SISINF

SUBPROGRAMAS Y LLAMAR

Versión 7.3

Kratos,S.A. de C.V.

La Tecnología en Software.

CONTENIDO

PREFACIO

En **SISINF** algunos módulos e instrucciones son opcionales de la instalación ya que no todos los usuarios necesitan de dichos módulos o instrucciones.

La opción de Subprogramas permite aprovechar las ventajas de **SISINF** para manejo de información y acceso a la base de datos, desde un programa hecho en otro lenguaje si así lo requiere la aplicación.

1) Introducción.

Dado que el lenguaje **SISINF** está orientado a la solución de problemas administrativos, cuando una aplicación requiera de hacer cálculos en matrices, evaluar polinomios, usar funciones trigonométricas, etc., será necesario que se use la instrucción de LLAMAR o la opción de SUBPROGRAMAS.

Usando la instrucción de LLAMAR el programa en **SISINF**, está estructurado en la siguiente forma:

a) Explotar el Banco de Datos y formar con la información necesaria un archivo temporal.

b) Por la instrucción de LLAMAR pasar el control a un programa en otro lenguaje, el cual lee el archivo temporal y efectúa los cálculos necesarios. Los resultados se dejan en uno o varios temporales.

c) Leer o los archivos temporales creados por el programa llamado y generar un reporte o afectar el Banco de Datos.

La expresión otro lenguaje se refiere a los lenguajes FORTRAN, C o C++, los cuales están disponibles en las diferentes computadoras dependiendo del proveedor.

Usando la opción de SUBPROGRAMAS en **SISINF**, el problema se resuelve de la siguiente forma:

a) El programa principal está escrito en otro lenguaje (FORTRAN, C o C++).

b) El programa principal es el que tiene el control y hace los cálculos.

c) El programa principal llama a un SUBPROGRAMA hecho en **SISINF**, para leer o escribir la información en el Banco de Datos.

d) La forma de llamar al subprograma, dependerá del lenguaje. Así, para FORTRAN se usará CALL.

Algunas consideraciones para cuál de las dos formas usar son las siguientes:

- La instrucción de LLAMAR está disponible en la mayoría de las computadoras donde está **SISINF**.

- Los SUBPROGRAMAS son opcionales en la instalación.

- Por restricciones propias de la implantación de cada lenguaje en las diferentes computadoras, NO en todas (las computadoras) y en todos (los lenguajes) está disponible la opción de SUBPROGRAMAS.

- En forma general lo que se puede hacer con LLAMAR se puede hacer en SUBPROGRAMAS, no así lo contrario.

- En forma general es más fácil trabajar con SUBPROGRAMAS, ya que con la instrucción de LLAMAR se deberá hacer uso de instrucciones para abrir, cerrar, leer y escribir los archivos temporales, los cuales son específicos de cada sistema operativo.

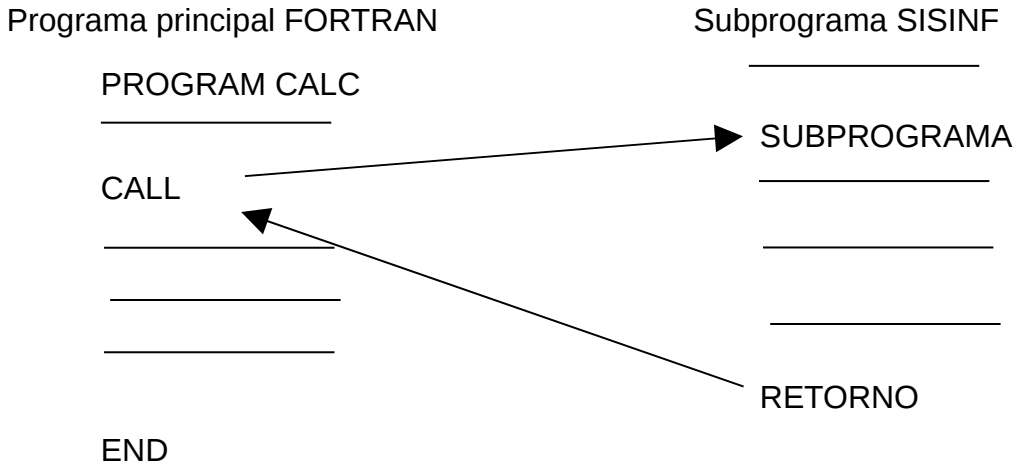
En este manual se describe el uso de los SUBPROGRAMAS lo cual es opcional en la instalación. Se usará FORTRAN como el lenguaje en que se hacen los programas, ya que su extensión a otros es sencilla. Hay convertidores de FORTRAN a C.

Se asume que el programador conoce el lenguaje FORTRAN y la forma de ejecutar programas en dicho lenguaje en la computadora que se está usando.

2) Funcionamiento general de Subprogramas.

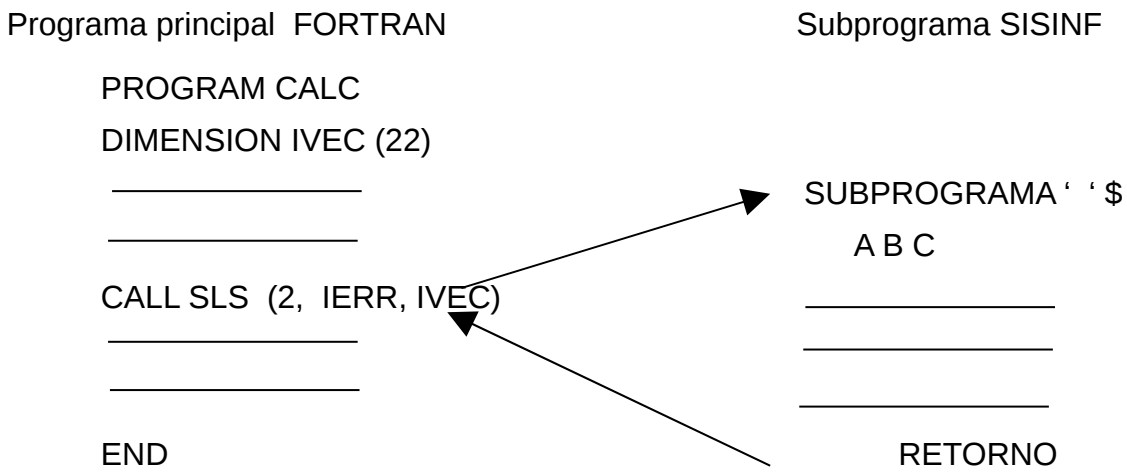
Se describirá a continuación el funcionamiento general de esta opción sin profundizar en ciertos requisitos que se verán posteriormente. Se usará FORTRAN como lenguaje.

En forma general los subprogramas de **SISINF** son parecidos a las subrutinas de FORTRAN, así los subprogramas de **SISINF** tiene como primera instrucción la de SUBPROGRAMA. Al ser llamados por el programa principal en FORTRAN, el control se transfiere al subprograma en **SISINF** y una vez que el subprograma ha hecho lo que se desea, se deberá poner una instrucción de RETORNO para regresar el control al programa principal. Gráficamente quedaría:



Cuando el programa principal llama al subprograma, le pasa la información que necesita. De la misma forma al terminar el subprograma y retornar al programa principal le regresa ciertos valores.

Para pasar la información en el programa principal de FORTRAN se deberá poner un vector que contenga las variables comunes a programa principal y subprograma, y en la instrucción de SUBPROGRAMA se deberá de poner una lista con los nombres de dichas variables. Gráficamente lo anterior queda así:

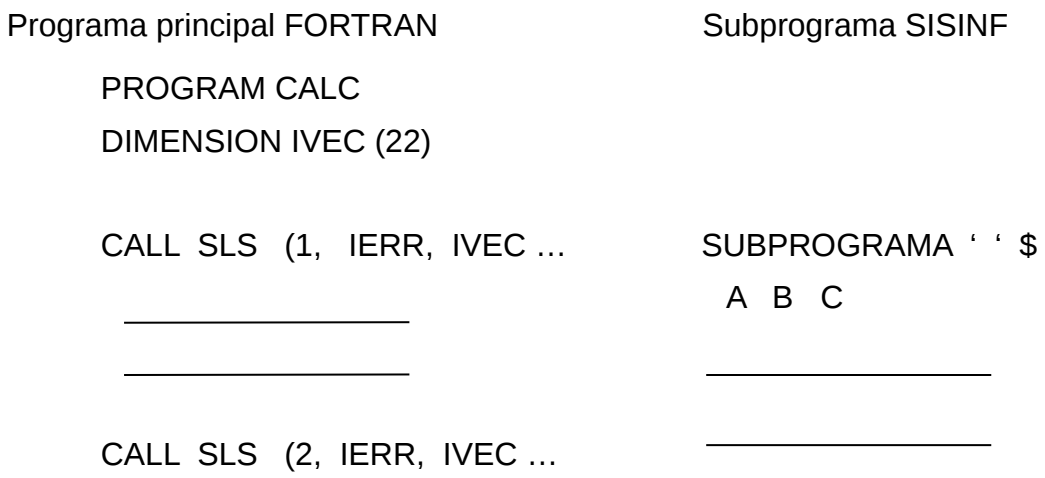


Al ejecutar el programa de FORTRAN y llegar a la instrucción CALL SLS se efectúan lo siguientes pasos:

- El contenido del vector IVEC se pasa a la variable A, B, C.
- Se ejecuta el subprograma el cual puede leer o modificar el contenido de las variables.
- Al ejecutarse la instrucción de RETORNO, se pasa el contenido de las variables al vector IVEC y se continúa la ejecución después del CALL SLS.

Una vez estudiada la forma en que se pasa la información, se verá cómo indicar que subprograma ejecutar.

El programa principal de FORTRAN deberá tener una llamada al inicio para indicar qué subprograma se va a usar y al final deberá poner otra llamada para indicarle que el subprograma ya no se va a usar. Gráficamente lo anterior queda como:



RETORNO

```
CALL  SLS  (3,  IERR,  IVEC …
END
```

Con la llamada CALL SLS (1, …) al inicio se indica en el vector IVEC qué subprograma se ejecuta, con la llamada de CALL SLS (2, …) se ejecuta el subprograma y con la llamada CALL SLS (3, …) al final se cierran archivos del subprograma y se termina. Las llamadas a CALL SLS (2, …) se pueden efectuar tantas como se deseen.

El programa principal en FORTRAN se hace como cualquier programa en este lenguaje, poniendo solamente las instrucciones de CALL indicadas. Una vez que se tiene, se compila en forma normal y cuando ya no tiene errores se carga pidiendo solamente que se use una biblioteca de **SISINF** con las instrucciones para efectuar los CALL SLS.

Aunque en esta descripción se usó FORTRAN como lenguaje, también es aplicable lo descrito aquí a C y otros lenguajes en que tengan instrucciones para llamar a subrutinas.

Por último, se verá cómo crear un subprograma de **SISINF**.

Para ello es necesario hacer:

- Con el editor del sistema operativo, teclear el subprograma en el archivo PSxxyy.
- Ejecutar el módulo CLS para generar el archivo ESxxyy cuando no hay errores.
- Si hay errores, quitarlos con el editor y ejecutar nuevamente CLS.

3) Estructura del Subprograma.

La estructura general de los subprogramas es como ya se describió, es decir, una primera instrucción de SUBPROGRAMA, instrucciones de **SISINF** y una o más instrucciones de RETORNO de acuerdo con la lógica. En esta sección se describen posibles estructuras.

Cuando el subprograma tiene una sola función, como puede ser leer un archivo, la única consideración especial es el caso de fin de archivo. Así por ejemplo:

```
        SUBPROGRAMA  'LEER SECUENCIAL ' INDF …
        BUSCAR  DISCO  … EJECUTA  10  SEC-G1
        _______________
        _______________
       _____________   _______________
                       _______________
        INDF = 0       _______________
        RETORNO
10   INDF  =  1
        RETORNO
```

En este caso cada llamada al subprograma se lee un registro, si no hay un fin de archivo INDF vale 0. Si hay fin de archivo INDF vale 1. El programa principal deberá consultar INDF y cuando vale uno pasar a otra etapa del proceso.

Cuando el subprograma sirve para varias funciones es conveniente que la primera variable sea un indicador que señale la operación deseada. Así por ejemplo, un subprograma puede quedar como:

```
          SUBPROGRAMA 'LISTAR' IND …
          SI IND = 1  EJECUTA 10
          SI IND = 2  EJECUTA 20
          SI IND = 3  EJECUTA 30
          RETORNO
10        _______________
```

```
                ________________
                ________________

                RETORNO
20              ________________
                ________________

                RETORNO
30              ________________

                RETORNO
```

En este ejemplo si IND = 1 se ejecutan las instrucciones en 10 y se retorna, si IND = 2 se ejecutan las instrucciones en 20 y se retorna, etc.

Con los ejemplos anteriores se da una idea de la estructura general de los subprogramas pero no se aclara qué instrucciones de **SISINF** se pueden usar. En un subprograma de **SISINF** se pueden usar **todas** las instrucciones del lenguaje **SISINF**, con excepción de PROGRAMA y FIN, las cuales son reemplazadas por las instrucciones de SUBPROGRAMA y RETORNO. Es decir, el subprograma de **SISINF** puede tener instrucciones de terminal, impresión, cálculo, archivos temporales y permanentes.

Dado que en el programa principal y en el subprograma se pueden tener instrucciones con iguales propósitos, los siguientes comentarios son aplicables:

- Al inicio con CALL SLS (1, …), se carga el subprograma y la pantalla se borra, apareciendo el formato usual de **SISINF**.

- Si en el programa principal y en el subprograma, se desea leer y escribir en la **terminal**, se pueden tener problemas con la posición del cursor.

- Si en el programa principal y en el subprograma se **imprime** información, dependerá del sistema operativo si se genera un solo listado o dos en forma independiente. Si se genera un solo listado, se pueden tener problemas con el brinco de la hoja.

4) Instrucción de Subprograma.

Esta instrucción es la primera de un subprograma y define el letrero que aparece como nombre del programa en la terminal, así como las variables que servirán como parámetros al ser llamado el subprograma.

El formato general es:

SUBPROGRAMA LET VAR VAR VAR …

Los siguientes comentarios son aplicables:

- Debe ser la primera instrucción de un subprograma:
- Debe no tener etiqueta.
- Sólo puede haber una instrucción de SUBPROGRAMA en un subprograma.
- Las variables VAR pueden ser de cualquier tipo, es decir N1, N2, N3, N4, F, X o S.
- Debe especificarse al menos una variable.

5) Instrucción de Retorno.

Esta instrucción indica que se regrese el control de la ejecución al programa llamador.

El formato general es:

RETORNO

Los siguientes comentarios son aplicables:

- Debe haber al menos una instrucción de RETORNO en el subprograma; si no la hay se asume uno al final.
- Puede haber varias instrucciones de RETORNO en un subprograma.
- El contenido de las variables especificadas en la instrucción de subprogramas se pasa al programa llamador.

6) Estructura del Programa que llama a un Subprograma.

En las descripciones anteriores se usó el término de programa principal para el programa que llama a un subprograma, en forma general el programa principal o cualquier subrutina puede ser el programa llamador. Se conservará el término programa principal.

El programa principal interactúa con los subprogramas por medio de la instrucción de:

CALL SLS (IOPT, INDE, IVEC, ILON, ISUB)

donde los parámetros contienen:

IOPT	1 al inicio 2 para ejecutar el subprograma 3 al final
INDE	Indicador de error (o = no hay error)
IVEC	Vector donde se tienen las variables a pasar al subprograma, así como el nombre del subprograma a ejecutar.
ILON	Longitud del vector IVEC. (Se debe poner la longitud total independientemente de IOPT).
ISUB	Número que identifica al subprograma. El parámetro de ISUB se discute más adelante. En esta descripción se deberá poner 1.

Antes de hacer uso del subprograma, es decir, al inicio del programa principal se deberá poner la siguiente:

CALL SLS (1, INDE, IVEC, ILON, 1)

En donde el contenido del IVEC es el siguiente:

IVEC (1)	Iniciales de DBD del subprograma.

IVEC (2)	Iniciales de CLS del subprograma.
IVEC (3) a IVEC (10)	Unidad de Disco (Directorio, Grupo y Cuenta).
IVEC (11) a IVEC (14)	Clave de acceso.
IVEC (15)	Código de error.
IVEC (16) a IVEC (20)	Para uso interno.
IVEC (21)	Variables de comunicación programa principal y subprograma.

El funcionamiento de CALL SLS (1, …) es el siguiente:

a) Con la información de IVEC (1) a IVEC (14) que es la misma que pide el módulo ELS al inicio, se carga el subprograma, si hay error en la carga se regresa en IVEC (15) un –1 y en la pantalla se despliega la causa.

b) Se borra la pantalla y aparece la pantalla estándar de ELS.

c) Se **inicializan** las variables del vector IVEC a los valores iniciales de ELS.

d) Si hay un error en la **clave de acceso** en IVEC (15) está el código de error, en cuyo caso en la pantalla se desplegó la causa.

e) Se regrese a la Siguiente Instrucción después de CALL SLS.

Después del CALL SLS se deberá revisar INDE y si hubo error, terminar.

Al final, cuando ya no se desea hacer uso del SUBPROGRAMA se deberá poner:

```
CALL  SLS  (3, INDE, IVEC, ILON, 1)
```

ES MUY IMPORTANTE ejecutar esta instrucción ya que con ella se cierran los archivos permanentes, temporales y de impresión que se usen.

Cuando se desee ejecutar el subprograma se deberá poner:

```
CALL  SLS  (2, INDE, IVEC, ILON, 1)
```

Es donde el contenido del vector IVEC es:

IVEC (1) a IVEC (14)	No se usan por esta instrucción ni se altera el

contenido.

IVEC (15)	Código de error, 0 no hay error.
IVEC (16) a IVEC (20)	para uso interno.
IVEC (21)	Variables de comunicación programa principal y subprogramas.

El funcionamiento del CALL SLS (2, …) es el siguiente:

a) La información en IVEC (21) en adelante se pasa a las variables especificadas en la instrucción de SUBPROGRAMA.

b) Se ejecuta en el subprograma la segunda instrucción.

c) Cuando se encuentre una instrucción de RETORNO se pasa el contenido de las variables especificadas en la instrucción de SUBPROGRAMA al vector IVEC (21) en adelante.

d) Si hay un error de cancelación se pone el número del error en IVEC (15) y se ejecuta internamente una instrucción de RETORNO. El error se despliega en la terminal como en la ejecución de ELS.

e) Se continúa la ejecución después de CALL SLS.

Es muy importante que se revise la variable INDE, si es cero se puede continuar, si no es cero hay error. Más adelante se dan los posibles errores.

En un programa principal se puede llamar CALL SLS (2, …) tantas veces como se necesite y en uno o varios lados de la codificación según la lógica.

La forma en que se almacena la información en IVEC (21) en adelante dependerá del modelo y marca de la computadora.

El siguiente ejemplo ilustra cómo se puede calcular IVEC para un subprograma.

- a) En DBD se definió:

```
NUM-CLT    *  *  R0  LL1  N1  O
NOM-CLT    *  *  R0  S  X  29
DIR-CLT    *  *  R0  S  X  40
FEC-FAC    *  *   R1  S  F
```

- b) Suponga que la instrucción de SUBPROGRAMA es:

SUBPROGRAMA 'CLIENTES' NUM-CLT NOM-CLT DIR-CLT FEC-FAC

- c) Si se consulta el Manual de Implantación, en el capítulo de la máquina que se usa y en la sección de subprogramas, se tendrá la información de cuántas palabras ocupan cada variable. Para este ejemplo si se supone que las variables N1 ocupan una palabra entera, N2 ocupa 2, N3 ocupa 3, N4 ocupa 4, F ocupa 2, S ocupan 1 y X ocupa el número de caracteres entre 2 redondeado hacia arriba.

- d) El cálculo de IVEC se resume en la siguiente tabla:

Nombre	Tipo	Long. en Pal.	IVEC
NUM-CLT	N1	1	21
NOM-CLT	X29	15	22
DIR-CLT	X40	20	37
FEC-FAC	F	2	57
			59

Es decir, NUM-CLT empieza en IVEC (21) y como ocupa 1 palabra NOM-CLT empieza en IVEC (22). Como NOM-CLT es de 29 caracteres, es decir, ocupa (29/2 redondeado hacia arriba) 15 palabras entonces DIR-CLT empieza en IVEC (37) (37 = 22 + 15).

- e) La longitud de IVEC es de 58 (59-1) y la equivalencia es:

NUM-CLT está en IVEC (21)

NOM-CLT está en IVEC (22) a IVEC (36)

DIR-CLT está en IVEC (37) a IVEC (56)

FEC-FAC está en IVEC (57) e IVEC (58)

Se recuerda nuevamente que el Manual de Implantación tiene un capítulo para cada máquina y una sección para los subprogramas. En esta sección se especifica la forma de calcular las posiciones en el vector IVEC, dependiendo de la forma en que se almacene internamente cada tipo de variable.

7) Ejemplo de Subprograma.

Se supone que en el archivo CLTS se tiene el campo CAN-ART (de N1 cero decimales) al cual se desea sacar el promedio. En AREA se tiene un campo llamado IND de N1 de 0.

El programa en FORTRAN sería:

```
                PROGRAMA PRUEBA
                DIMENSION IVEC (22), IDIS (8),ICU (4)
                EQUIVALENCE (IVEC (1), IDBD), IVEC (2), ICLS)
                *, (IVEC (3), IDIS), IVEC (11), ICU),
                *, (IVEC (21), IND), (IVEC (22), CANT)
                INTEGER CANT
                DATA IDND/2HEJ/, ICLS/2HAI/, IDIS/8*IH /
                *, ICU/4IH /
        C       LLAMADA INICIO
                CALL SLS (1, INDE, IVEC, 22, 1)
                IF (INDE. NE. 0) STOP
        C       ACUMULAR
        1       N = 0
                SACUM = 0.0
        2       CALL SLS (2, INDE, IVEC, 22, 1)
                IF (INDE.NE.0) STOP
                IF (IND.NE.0) GOTO 3
                N = N + 1
                SACUM = SACUM +CANT
                GO TO 2
    C       IMPRIMIR
            3       PROM = SACUM/N
                    WRITE (5,4) PROM
            4       FORMAT (‘ PROMEDIO = ‘F10.2)
            C       TERMINAR
```

```
            CALL SLS (3,INDE,IVEC,22,1)
            STOP
            END
```

El subprograma en **SISINF** está hecho de forma que si el valor de IND vale 1 se terminó el archivo. El subprograma se tecleó en PSEJAI.

```
            SUBPROGRAMA 'PROMEDIO' IND CAN-ART
            BUSCAR DISCO CLTS EJECUTA 10 SEC-G1
            LEER DISCO CLTS CAN-ART
            IND = 0
            RETORNO
10          IND = 1
            ESCRIBIR TERMINAL ' '
            RETORNO
```

La instrucción de ESCRIBIR TERMINAL está puesta para que no se tengan problemas con el WRITE.

8) Recomendaciones al usar Subprogramas.

La siguiente es una lista de recomendaciones al usar SUBPROGRAMAS.

- Revisar que esté el CALL SLS (3, …).
- Revisar siempre INDE y si no es cero, terminar.
- Si el programa principal usa vectores o matrices, se deberá tener cuidado de no tener valores fuera de dimensión ya que un valor fuera de dimensión puede modificar el programa principal, así como el subprograma con resultados IMPREDECIBLES que afecten el banco de datos, o la correcta ejecución de las instrucciones.
- Revisar la correspondencia entre IVEC y las variables del SUBPROGRAMA.
- Ver el Manual de Implantación de **SISINF** para el uso de esta opción en una máquina específica.
- Si se modifica el programa principal se deberá compilar y cargar nuevamente.
- Si se modifica el subprograma se deberá ejecutar el módulo CLS.

9) Varios Subprogramas.

En algunas ocasiones se desea que un mismo programa principal en FORTRAN, C o C++. interactúe al **mismo** tiempo con varios SUBPROGRAMAS de **SISINF**.

Entre las razones para hacerlo se pueden tener:

- Si se pusiera todo en un solo SUBPROGRAMA éste sería más complejo y más difícil de mantener.
- Cada SUBPROGRAMA pertenece a una DBD diferente.

Para ello se tiene el último parámetro en el CALL SLS. En esta forma se pondrá un 1 para el primer subprograma, un 2 para el segundo, etc.

Por razones de implantación y de tiempo de ejecución, se recomienda revisar en el Manual de Implantación de **SISINF** de la máquina que se usa, cuál es el límite máximo de subprogramas.

Un ejemplo del uso de 2 subprogramas sería:

```
      PROGRAM EJEM
      DIMENSION IVEC1 (50), IVEC2 (70)
C     INICIAR SUBPROGRAMAS
      CALL SLS (1, INDE, IVEC1, 50, 1)
      IF (INDE.NE.0) STOP
      CALL SLS (1, INDE, IVEC2, 70, 2)
      IF (INDE.NE.0) STOP
C     CALCULAR
      _________________
      _________________
      _________________
      CALL SLS (2,INDE,IVEC1,50,1)
      _________________
      _________________
      _________________
```

```
      CALL SLS (2,INDE,IVEC2,70,2)
      _____________
      _____________
      _____________
C     TERMINAR
      CALL SLS (3,INDE,IVEC1,50,1)
      IF (INDE.NE.0) STOP
      CALL SLS (3,INDE,IVEC2,70,2)
      IF (INDE.NE.0) STOP
      _____________
      _____________
      _____________
      _____________
      _____________
      _____________
      STOP
      END
```

Note cómo cada programa usa su propio vector IVEC.

10) Errores en Subprogramas.

Como se ha descrito en las secciones anteriores, después de llamar a SLS es necesario revisar INDE. Los valores que puede tener son:

0 No hay error, se puede continuar.

1 El subprograma no se puede cargar.

2 La longitud del vector IVEC no es la misma en el programa y en el subprograma. El módulo CLS al pedir la opción A indica cuántas palabras usa IVEC en el subprograma.

3 Error al llamar a SLS posiblemente no exista el módulo para ejecutar el subprograma o no esté definido para el sistemaConsulte el Manual de Implantación sobre la forma de hacerlo.

4 Error de comunicación. En IVEC (15) está el error.

5 Error al ejecutar el subprograma, en IVEC (15) está el código del error.

6 Parámetro inválido en la llamada de SLS (ejemplo opción no es 1, 2 o 3).

En los casos 1 y 5 ya fue desplegada en la terminal una descripción del error.

11) Inicio y terminación de Programas Llamados.

Los pasos que se efectúan en el módulo ELS al ejecutarse la instrucción de LLAMAR, así como los que realiza el programa llamado, se pueden resumir en los siguientes:

a) El módulo ELS pide la ejecución del programa llamado al Sistema Operativo.

b) El módulo ELS forma y envía un mensaje al programa llamado con la información suficiente para poder abrir el archivo temporal.

c) El programa llamado recibe el mensaje.

d) El programa llamado procesa el temporal.

e) El programa llamado forma y envía un mensaje a ELS para indicarle si hay error en el proceso.

f) El módulo ELS recibe el mensaje y si no hay error, continúa con la siguiente instrucción después de la de LLAMAR.

Dado que la forma de recibir y de enviar mensajes cambia dependiendo del Sistema Operativo de la máquina y para que el programador no necesite conocer el de una máquina específica, se tienen las siguientes subrutinas:

CTII iniciar la comunicación.

CTIR recibir un mensaje.

CTIE enviar un mensaje.

CTIT terminar la comunicación.

CTSE sacar código del error.

Con estas subrutinas la estructura del programa llamado quedaría:

a) Llamar a CTII para iniciar la comunicación.

b) Llamar a CTIR para recibir el mensaje de ELS con la información del temporal.

c) Procesar la información del temporal.

d) Enviar a ELS con CTIE el mensaje de no error o de error.

e) Terminar la comunicación con CTIT.

Los parámetros de estas subrutinas son:

```
CALL   CTII   (ARC, INDE, NOMI)
CALL CTIR   (ARC, INDE, BUF, LONGM, LONGR)
CALL CTIE   (ARC, INDE, BUF, LONG)
CALL CTIT   (ARC, INDE)
CALL CTSE  (ARC,IER1, IER2)
```

En donde:

ARC Vector entero cuya dimensión dependerá del modelo y marca de computadora. Usado como área de trabajo para poder usar estas subrutinas. Se deberá tener cuidado de NO modificar su contenido.

INDE Variable entera para indicador de error 0 = no hay, 1 = SI.

NOMI Vector entero donde está el nombre del programa llamado.

BUF Vector entero donde se tiene el mensaje.

LONGM Variable entera o constante con la longitud máxima que puede tener el mensaje que se recibe.

LONGR Variable entera donde se almacena la longitud del mensaje recibido.

LONG Variable entera o constante con la longitud de envío.

IER1

IER2 Variables enteras donde se almacena el código del error.

En el Manual de Implantación se deberá revisar en la sección de LLAMAR para conocer:

- La dimensión del vector ARC.
- La información que se tiene en el vector BUF donde se recibe el mensaje.
- Qué manual del Sistema Operativo consultar para investigar el significado

de IER1 y IER2.

El siguiente ejemplo supone que la dimensión de ARC es de 6 y la de BUF es de 20.

```
      PROGRAM EJEM
      DIMENSION ARC(6), NOMI(3), BUF (20)
      INTEGER ARC, BUF
      DATA NOMI/2HEJ, 2HEM, 2H /
C     INICIAR COMUNICACIÓN Y RECIBIR MENSAJE
      CALL CTII (ARC, INDE, NOMI)
      IF (INDE.NE.0) GO TO 100
      CALL CTIR (ARC, INDE, BUF, 20, LONGR)
      IF (INDE.NE.0) GO TO 100
      IF (LONGR.NE.20) GO TO 100
C     PROCESAR EL TEMPORAL
      _____
      _____
C     TERMINAR CON ERROR
                  90  BUF (1) = 0
      BUF (2) = IERR
      GO TO 92
C     TERMINAR CON NO ERROR
91    BUF (1) = 9999
      BUF (2) = IERR
C     ENVIAR MENSAJE
92    CALL CTIE (ARC, INDE, BUF, 2)
      IF (INDE.NE.0) GO TO 100
      CALL CTIT (ARC, INDE)
      IF (INDE.NE.0) GO TO 100
      STOP
C     ERROR DE COMUNICACIÓN
100   CALL CTSE (ARC, IER1, IER2)
      WRITE (101, 5) IER1, IER2
```

```
101   FORMAT (10X 2I 10)
      STOP
      END
```

Note que el mensaje del programa llamado a ELS es de 2 palabras, en la primera se deberá poner 9999 si no hay error, o cualquier número diferente a 9999 (para el ejemplo se puso 0) cuando sí hay error. En la segunda se deberá de poner el código del error en caso de ocurrir. Dicho código será el que despliega el módulo ELS.

12) Rutinas de Conversión de Números.

El lenguaje FORTRAN maneja la información numérica en lo que se llama aritmética entera, aritmética real y aritmética de doble precisión. El rango válido de números que se pueden manejar dependerá del tipo de aritmética y de la marca de computadora.

El lenguaje **SISINF** maneja variables numéricas N1, N2, N3 y N4 con el mismo rango en todas las máquinas pero con diferente forma interna de manejo así en una máquina una variable N2 puede ocupar 2 palabras y en otra 1.

Cuando e hace un programa en FORTRAN que interactúe con un programa de **SISINF** es necesario conocer:

- Rangos de números en las diferentes aritméticas de FORTRAN.
- Representación interna de las variables N1, N2, N3 y N4.
- Número de palabras que ocupa una variable N1, N2, N3 y N4.

El primer punto se puede aclarar casi siempre en el Manual de FORTRAN de la computadora que se use, los dos siguientes se discuten en el Manual de Implantación en la sección de la máquina que se use.

Teniendo la información anterior se puede escoger el tipo de variables que se usarán en el programa de FORTRAN, siendo un caso muy común que las variables N2, N3 y N4 no se pueden manejar en aritmética entera por falta de capacidad y que sea necesario usar aritmética real o de doble precisión.

Para evitar que el programador haga la lógica de conversión entre números de **SISINF** y números reales de FORTRAN, se tienen 4 subrutinas con los siguientes nombres de función:

PASR Pasa de **SISINF** a Real.

PARS Pasa de Real a **SISINF**.

PASD Pasa de **SISINF** a doble precisión.

PADS Pasa de doble precisión a **SISINF**.

Con los siguientes parámetros:

```
CALL PASR (N,ITIPO,IDEC,R)
CALL PARS (R,N,ITIPO,IDEC)
```

```
CALL  PASD (N,ITIPO,IDEC,D)
CALL  PADS (D,N,ITIPO,IDEC)
```

En donde:

N Vector entero donde está el número en **SISINF**.

ITIPO Variable entera o constante entera con el tipo del número (1=N1, 2=N2, 3=N3, 4=N4)

IDEC Variable entera o constante entera con los decimales del número en **SISINF**.

R Variable Real donde se tiene el número a convertir o convertido.

D Variable de doble precisión donde se tiene el número a convertir o convertido.

Con el uso de estas subrutinas el programador sólo necesita saber cuántas palabras ocupan cada una de sus variables en la computadora que esté usando y no necesita conocer su representación interna.

Un ejemplo del uso de estas subrutinas aplicadas a programas llamados sería:

a) En DBD se definió:

```
AREA
N10   *     *     R0    S     N1    0
N22   *     *     R0    S     N2    2
N34   *     *     R0    X     N3    4
```

b) En un programa de **SISINF** que se escriba el temporal 950 con N10, N22 y N34 se tendría:

```
_____________________        Se definen N10, N22 y N34
_____________________
_____________________
ESCRIBIR TEMPORAL 950 $
                N10  N22  N34
_____________________
_____________________
```

_____

LLAMAR ‘EJEM ’ 950

_____

_____

_____

c) Para hacer el programa de FORTRAN se supone que:

- Las variables N22 y N34 se manejarán en real y doble precisión, respectivamente.
- Las variables N2 ocupan 2 palabras y N3 ocupan 3 palabras.

Entonces se tiene:

```
      PROGRAMA EJEM
      DIMENSION IBUF (6)
      REAL N22
      DOUBLE PRECISION N34
      _____________
      _____________  Se abre el temporal 960 y se lee un registro en IBUF.
      _____________
      _____________
C     IBUF (1) ES N1
C     IBUF  (2)  -  (3)  ES   N22
C     IBUF  (4)  -  (6)  ES  N34
      N1 = IBUF(1)
      CALL PASR (IBUF(2),2,2,N22)
      CALL PASD (IBUF(4),3,4,N34)
```

En una situación similar el programador deberá hacer lo siguiente:

- Consultar el manual de FORTRAN para determinar el tipo de aritmética a usar.
- Consultar el Manual de Implantación para saber el número de palabras que ocupa cada variable.

13) Residencia de las Subrutinas.

Para hacer programas llamados, conversión de números con las subrutinas descritas en las secciones previas, al igual que el CALL SLS de los subprogramas, se encuentran en una biblioteca de subrutinas llamada SSVXX (Subrutinas **SISINF** vXX) que se provee si se tiene la opción de subprogramas.

En el Manual de Implantación se describen las instrucciones que se necesitan especificar al cargador del sistema operativo que se use, para poder usarlas en cada ambiente.