1. Módulos de Funciones

1.1 Introducción

Los módulos de funciones son objetos que realizan operaciones que pueden ser utilizadas en varios programas. Al crear un módulo de función con el código que realiza una operación, se evita tener que repetirlo en todos los programas que realicen esa operación añadiendo en ellos una llamada al módulo de función. Además de evitar que se repita el mismo código en diferentes programas, se facilita el mantenimiento del proceso, ya que las modificaciones que se realicen en un módulo de función afectan a todos los programas que lo utilicen.

SAP dispone de un gran número de módulos de función predefinidos que se pueden utilizar en nuestros programas, a las que se añadirán los que se desarrollen a medida.

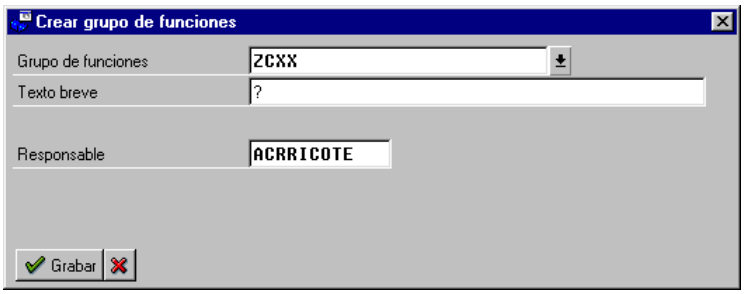
Los módulos de función pertenecen a grupos de funciones, que los agrupan según su funcionalidad. Los módulos de funciones de un mismo grupo de funciones comparten las definiciones de datos globales.

1.2 Creación de un grupo de funciones.

Para crear un grupo de función iremos por la opción de menú del OBJET NAVIGATOR
Ruta de acceso: (En el menú principal de SAP) 'Herramientas→ Workbench ABAP4→ Resumen → Object Navigator (SE80).

Seleccionaremos 'Grupo de funciones' , pondremos el nombre del grupo de funciones a Crear
Ej 'ZCXX'.

Aparecerá una ventana el la que definir los atributos del grupo de funciones.

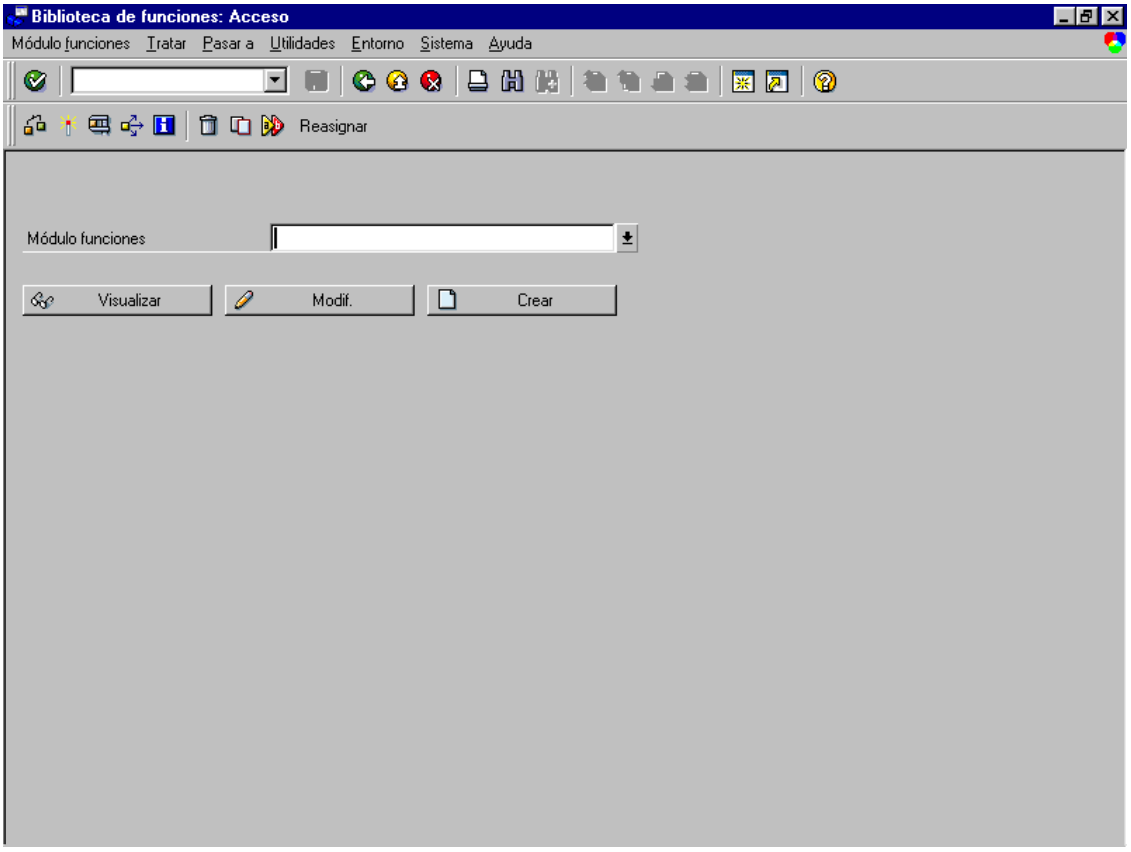


Grupo de funciones: Nombre del grupo de funciones.

Texto breve: Descripción del grupo de funciones.
Ej: ‘Asignación de números’.
Respnsable: Usuario responsable del grupo de funciones.

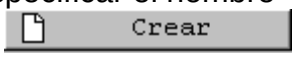
1.3 Datos de gestión

Los módulos de función se mantienen utilizando la biblioteca de funciones.
Ruta de acceso: (En el menú principal de SAP) ‘Herramientas→ Workbench ABAP4-Desarrollo→Biblioteca funciones’ (SE37).



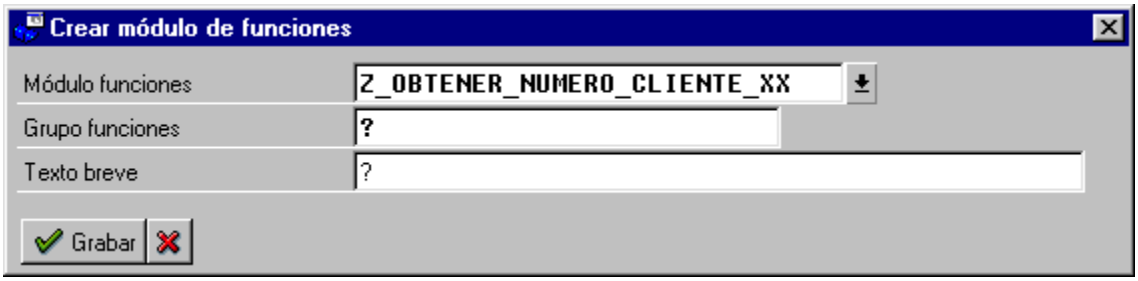
Desde esta pantalla se pueden crear, modificar o visualizar todas las partes de un módulo de función marcando las distintas opciones de objetos parciales.

Ej.: Se creará una función que recibirá como parámetro de entrada una sociedad y devolverá como parámetro de salida el primer número de cliente desocupado en la tabla de clientes ZCLIENXX para esa sociedad.

Para crear un módulo de función se deberá especificar el nombre en la pantalla inicial y pulsar el botón de crear 

con la opción 'Gestión' activada.
Ej.: 'Z_OBTENER_NUMERO_CLIENTE_XX'.

Aparecerá una pantalla en la que se debe indicar el grupo de funciones a la que pertenece el módulo de función junto con su la descripción de la función.
Ej.: 'ZCXX'..
' Devuelve un número de cliente libre'.



Una vez especificado el grupo de función se deberán informar los datos de gestión del módulo de función.

Clasificación:

- Aplicación: Módulo al que pertenece el programa (FI , HHRR …).
  Ej.: '' Multiaplicación.*
- Texto breve: Descripción de la funcionalidad del módulo de función.
  Ej.: 'Determinación de número de cliente'.

Forma ejec.:

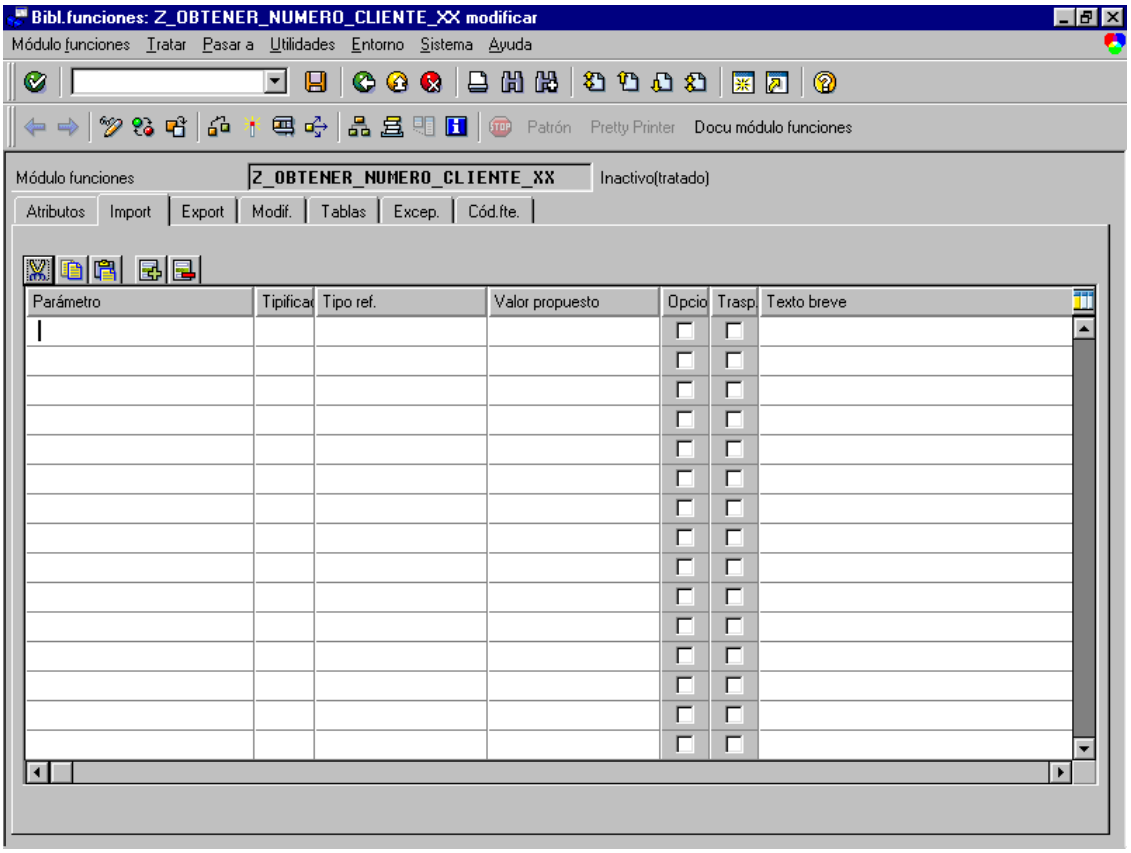
- Normal: Módulo de función normal.
- Apoyo Remote Function Call: Funciones de jecución remota. Estas funciones pueden ser ejecutadas desde otros sistemas externos a SAP.
- Actualizable: Funciones de actualización asíncrona. Se puede especificar el modo de tratamiento de la tarea de actualización (Inicio inmediato, inicio inmediato sin actualización posterior, inico retardado o lanzamiento colectivo.
  Ej.: 'Normal'.

Al finalizar la introducción de los datos de gestión se deberá grabar el módulo de función.

1.4 Parámetros de la función.

1.4.1 Parámetros Import.

Deberemos indicar la lista de parámetros de entrada de la función:



Parámetro: Nombre de los parámetros de entrada.
Tipificación: Tipo de dato del parámetro.
Tipo Ref. Tipo de referencia (Solo se indicará una de estas dos últimas).
Valor Propuesto: Valor por defecto cuando se declara como opcional.
Flag opc. Indicador de parámetro no obligatorio.
Flag. Tras. Indicación de que el parámetro se recibe por valor/referencia.

1.4.2 Parámetros de salida (Export).

Son los parámetros de salida de función.

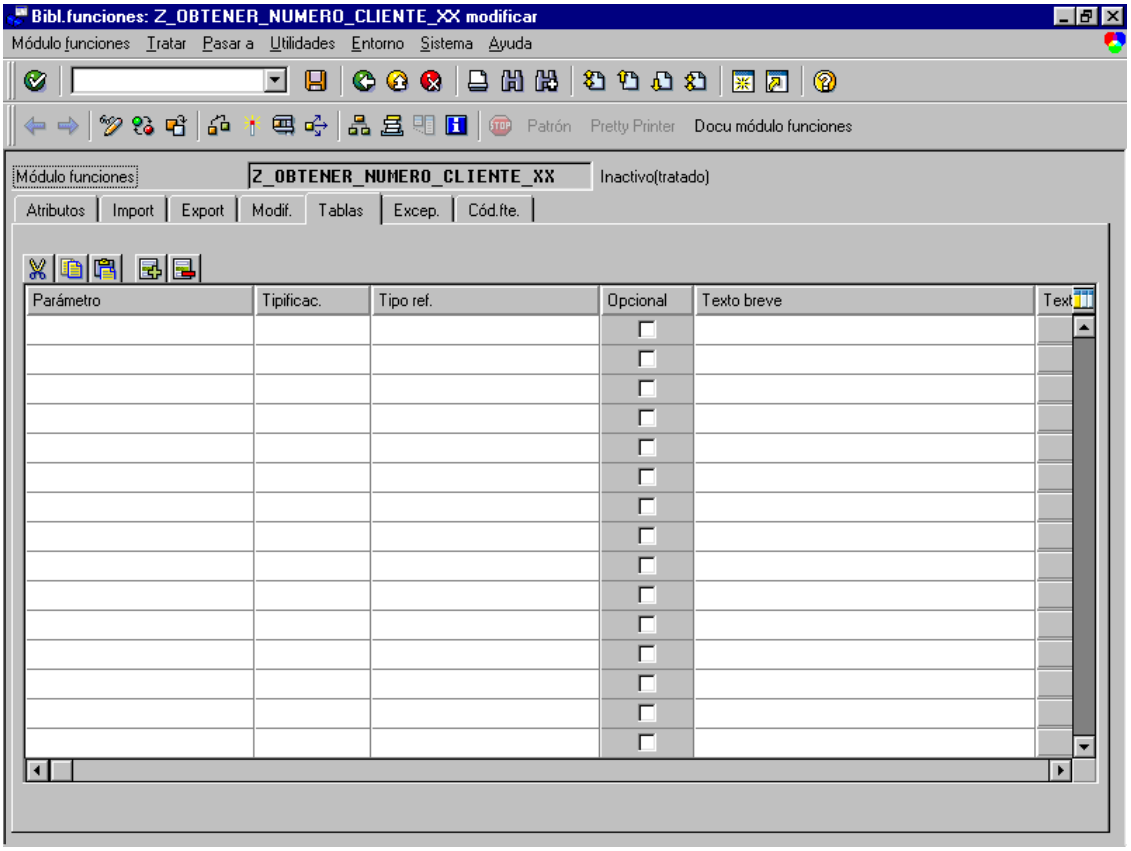
1.4.3 Parámetros (changing):

Parámetros de entrada y salida del módulo de función. Estos parámetros devuelven el valor de salida al finalizar la ejecución del módulo de función sin errores, manteniendo el valor de entrada en caso contrario.

Ej.: Se crea el parámetro import 'BUKRS' con el campo de referencia 'ZCLIENXX-BUKRS' para recibir la sociedad del cliente, y el parámetro export 'NCLIE' con el campo de referencia 'ZCLIENXX-NCLIE' para devolver el número de cliente.

1.5 Tablas

Además de los parámetros de entrada y salida definidos anteriormente, también se pueden traspasar tablas por referencia en la llamada al módulo de función.



- **Parámetro tabla**: Nombre del parámetro de tabla.
- **Tipificaión** : Tipo de datos de la tabla
- **Tipo ref.**: Estructura de referencia del diccionario de datos para especificar la estructura del parámetro de tabla.

- **Opcional**: Si se activa este flag no será obligatorio informar el parámetro de tabla en la llamada al módulo de función.

1.6 Excepciones

Las excepciones son una serie de errores predefinidos en los módulos de función que pueden devolver como valor de retorno de su ejecución en la variable del sistema SY-SUBRC.

> **Excepción:** Se indicará un nombre descriptivo para cada posible error predefinido que va a poder retornar el módulo de función. La posición en la tabla de excepciones se corresponderá con el valor que devolverá en la variable SY-SUBRC (la primera excepción definida devolverá 1, la siguiente 2, etc.).

Por defecto siempre existe la excepción 'OTHERS', aunque aparezca definida, que se utiliza para devolver un error genérico y devuelve en la variable SY-SUBRC el valor siguiente al de la última excepción creada.

Ej.: Se crea la excepción 'SOCIEDAD_INEXISTENTE' que será devuelta cuando la sociedad informada en el parámetro de entrada correspondiente no exista en la tabla estándar de sociedades.

Para devolver una excepción desde el código del modulo de función se utiliza la sentencia RAISE <excepción>, finalizando así la ejecución de la función y devolviendo el código asociado a la excepción en la variable SY-SUBRC. En caso de devolver una excepción no se actualiza el valor de salida de los parámetros CHANGING.

Con la cláusula RAISING de la instrucción MESSAGE se puede dar la posibilidad de que el módulo de función trate el error mostrando el mensaje de error especificado o que devuelva la excepción correspondiente sin mostrar el mensaje de error, en función de si se especifica la cláusula 'EXCEPTIONS' en su llamada. Si no se expecifica la cláusula, el módulo de función mostrará los mensajes de error que tengan la cláusula 'RAISING', finalizando así la ejecución del programa que realiza la llamada, en caso contrario se devolverán las excepciones asociadas a los mensajes traspasando el control de los errores al programa que realiza la llamada al módulo de función.

1.7 Datos globales

Las definiciones globales de datos son compartidas por todos los módulos de función de un grupo de funciones. Las definiciones de objetos globales se mantienen a través de la opción de menú 'Pasar a→ Datos globales'.

Nota: Las definiciones globales y el texto fuente del módulo de función se codifican realmente en el editor ABAP/4 utilizando los mismos comandos que en la codificación de listados.

Ej.: Se define la tabla estándar de sociedades 'T001' en la que se chequeará que la sociedad recibida como parámetro es correcta, y la tabla de clientes 'ZCLIENXX' para seleccionar los números de cliente existentes.

```
Include LZCLITOP:
FUNCTION-POOL ZCLI.                    "MESSAGE-ID ..
************************************************************************
* Definición de tablas
************************************************************************
  TABLES: T001,      " Sociedades
          ZCLIENXX.  " Clientes
```

Los parámetros definidos en el módulo de función son locales, por lo tanto solo son visibles en el cuerpo principal de la función. Para que puedan ser utilizados en las subrutinas, sin necesidad de pasarlos como parámetros, se deberán globalizar utilizando la opción de menú 'Tratar→Interfase→Globalizar parám.' en la pantalla de mantenimiento de parámetros de entrada/salida. Para eliminar la globalización se utiliza la opción de menú 'Tratar→Interfase→Localizar parám.'.

1.8 Código fuente

Al crear una subrutina, haciendo doble click sobre el nombre en la llamada, el sistema propondrá la creación de un Include que contendrá todas las subrutinas del grupo de funciones.

Al finalizar la introducción del texto fuente se deberá grabar, verificar y activar el módulo de función.

Ej.: Se crea la subrutina 'CHEQUEAR_SOCIEDAD' en un nuevo Include LZCLIF01 pasándole como parámetro de entrada el parámetro de sociedad, que chequeará que la sociedad existe en

la tabla estandar de sociedades, devolviendo la excepción 'SOCIEDAD_INEXISTENTE' en caso contrario.

Se crea la subrutina 'SELECCIONAR_NUMERO_CLIENTE' pasándole como parámetro de entrada el parámetro de sociedad, que devolverá como parámetro de salida el primer número de cliente desocupado para la sociedad indicada.

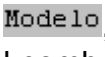
```
Include LZCLIU01:
FUNCTION Z_OBTENER_NUMERO_CLIENTE.
*"----------------------------------------------------------------------
*"*"Interfase local
*"       IMPORTING
*"            VALUE(BUKRS) LIKE  ZCLIENXX-BUKRS
*"       EXPORTING
*"            VALUE(NCLIE) LIKE  ZCLIENXX-NCLIE
*"       EXCEPTIONS
*"            SOCIEDAD_INEXISTENTE
*"----------------------------------------------------------------------

* Se chequea la sociedad
  PERFORM CHEQUEAR_SOCIEDAD USING BUKRS.

* Se selecciona el número de cliente para la sociedad
  PERFORM SELECCIONAR_NUMERO_CLIENTE USING BUKRS
NCLIE.

ENDFUNCTION.

Include LZCLIF01:
*---------------------------------------------------------------------
***INCLUDE LZCLIF01 .
*---------------------------------------------------------------------
*&---------------------------------------------------------------------*
*&      Form  CHEQUEAR_SOCIEDAD
*&---------------------------------------------------------------------*
* Chequea que la sociedad recibida como parámetro existe en la
tabla
* de sociedades.
*----------------------------------------------------------------------*
*  -->  PE_BUKRS        Sociedad
*----------------------------------------------------------------------*
FORM CHEQUEAR_SOCIEDAD USING VALUE(PE_BUKRS)
LIKE ZCLIENXX-BUKRS.

* Se selecciona la sociedad en la tabla de sociedades
```

```
  SELECT SINGLE BUKRS INTO T001-BUKRS
                FROM T001
               WHERE BUKRS = PE_BUKRS.

* Si no se ha seleccionado se devuelve la excepción
SOCIEDAD_INEXISTENTE
  IF SY-SUBRC <> 0.
    RAISE SOCIEDAD_INEXISTENTE.
  ENDIF.

ENDFORM.                    " CHEQUEAR_SOCIEDAD
*&---------------------------------------------------------------------*
*&      Form  SELECCIONAR_NUMERO_CLIENTE
*&---------------------------------------------------------------------*
* Devuelve en el parámetro de salida PS_CLIE el primer número
de cliente
* desocupado en la tabla de clientes para la sociedad especificada
en el
* parámetro de entrada PE_BUKRS.
*----------------------------------------------------------------------*
*      -->PE_BUKRS  Sociedad
*      -->PS_NCLIE  Número de cliente                                  *
*----------------------------------------------------------------------*
FORM SELECCIONAR_NUMERO_CLIENTE
            USING VALUE(PE_BUKRS) LIKE ZCLIENXX-BUKRS
                  PS_NCLIE LIKE ZCLIENXX-NCLIE.

* Se selecciona el nº mayor existente en la tabla

         SELECT MAX( NCLIE )
FROM ZCLIENXX INTO (ZCLIENXX-NCLIE)
WHERE BUKRS = PE_BUKRS.
* Si se encuentra valor máximo
IF ( SY-SUBRC = 0 ).
           PS_NCLIE = ZCLIENXX-NCLIE + 1.
         *  Si no se encuentra valor máximo
   PS_NCLIE = 1.
ENDIF.
* Completamos con ceros por la Izq.
UNPACK PS_NCLIE.

ENDFORM.                    " SELECCIONAR_NUMERO_CLIENTE
```

1.9 Ejecución

Para ejecutar un módulo de función desde la biblioteca de funciones se utiliza la opción de menú 'Utilidades→Entorno test' (F8) de la biblioteca de funciones o pulsando el botón  en la pantalla inicial.

Ej.: Ejecutar la función 'Z_OBTENER_NUMERO_CLIENTE_XX' desde la biblioteca de funciones.

Para ejecutar un módulo de función desde un programa se utiliza la sentencia CALL FUNCTION <función>. Para que el sistema nos proponga la sentencia de la llamada a una función con todos sus parámetros desde el editor ABAP/4 utilizaremos el botón  , marcando la opción 'CALL FUNCTION' y especificando el nombre de la función.

Nota: Los campos que se utilizan en la llamada a un módulo de función deben ser del mismo tipo que los parámetros a los que hacen referencia (definidos en el módulo de función), sino se pueden producir errores en la ejecución del programa.

2. Llamadas a programas y utilización de la memoria.

2.1 Introducción

Para intercambiar datos entre diferentes programas se puede utilizar la memoria SAP y la memoria ABAP/4.

- Memoria SAP.
  Es un área de memoria especifico para cada usuario que se utiliza para almacenar valores que son retenidos durante toda la sesión del usuario.
- Memoria ABAP/4.
  Los valores almacenados en la memoria ABAP/4 solamente son retenidos durante la ejecución de un programa. Esta memoria es utilizada para la transferencia de datos entre dos programa cuando uno de ellos realiza una llamada al otro.

2.2 Parámetros de memoria SAP

En la memoria SAP se pueden almacenar valores asociados a un identificador de tres caracteres, que se mantienen disponibles hasta que finalice la sesión.

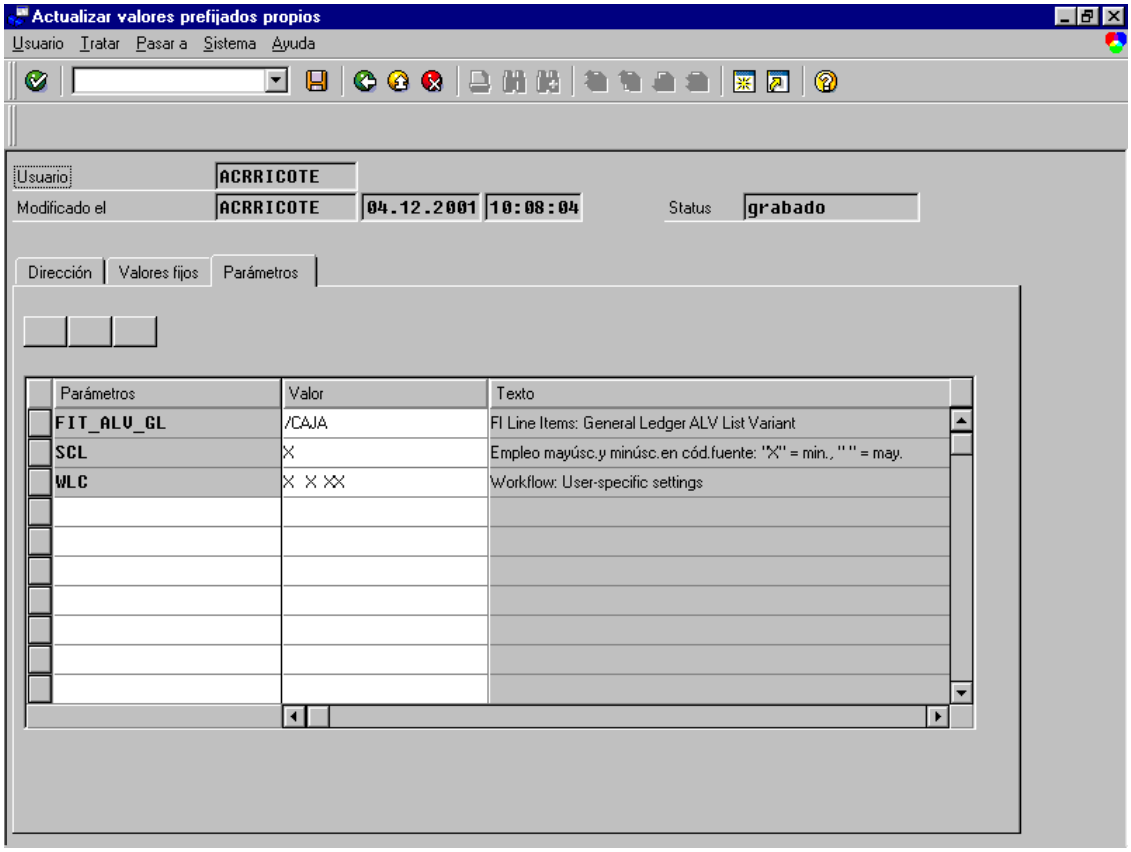
En los campos de las pantallas existen tres atributos relacionados con los parámetros de memoria:

- Id-parám.: Identificador de parámetro para los valores del campo en la memoria SAP. Si el campo está referenciado al diccionario de datos, se informará con el identificador asociado al dominio del campo del diccionario.
- SET Parám.: Al activar este atributo, el sistema almacenará en la memoria SAP el valor que contiene al campo bajo el identificador asociado.
- GET Parám.: Al activar este atributo, el campo se inicializará con el valor definido en la memoria SAP bajo el identificador asociado en lugar de utilizar el valor inicial en función del tipo de dato del campo.

Desde un programa se pueden almacenar y recuperar datos de la memoria SAP con las siguientes sentencias:

- SET PARAMETER.
  La sentencia SET PARAMETER ID <identificador> FIELD <campo> almacena el valor del campo en la memoria SAP asociado al identificador especificado.
- GET PARAMETER.
  La sentencia GET PARAMETER ID <identificador> FIELD <campo> almacena el valor asociado al identificador en la memoria SAP en el campo especificado.

Los parámetros de memoria se inicializan al comenzar una sesión con los valores almacenados en los parámetros de usuario, que se mantienen desde la opción de menú ‘Sistema→Valores prefijados→Datos propios’ (SU3).



- Idp: Identificador de parámetro.
- Valor parámetro: Valor asociado al identificador.

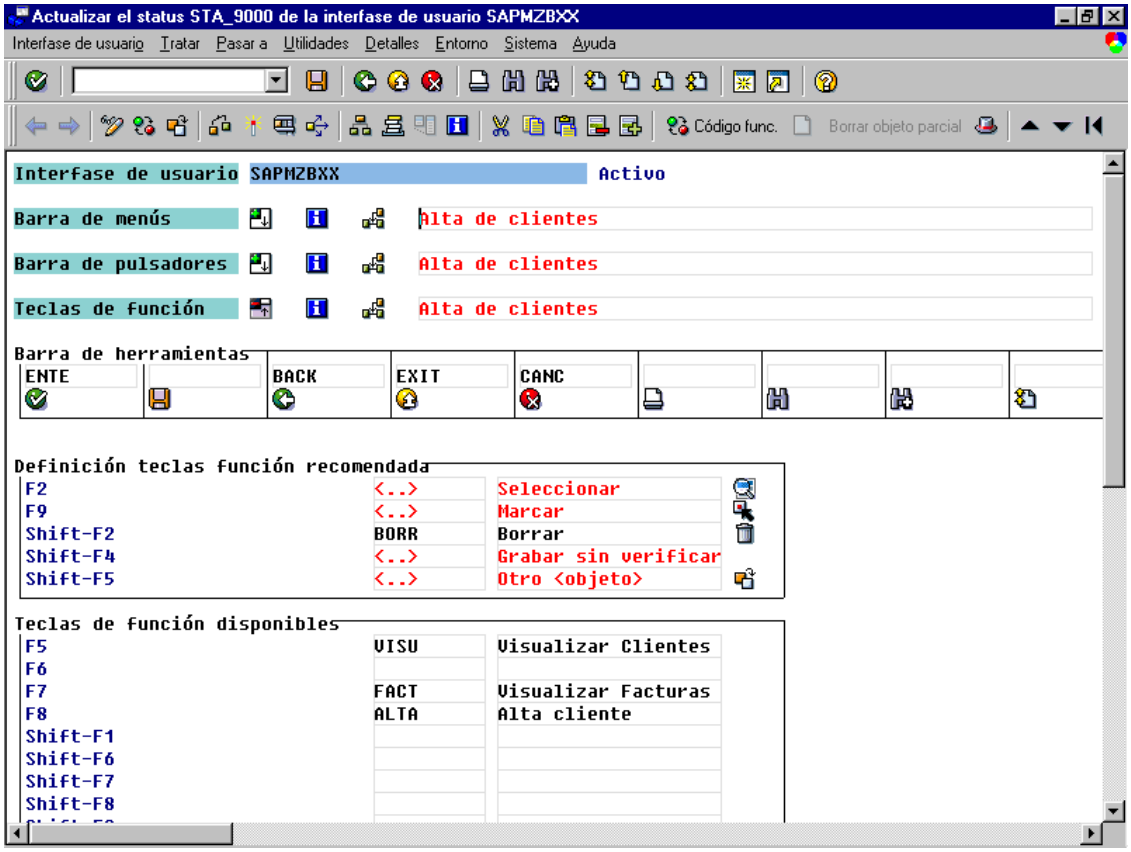
2.3 Sentencias de llamada a programas

Para realizar llamadas a otros programas desde un programa se utilizan las siguientes sentencias:

- SUBMIT <listado> AND RETURN: Realiza una llamada a un listado. Si no se especifica la cláusula 'AND RETURN' finalizará el programa actual y se ejecutará el listado sin regresar al programa actual.
- CALL TRANSACTION <transacción>: Realiza una llamada a una transacción.

Ej.:Se incluirá un pulsador en la pantalla de altas de clientes para ejecutar el listado de visualización de facturas de clientes.

Se incluye un pulsador en la barra de pulsadores el Status 'STA_9000' con el código de función 'FACT' de tipo 'E' (Comando Exit) y el texto 'Visualizar clientes' asociado a la tecla de función 'F6'.



Se añade el procesamiento del código de función 'FACT' en el módulo PAI 'SALIR_9000', realizando una llamada al listado 'ZREPO1XX' en caso de que se selecciones la función 'Visualizar facturas'

```
WHEN 'FACT'.                    " Visualizar facturas
*    Se ejecuta el listado de facturas de clientes
     SUBMIT ZREPO1XX AND RETURN.
```

Se ejecuta el listado de facturas de clientes desde la pantalla de altas de clientes pulsando el botón Visualizar facturas.

2.4 Intercambio de datos a través de la memoria ABAP/4

Al realizar una llamada a un programa desde otro programa, se pueden intercambiar datos a través de la memoria ABAP/4 utilizando las siguientes sentencias:

- EXPORT <objeto> TO MEMORY ID <identificador>.
  Almacena el objeto en la memoria ABAP/4 asociado al identificador especificado, que puede tener una longitud de 32 caracteres. Cada vez que se exportan datos bajo un mismo identificador se sobreescriben los anteriores.

- IMPORT <objeto> FROM MEMORY ID <identificador>.
  Recupera el objeto asociado al identificador especificado de la memoria ABAP/4.
- FREE MEMORY ID <identificador>.
  Libera de la memoria ABAP/4 los datos almacenados bajo el identificador especificado. Si no se especifica la cláusula ‘ID’ se liberará toda la memoria ABAP/4.