendo incluso manipular la información contenida en los mismos.

Ejemplo de la Utilidad: IEBGENER

```
//EJEMPLO4 EXEC PGM=IEBGENER
//STEPLIB DD DSN=SYS1.LINKLIB,DISP=SHR
//SYSPRINT DD DUMMY
//* ----- fichero de entrada
//SYSUT1 DD DISP=SHR,DSN=JMPDES.ESPECIAL.REP4
//* ----- fichero de salida
//SYSUT2 DD SYSOUT=A,HOLD=YES,DEST=RMT103
//SYSIN DD DUMMY
```

Figura 4

El ejemplo mostrado en la **figura 4** realiza la copia de un fichero de entrada sobre la impresora que tiene por nombre lógico RMT103 ya que es el destino dado al fichero de salida.

Cuando la Ddname **SYSIN** es **DUMMY** (fantasma) los parámetros **LRECL** y **RECFM** de los ficheros asociados a las Ddnames **SYSUT1** y de **SYSUT2** deben ser iguales, pues eso significa que se realiza la copia sin manipular la información.

Por el contrario, cuando se desea manipular la información, debe especificarse dentro del fichero asociado a la DDname **SYSIN** los comandos de la utilidad especificados en la **figura 5**:

Comandos y operadores para la utilidad: IEBGENER

GENERATE Nombre del comando.
Especifica que los registros de salida son diferentes a los de entrada

MAXFLDS Operación de **GENERATE**.
Especifica el numero máximo de elementos **field** del comando **record**

RECORD Nombre del comando.
Especifica el formato de los registros de salida

FIELD =(parm_1,parm_2,parm_3,parm_4)
siendo
parm_1: tamaño del área de registro de salida
parm_2: posición del primer byte seleccionado
parm_3: posible conversión
parm_4: posición del primer octeto del área de salida.

Especifica el formato de los distintos campos de salida.
Ejemplo:(30,1,,51) especifica que las 30 primeras posiciones del registro de entrada debe colocarlas a partir de la posición 51 en el registro de salida

Figura 5

Cuando se desea que el formato del registro de salida (cuando el fichero es nuevo), coincida con el de la entrada, basta con codificar como parámetro **DCB** (Definition Control Block) de **SYSUT2** :

DCB=*.SYSUT1

En el caso de que los datos de **SYSUT1** vengan especificados en la propia cadena (IN STREAM), debe especificarse obligatoriamente la **DCB** de **SYSUT2**.

Cuando la salida se direcciona a la salida del JOB, no es necesario especificar **DCB**, ya que asume la estándar.

SYSUT2 DD SYSOUT=*

Otro ejemplo de esta utilidad seria el de la **figura 6**, mediante el cual, se desea copiar las posiciones 1-30 de entrada a las posiciones 51-80 del registro del fichero de salida, y las posiciones 51-100 a las posiciones 1-50 del registro de salida.

```
//EJEMPLO5 EXEC PGM=IEBGNER
//SYSPRINT DD DUMMY
//STEPLIB DD DSN=SYS1.LINKLIB,DISP=SHR
//SYSUT1 DD DSN=JMPDES.ENTRADA,DISP=SHR
//SYSUT2 DD DSN=JMPDES.SALIDA,DISP=(NEW,CATLG,DELETE),
// UNIT=SYSDA,
// SPACE=(CYL,(40,10),RLSE),
// DCB=BLKSIZE=23440,LRECL=80,RECFM=FB)
//SYSIN DD *
GENERATE MAXFLDS=2
RECORD FIELD=(30,1,,51),FIELD=(50,51,,1)
/*
```

Figura 6

5.-IEHLIST

Esta utility es usada para listar nombres de ficheros.

Como ya se comentó en el primer artículo del tema dedicado a los ficheros de MVS (Solo Programadores-num.19) La VTOC (Volume Table of Contents) es equiparable a la FAT (File Allocation Table) del MS-DOS, y contiene la dirección física dentro del volumen (disco) de todos los ficheros contenidos en el mismo.

Por su parte, el catálogo, es un fichero que contiene el nombre de los ficheros, y el nombre del volumen en el que se encuentra.

De esta forma, combinando ambas búsquedas puede localizarse cualquier fichero o dataset de la instalación.

También, a modo de recordatorio, se comenta que los ficheros particionados o PDS (Partitioned Data Set), se componen de dos partes, el **directorio**, y el **área de datos**, estando referenciado en la VTOC el área de directorio. Por su parte, este área contienen las referencias a cada uno de los miembros incluidos en el área de datos del fichero PDS.

Los comandos admitidos por esta utilidad a través de la SYSIN son:

- **LISTVTOC** : Para listar la VTOC
- **LISTPDS** : Listar el directorio.
- **LISTCTLG** : Listar el catálogo,

Como particularidad, se puede citar el hecho de que este programa para poder ejecutarse, debe contener fichas DD para alocar (incorporar al entorno) el Volumen que se referenciará en el comando a ejecutar en la SYSIN.

Ejemplo de la utilidad: IEHLIST

```
//EJEMPLO6 EXEC PGM=IEHLIST
//STEPLIB DD DSN=SYS1.LINKLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//DISCO1 DD UNIT=SYSDA,VOL=SER=PACK11,DISP=SHR
//DISCO2 DD UNIT=SYSDA,VOL=SER=PACK12,DISP=SHR
//DISCO3 DD UNIT=2314,VOL=SER=PACK13,DISP=SHR
//SYSIN DD *
LISTVTOC VOL=SYSDA=PACK11 (1)
LISTVTOC VOL=SYSDA=PACK12,FORMAT,DSN=(JMPDES) (2)
LISTVTOC VOL=2314=PACK13,DUMP (3)
LISTPDS VOL=SYSDA=PACK11,DSN=JMPDES.PRE1 (4)
LISTPDS VOL=SYSDA=PACK12,DSN=JMPDES.PRE2,FORMAT (5)
LISTCTLG VOL=SYSDA=PACK11 (6)
LISTCTLG VOL=SYSDA=PACK12,NODE=JMPDES (7)
/*
```

Figura 7

La **figura 7** contiene un ejemplo de esta utilidad.

A la vista de dicha figura, y teniendo en cuenta que los distintos listados se direccionan a la cola de salida A, los comandos que ejecuta este programa, son:

- (1) Lista todas las entradas del VTOC del volumen alocado con disco1.
- (2) lista todas las entradas que cuelgan del prefijo (puntero) JMPDES, en formato editado.
- (3) Lista la VTOC del volumen alocado con la Ddname disco3 en formato hexadecimal.
- (4) Lista el directorio o relación de miembros del fichero PDS cuyo DSN se especifica.
- (5) Idem pero en formato editado.

(6) Lista el catalogo que reside en el volumen alocado con la Ddname Discol.

(7) Lista del catalogo que reside el volumen alocado con la Ddname disco2, las entradas asociadas al cualificador (puntero) JMPDES.

6.- Próximo articulo:

Puesto que el espacio no perdona, se aplaza para el próximo mes las utilidades **IEBCOPY**, **IEBPTPCH** e **IEHPROGM**, así como otros dos programas de suma importancia en cualquier gran instalación, los programas **IDCAMS** e **IKJEFT01** o programa TSO para ejecutar en batch.

7.- Utilidad

La utilidad que acompaña este mes al articulo tiene por finalidad listar los **objetos natural** cuyo nombre figura en un fichero que sirve de entrada y que se denomina con la variable **&ENTRADA**.

A su vez, este fichero de entrada, se ve incrementado con los objetos referenciados en los objetos listados.

A ser posible, el próximo mes se comentará con mas detenimiento esta utilidad, así como las modificaciones introducidas en los programas de la utilidad que acompañaron al tema anterior para adaptarles a estas nuevas necesidades.

Como curiosidad, este JCL muestra cómo se puede ejecutar de forma iterativa un JCL, pues el ultimo paso del mismo vuelve a enviar a la cola de entrada (submite) el mismo JCL.