



MILE Model Integrator Layout Engine (Capacitación Interna)



Resumen

1.	¿Qué es MILE?	5
1.1.	Concepto básico de la herramienta	5
1.2.	Qué es un Canal.....	5
1.3.	Qué es una Unidad de Información	6
1.4.	Qué es un Adapter	6
1.5.	Qué es un Layout.....	6
1.6.	Composición del layout.....	6
1.6.1.	General.....	7
1.6.2.	Formateo del Archivo	8
1.6.3.	Tratamientos y validaciones.....	10
1.6.4.	Adapters de Rutina Automática	10
1.6.5.	Adapters en MVC	11
1.6.6.	Canales	12
1.6.7.	Detalles del Canal	12
1.6.8.	Salidas.....	13
1.6.9.	Campos.....	14
1.6.10.	Variables.....	16
1.6.11.	Acciones relacionadas	17
1.6.12.	Campo Prejecución.....	18
1.6.13.	Campo Posejecución	20
1.6.14.	Campo Trat. Datos.....	21
1.6.15.	Campo Valid. Operación.....	23
1.7.	Límites de la herramienta	24
2.	Funcionalidades especiales archivo de layouts.....	25
2.1.	Layouts Exportar	25
2.2.	Layouts Importar	25
2.3.	Procesar TXT	25
2.4.	LOG	25
2.5.	Activar/Desactivar.....	26
2.6.	Documentación	26



3. LOG de Procesamiento	27
4. Integración con el Browse	29
5. Características de las exportaciones	34
6. Utilización de la herramienta directamente en aplicaciones AdvPL	35
6.1. Ejemplo de uso de la herramienta directamente en aplicaciones AdvPL	36
6.2. Métodos de la Clase FWMILE	37
6.2.1. Método New()	37
6.2.2. Método Activate()	37
6.2.3. Método Deactivate()	37
6.2.4. Método HasInterface()	38
6.2.5. Método IsActive()	38
6.2.6. Método IsSimulation()	38
6.2.7. Método ClassName()	39
6.2.8. Método SetLayout()	39
6.2.9. Método SetLogGeneration()	39
6.2.10. Método SetLogApplication()	40
6.2.11. Método SetSimulation()	40
6.2.12. Método SetTXTFile()	41
6.2.13. Método SetInterface()	41
6.2.14. Método SetInitLine()	42
6.2.15. Método SetOperation()	42
6.2.16. Método SetAlias()	43
6.2.17. Método Error()	43
6.2.18. Método GetError()	43
6.2.19. Método GetLayout()	44
6.2.20. Método GetOperation()	44
6.2.21. Método GetTXTFile()	44
6.2.22. Método CanActivate()	45
6.2.23. Método Import()	45
6.2.24. Método Export()	45
6.2.25. Método Dialog()	45
6.2.26. Método MakeLog()	46
6.2.27. Método GetIDOperation()	46
7. Agendando importaciones	48



8.	Ejemplificando algunos de los Layouts	50
8.1.	Layout de importación simple sin canal.....	50
8.2.	Layout de importación con 2 canales MASTER - DETAIL.....	51
8.3.	Layout de importación 2 canales para 1 salida	54
8.4.	Layout de importación con ocurrencia única-única.....	57
8.5.	Layout de importación con uso de variables	62
9.	Trabajando con una función específica.....	66
Apéndice A – Estructura del vector sobre datos adicionales.....		69
Apéndice B – Estructura del vector sobre datos del layout.....		70
Apéndice C – Estructura del vector sobre datos de salida.....		72
Apéndice D – Estructura del Vector de datos para Rutina Automática (MSExecAuto)		73



1. ¿Qué es MILE ?

MILE es el acrónimo de Model Integrator Layout Engine. El objetivo de esta herramienta es facilitar la importación / exportación de datos hacia el sistema por medio de rutinas automáticas (MSExecAuto) y/o rutinas desarrolladas en MVC utilizando archivos en formato texto (TXT);

1.1. Concepto básico de la herramienta

La idea básica es mapear la información que se importará o exportará en un layout (vea el ítem 1.5 – Qué es un Layout). Este layout trabaja con el concepto de canales (vea el ítem 1.2 – Qué es un Canal), se realiza la lectura del archivo texto y a través del layout envía los datos para que sean procesados por el adapter (vea el ítem 1.4 – Qué es un Adapter)

1.2. Qué es un Canal

El canal puede utilizarse para definir qué información se está usando, por ejemplo, en una importación de pedido de ventas hay información de encabezado e ítems y la información que compone el encabezado puede estar en un canal y la de los ítems en otro canal.

Ejemplo:

```
| 01 | 000001 | 20121005 | 000001 | 01 | 20 |  
| 02 | P0000002 | 10.00 | 502 |  
| 02 | P0000002 | 3.00 | 502 |  
| 02 | P0000002 | 120.00 | 502 |
```

En este ejemplo el canal “01”, (al comienzo de la línea) podría ser el encabezado y el “02” los ítems.

El canal es una información que debe constar en el archivo texto y que define una separación o tipo para los datos que se están trabajando.

Un layout puede poseer uno, varios o ningún canal.



1.3. Qué es una Unidad de Información

Dentro de un archivo texto importaremos/exportaremos varias informaciones, pero cada layout se refiere a un contexto (facturas, pedidos, clientes, etc.). Dentro de cada contexto, cada conjunto de información es una unidad de información.

Por ejemplo, en un archivo de pedidos de venta pueden existir varios pedidos con varios canales, pero vamos a importar/exportar 1 pedido cada vez. La idea es que cada pedido es una unidad de información. Si vamos a importar clientes, cada uno de los clientes que se importará (independiente de cuántos canales tengan el layout) será una unidad de información y así sucesivamente.

1.4. Qué es un Adapter

El adapter es la aplicación responsable de procesar la información que se obtuvo a partir del archivo texto.

1.5. Qué es un Layout

El layout es la configuración que permite, al leer un archivo texto, identificar los datos contenidos en ese archivo y relacionarlo con la información de los adapters.

1.6. Composición del layout

Un layout puede configurarse a través de la aplicación de mantenimiento de layouts (CFG600) que se encuentra en el módulo del Configurador (SIGACFG) en las opciones Entorno / Aceleradores / MILE / Layouts

Detallamos a continuación la composición de un layout.



TOTVS Série T Manufatura (Microsiga) 02.9.0097

Protheus 11 > Específicos > Miscelanea > MILE

TOTVS Série T Manufatura (Microsiga) 02.9.0098

Manutenção Layouts MILE - IMPORTAÇÃO

Geral

Layout*	Descrição*	Tipo Adapter*	Adapter*
<input type="text"/>	<input type="text"/>	MSExecAuto	<input type="text"/>
Tabela Principal*	Desc. Tabela	Ordem*	Tipo Layout*
<input type="text"/>	<input type="text"/>	1	Importação
Versão do Layout	<input type="text"/>		
1.0			

Formatação do Arquivo

Arquivo TXT*	Separador	Separador Inicial	Separador Final
Canais	Detalhes do Canal		Saídas

Canal

ID Saída

Pós Execução

Descrição*

Ocorrência*

Incluir

Alterar

Excluir

Campos

Variáveis

Seq	Campo	Descrição	Tipo	Origem Dado	Execução
001					

Registro: 91 Status: Em Edição...

Confirmar Fechar Ações relacionadas

Acima

Abaixo

Campos

Início

Fim

TOTVS 2011 Série T Manufatura MSSQL P11 | Administrador | 12/12/2012 | Grupo Totvs 2 / Unidade Belo Ho

1.6.1. General

Contiene datos generales del layout.

Geral

Layout*	Descrição*	Tipo Adapter*	Adapter*
<input type="text"/>	<input type="text"/>	MSExecAuto	<input type="text"/>
Tabela Principal*	Desc. Tabela	Ordem*	Tipo Layout*
<input type="text"/>	<input type="text"/>	1	Importação
Versão do Layout	<input type="text"/>		
1.0			

Posee los campos:

Layout: Código del Layout

Descripción: Descripción del layout

Tipo de Adapter: Tipo de Adapter. Vea el ítem 1.4 – Qué es un Adapter

1=MSExecAuto - Tratamiento por rutina automática (MSExecAuto) (la rutina debe poseer esta característica)



2=MVC - Tratamiento por rutina en MVC

3=Función - Tratamiento por función específica. En este caso los datos se leen mediante la herramienta y se pasan a la función. Para más detalles, vea el ítem 9 - Trabajando con una función específica.

Adapter: Nombre del Adapter.

Cuando el tipo de adapter es:

MSExecAuto: Se informa el nombre de la función de rutina automática (MSExecAuto).

MVC: Se informa el nombre del **FUENTE** (.pr?) que contiene el modelo de datos (MODELDEF).

Función: Se informa el nombre de la función que recibirá los datos leídos

Tabla principal: Alias de la tabla principal utilizada en la importación. El área corriente se apunta para este alias antes de que se efectúe la importación.

Orden: Orden de la tabla principal utilizada en la importación. El orden de la tabla principal se apunta para este orden antes de que se efectúe la importación.

Versión del Layout: Versión del layout. Campo libre para informar la versión del layout.

ATENCIÓN: No hay control de nuevas versiones, este es solo un campo libre para el control manual de la versión.

1.6.2. Formateo del Archivo

Contiene datos sobre el formato del archivo texto.

Posee los campos:

Archivo TXT: Formato del archivo texto

1=**Fijo:** Datos con ancho fijo

2=**Separador:**- Los datos usan algún separador entre sí



Separador: Si el formato del archivo texto es por separador, informar el carácter separador utilizado. Los caracteres que se aceptan son:

| Barra vertical

; Punto y coma

, Coma

/ Barra

- Guión

Tab Tabulación (Chr(9))

ATENCIÓN: Si los datos contienen alguno de los símbolos de los separadores, estos no se importarán correctamente.

Separador Inicial: Cuando el layout posee separadores, define si el layout posee un separador inicial en sus líneas.

Separador Final: Cuando el layout posee separadores, define si el layout posee un separador final en sus líneas.

Origen del Canal: Posición desde donde se encuentra la información de canal del layout.

Cuando el formato de TXT es de ancho fijo se informa la posición inicial y final separada por un guión. Ej. 0001-0005.

Cuando el formato del TXT es por separador se informa la posición de la información. Ej. 0001, esto significa que el canal es el 1º campo de la línea.

Cuando no hay canales, informar 0000-0000 o 0000 de acuerdo con el formato.

Formato Fecha: Formato de los campos de fecha.

1=**dd/mm/aa** Día, mes y año. Ej. 01/05/12 o 01/05/2012

2=**aaaammdd** Año, mes y día Ej. 20120501

Separador Decimal: Tipo de separador de cifras decimales de los datos numéricos.

1=**Punto** Ej. 12345.67

2=**Coma** Ej. 12345,67



ATENCIÓN: Si el tipo de separador de cifras decimales no está correcto, los datos no se importarán correctamente. Si los datos poseen separadores de unidad de mil, se debe tratar en el mismo layout usando el campo **Ejecución**.

Entrada MultiCanal: Informa si el archivo texto posee varios canales.

1.6.3. Tratamientos y validaciones

Contiene los nombres de funciones específicas que pueden definirse para tratamientos puntuales de los datos leídos.

Tratamentos e Validações			
Pré Execução	Pós Execução	Trat.Dados	Valid.Operação
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Posee los campos:

Preejecución: Nombre de la función que se ejecutará antes de la ejecución del adapter. Para más información vea el ítem 1.6.12 - Campo Preejecución

Posejecución: Nombre de la función que se ejecutará luego de la ejecución del adapter. Para más información vea el ítem 1.6.13 - Campo Posejecución

Trat. Datos: Nombre de la función que se ejecutará para el tratamiento de los datos. Para más información vea el ítem 1.6.14 - Campo Trat. Datos

Valid. Operación: Nombre de la función para validación de la operación. Para más información vea el ítem 1.6.15 - Campo Valid. Operación

1.6.4. Adapters de Rutina Automática

Contienen definiciones de características específicas para adapters que son una rutina automática (MSExecAuto).

Adapters de Rotina Automática	
Tipo MSExecAuto	Detalhes Opcional
Modelo 1	Não

Posee los campos:



Tipo MExecAuto: Si el tipo de adapter es una rutina automática (MExecAuto), informar el modelo de la rutina automática. Son compatibles 3 modelos:

1=**Modelo 1** (Tabla simple)

2=**Modelo 2** (1 Tabla con encabezado/ítems)

3=**Modelo 3** (2 Tablas diferentes encabezado/ítems)

Otros modelos **no** son compatibles.

Detalles Opcional: Cuando el adapter es una rutina automática (MExecAuto), define si los detalles de esta rutina automática son opcionales. Este campo debe completarse de acuerdo con cada rutina automática, pues algunas aceptan esta característica y otras no.

IMPORTANTE: La operación ejecutada para adapters de rutina automática (MExecAuto) siempre será solamente **INCLUSIÓN**.

1.6.5. Adapters en MVC

Contienen definiciones de características específicas para adapters en MVC.

Adapters em MVC

Operações Importação	Método de Alteração
Apenas Inclusão	Alteração Direta

Posee los campos:

Operaciones Importación: Para adapters en MVC, define qué operaciones se considerarán en la importación.

Solamente Inclusión: Todos los datos se tratarán siempre como una nueva inclusión.

Inclusión/Modificación: Se verificará la clave única del modelo y se determinará si el dato es una inclusión o modificación, si la clave no se encuentra será una inclusión, si se encuentra, será una modificación.

Método de Modificación: Define el método para efectuar las modificaciones.

Modificación Directa: Los datos se modificarán directamente en el modelo.

Borrar/Incluir: Se realiza el borrado de los datos por el adapter en MVC y luego una nueva inclusión.



1.6.6. Canales

Realiza el mantenimiento de los canales. Aunque el archivo texto (TXT) no posea canales, se deberá crear una canal ficticio con cualquier código.



Botón Incluir: Incluye un nuevo canal.

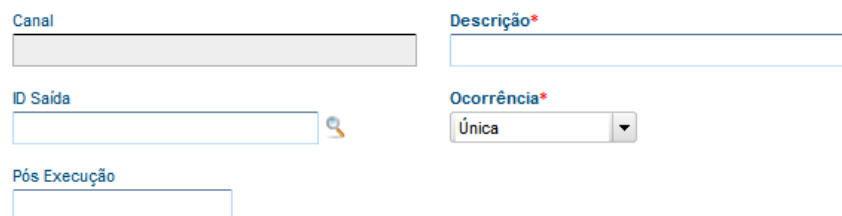
Botón Modificar: Modificar el código de un canal

Botón Borrar: Borra un canal y todos los otros que se encuentren por debajo de él.

1.6.7. Detalles del Canal

Contienen datos sobre el canal.

Detalhes do Canal



Posee los campos:

Canal: Código del canal

Descripción: Descriptivo del canal



ID Salida: Identifica el destino de los datos.

Si el tipo de adapter es rutina automática (MSExecAuto), debe ser MASTER o DETAIL de acuerdo con el modelo de rutina automática (Modelo 1 solo MASTER, el resto MASTER/DETAIL).

Si el tipo de adapter es MVC, debe ser uno de los ID de los componentes del modelo de datos.

Si el tipo de adapter es Función debe ser MASTER o DETAIL de modo similar a la rutina automática (MSExecAuto)

Ocurrencia: Ocurrencia del canal en el layout. Para una unidad de información (1 archivo, 1 pedido, 1 nota, etc.) informa la ocurrencia del canal.

1=**Única:** El canal ocurre solo 1 vez para cada unidad de información (vea el ítem 1.3 – Qué es una Unidad de Información).

N=**Varías:** El canal ocurre varias veces para cada unidad de información (vea el ítem 1.3 – Qué es una Unidad de Información).

Ej.: En un pedido de ventas, el canal de encabezado ocurre 1 vez y el canal de ítems ocurre varias veces, eso para cada pedido.

Obs.: Siempre pensar en las ocurrencias en una unidad de información (1 pedido, 1 nota, etc.)

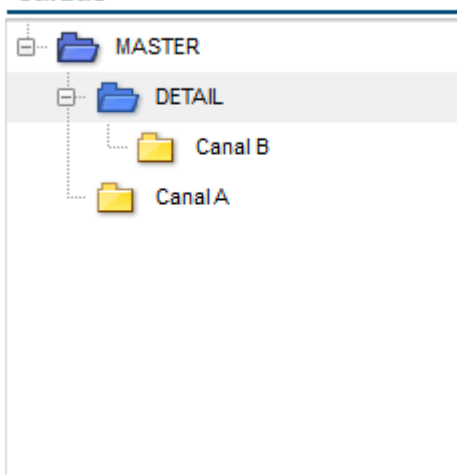
Posejecución: Función ejecutada luego de la lectura de los datos del canal.

1.6.8. Salidas

Componente visual que exhibe la relación entre el canal con el destino. Las carpetas azules son los niveles en el destino y las amarillas los canales vinculados a ellos.



Saídas



1.6.9. Campos

Contienen datos sobre los campos del layout.

Campos					
Seq.	Campo	Descrição	Tipo	Origem Dado	Execução
001					

Posee los campos:

Sec.: Secuencia de los campos.

Obs. En la importación a una rutina automática (MSExecAuto) los campos se ordenan de acuerdo con el diccionario de campos (SX3).

Campo: Nombre del campo.

Descripción: Descripción del campo.

Tipo: Tipo de campo, solo informar si el campo no consta de diccionario, como por ejemplo, campos adicionales de rutina automática (MSExecAuto) (AUTBANCO, AUTBAIXA, etc.)

Origen Dato: Posición donde se encuentra el dato en la línea del archivo texto.

Cuando el formato del archivo texto es de ancho fijo se informa la posición inicial y final separadas por un guión. Ej. 0001-0005.



Cuando el formato del archivo texto es por separador, se informa la posición de la información. Ej. 0001, esto significa que el dato es el 1º campo de la línea y así sucesivamente.

Cuando el dato no viene directamente del archivo texto (viene de una variable o es valor fijo) usar 0000-0000 o 0000 de acuerdo con el formato.

Obs: Si el dato proviene de más de un lugar de la línea del archivo texto, se puede informar los diversos orígenes separados por un punto y coma (;), por ejemplo, 0001-0005;0008-0010 significa que el contenido de las posiciones 1 a 5 y de 8 a 10 de la línea del archivo texto realizarán la composición del dato. Lo mismo se aplica si el formato es por separadores, por ejemplo, 0001;0005 eso significa que el contenido del 1º y del 5º campo de la línea del archivo texto realizarán la composición del dato.

Ejecución: Función ejecutada para el tratamiento del dato leído. Recibe como parámetros las variables xA hasta xZ de acuerdo con la cantidad de orígenes. Por ejemplo, si el origen está definido como 0001-0005, el dato leído se pasa en la variable xA, si está como 0001-0005;0008-0010 el dato leído de 0001-0005 se pasa en la variable xA y el leído de 0008-0010 en la variable xB y así sucesivamente. El retorno de esta ejecución se enviará al campo en vez de directamente al dato leído.

Un ejemplo:

Upper(xA)

Al completar el campo de esta forma, indica que el dato leído (xA) será usado en la función UPPER() y el resultado se enviará al campo correspondiente.

Condición: Condición para importación/exportación de un campo. Es posible definir una expresión AdvPL o función que retornando verdadero (.T.) el dato se enviará al campo y retornando falso (.F.) el dato no se enviará. Recibe como parámetro el dato leído y ya tratado por lo que está definido en **EJECUCIÓN**, en caso de que haya.

Observación: Observaciones que se desean hacer.

Botón Arriba: Mueve un campo una secuencia arriba.

Botón Abajo: Mueve un campo una secuencia abajo.

Botón Campos: Si el tipo del adapter es una rutina automática (MSEExecAuto), complete el layout con los campos de la tabla principal informada. Si el tipo del adapter es MVC



complete con los campos de la estructura del componente del modelo de datos definido en el ID de Salida.

Botón Inicio: Mueve un campo hacia la primera secuencia.

Botón Final: Mueve un campo hacia la última secuencia.

1.6.10. Variables

Contienen datos sobre las variables que pueden crearse para usar en el layout. Estas variables se crearán en la memoria en el transcurso de la importación/exportación. Estas se crearán en el momento en que se realiza la lectura del canal donde esta fue definida. Las variables se completan en la secuencia en las que fueron definidas.

Por ejemplo, la variable cXPTO se creó en el canal X, y deberá completarse con una información del canal, cuando el canal X se lea, la variable cXPTO se completará con el contenido determinado.

Estas variables se crean con el alcance **PRIVATE**.

Seq.	Variável	Tipo	Origem Dado	Execução
001				

Posee los campos:

Sec.: Secuencia de los variables.

Variable: Nombre del variable.

Tipo: Tipo de variable.

Origen Dato: Posición donde se encuentra el dato en la línea del archivo texto.

Cuando el formato del archivo texto es de ancho fijo se informa la posición inicial y final separadas por un guión. Ej. 0001-0005.

Cuando el formato del archivo texto es por separador, se informa la posición de la información. Ej. 0001, esto significa que el dato es el 1º campo de la línea y así sucesivamente.



Cuando el dato no viene directamente del archivo texto (es valor fijo) usar 0000-0000 o 0000 de acuerdo con el formato.

Obs.: Si el dato proviene de más de un lugar de la línea del archivo texto, se puede informar los diversos orígenes separados por un punto y coma (;), por ejemplo, 0001-0005;0008-0010 significa que el contenido de las posiciones 1 a 5 y de 8 a 10 de la línea del archivo texto realizarán la composición del dato. Lo mismo se aplica si el formato es por separadores, por ejemplo, 0001;0005 eso significa que el contenido del 1º y del 5º campo de la línea del archivo texto realizarán la composición del dato.

Ejecución: Función ejecutada para el tratamiento del dato leído. Recibe como parámetros las variables xA hasta xZ de acuerdo con la cantidad de orígenes. Por ejemplo, si el origen está definido como 0001-0005, el dato leído se pasa en la variable xA, si está como 0001-0005;0008-0010 el dato leído de 0001-0005 se pasa en la variable xA y lo leído de 0008-0010 en la variable xB y así sucesivamente. El retorno de esta ejecución se enviará a la variable en vez de directamente hacia el dato leído.

Condición: Condición para el llenado de una variable. Es posible definir una expresión AdvPL o función que, al retornar verdadera (.T.) el dato se enviará a la variable y al retornar falsa (.F.) el dato no se enviará. Recibe como parámetro el dato leído y ya tratado por lo que está definido en EJECUCIÓN, en caso de que haya.

Observación: Observaciones que se desean hacer.

Botón Arriba: Mueve una variable una secuencia arriba.

Botón Abajo: Mueve una variable una secuencia abajo.

Botón Inicio: Mueve una variable hacia la primera secuencia.

Botón Final: Mueve una variable hacia la última secuencia.

1.6.11. Acciones relacionadas

En la opción **Acciones Relacionadas** existen las siguientes funcionalidades:

- **Crear de MVC:** Cuando se va a crear un nuevo layout para importación o exportación utilizando un adapter en MVC esta funcionalidad sirve para completar el layout con la estructura del modelo de datos con sus componentes y campos. Donde:



Fuente MVC: Nombre del **FUENTE** (.pr?) que contiene el modelo de datos (MODELDEF)

Superponer los datos: Indica que los datos del layout se descartarán y recrearán con los del modelo de datos del adapter en MVC.

Incluir en el canal actual: Indica que los datos creados a partir del MVC se colocarán a partir del canal posicionado.

1.6.12. Campo Prejecución

En este campo es posible informar una función que se ejecutará antes de la ejecución del adapter (FUNCTION o USER FUNCTION) y puede ser usada para manejar los datos que se enviarán al adapter. Para la función informada se pasan algunos parámetros y se espera un retorno específico.

Si es una USER FUNCTION, los parámetros se pasan vía PARAMIXB. Ejemplificando:

```
USER FUNCTION XPTO()  
  
Local aParam := PARAMIXB
```

Si es una FUNCTION los parámetros se pasan directamente como parámetros de la propia función, por lo tanto, la función debe estar en el siguiente formato para que logre recibir los parámetros.

```
FUNCTION XPTO( param1, param2, param3, param4, param5 )
```

La función recibirá como parámetros:

- [1] .T. se está ejecutando con interfaz / .F. se está ejecutando sin interfaz
- [2] Vector con información adicional. Vea Apéndice A – Estructura del vector sobre datos adicionales.
- [3] Información de las definiciones del layout. Vea Apéndice B – Estructura del vector sobre datos del layout

Para adapters de rutina automática (MSExecAuto)

- [4] Datos de Salida. Vea Apéndice C – Estructura del vector sobre datos de salida
- [5] Vectores de rutina automática (MSExecAuto). Vea Apéndice D – Estructura del Vector de datos para Rutina Automática (MSExecAuto)



Para adapters en MVC

[4] Modelo de datos completado.

El retorno esperado de la función deberá ser:

Si el adapter es rutina automática (MSExecAuto), el retorno debe ser un array en el formato:

Adapters de rutina automática (MSExecAuto) Modelo 1:

[1] Datos de los campos

[1][x][1] Campo

[1][x][2] Contenido

[1][x][3] NIL

Donde:

[x] Varía para cada campo

[2] Vacío

Adapters de rutina automática (MSExecAuto) Modelo 2 y Modelo 3

[1] Datos de los campos de encabezado

[1][x][1] Campo

[1][x][2] Contenido

[1][x][3] NIL

Donde:

[x] Varía para cada campo

[2] Datos de los campos de ítems

[2][x][1][y][1] Campo

[2][x][1][y][2] Contenido

[2][x][1][y][3] NIL

Donde:

[x] Varía para cada línea



[y] Varía para cada campo

Si el adapter es en MVC:

El retorno debe ser un objeto del modelo de datos del adapter (oModel)

El retorno de esta función se usará en la ejecución del adapter.

1.6.13. Campo Posejecución

En este campo se puede informar una función que se ejecutará luego de la ejecución del adapter (FUNCTION o USER FUNCTION) y puede usarse para algún tratamiento adicional. Para función informada se pasan algunos parámetros y no se espera un retorno específico.

Si es una USER FUNCTION, los parámetros se pasan vía PARAMIXB. Ejemplificando:

```
USER FUNCTION XPTO()  
  
Local aParam := PARAMIXB
```

Si es una FUNCTION los parámetros se pasan directamente como parámetros de la propia función, por lo tanto, la función debe estar en el siguiente formato para que logre recibir los parámetros.

```
FUNCTION XPTO( param1, param2, param3, param4, param5 )
```

La función recibirá como parámetros:

- [1] .T. se está ejecutando con interfaz / .F. se está ejecutando sin interfaz
- [2] Vector con información adicional. Vea Apéndice A – Estructura del vector sobre datos adicionales
- [3] Información de las definiciones del layout. Vea Apéndice B – Estructura del vector sobre datos del layout

Para adapters de rutina automática (MSExecAuto)c

- [4] Datos de Salida. Vea Apéndice C – Estructura del vector sobre datos de salida
- [5] Error en la rutina automática (MSExecAuto), Si .T. hubo un error que impide la importación / .F. la importación se realizó



Para adapters MVC

- [4] Modelo de datos completado.
- [5] Si .T. hubo un error que impide la importación / .F. la importación se realizó

No hay retorno esperado.

1.6.14. Campo Trat. Datos

En este campo es posible informar una función para tratamiento de los datos (FUNCTION o USER FUNCTION). Para la función informada se pasan algunos parámetros y se espera un retorno específico.

Si es una USER FUNCTION, los parámetros se pasan vía PARAMIXB. Ejemplificando:

```
USER FUNCTION XPTO()  
  
Local aParam := PARAMIXB
```

Si es una FUNCTION los parámetros se pasan directamente como parámetros de la propia función, por lo tanto, la función debe estar en el siguiente formato para que logre recibir los parámetros.

```
FUNCTION XPTO( param1, param2, param3, param4 )
```

La función recibirá como parámetros:

- [1] .T. se está ejecutando con interfaz / .F. se está ejecutando sin interfaz
- [2] Vector con información adicional. Vea Apéndice A – Estructura del vector sobre datos adicionales
- [3] Vector con información de las definiciones del layout. Vea Apéndice B – Estructura del vector sobre datos del layout

Si el adapter rutina automática (MSExecAuto):

- [4] Vector con datos de salida. Vea Apéndice C – Estructura del vector sobre datos de salida

Si el adapter es en MVC:



[4] Modelo de datos completado.

El Retorno esperado de la función deberá ser:

Si el adapter es rutina automática (MSExecAuto), el retorno debe ser un array en el formato:

Adapters de rutina automática (MSExecAuto) Modelo 1:

[1] Datos de los campos

[1][x][1] Campo

[1][x][2] Contenido

[1][x][3] NIL

Donde:

[x] Varía para cada campo

[2] Vacío

Adapters de rutina automática (MSExecAuto) Modelo 2 y Modelo 3:

[1] Datos de los campos de encabezado

[1][x][1] Campo

[1][x][2] Contenido

[1][x][3] NIL

Donde:

[x] Varía para cada campo

[2] Datos de los campos de ítems

[2][x][1][y][1] Campo

[2][x][1][y][2] Contenido

[2][x][1][y][3] NIL

Donde:

[x] Varía para cada línea

[y] Varía para cada campo



Si el adapter es en MVC:

El retorno debe ser un objeto del modelo de datos del adapter (Model)

1.6.15. Campo Valid. Operación

En este campo es posible informar una función para validación de la operación (FUNCTION o USER FUNCTION). Para la función informada se pasan algunos parámetros y se espera un retorno específico. Si el retorno es verdadero (.T.) se ejecuta la importación de aquella unidad de información, si el retorno es falso (.F.) no se ejecuta la importación.

Si es una USER FUNCTION, los parámetros se pasan vía PARAMIXB. Ejemplificando:

```
USER FUNCTION XPTO()  
  
Local aParam := PARAMIXB
```

Si es una FUNCTION los parámetros se pasan directamente como parámetros de la propia función, por lo tanto, la función debe estar en el siguiente formato para que logre recibir los parámetros.

```
FUNCTION XPTO( param1, param2, param3, param4, param5 )
```

La función recibirá como parámetros:

- [1] .T. se está ejecutando con interfaz / .F. se está ejecutando sin interfaz
- [2] Vector con información adicional. Vea Apéndice A – Estructura del vector sobre datos adicionales
- [3] Información de las definiciones del layout. Vea Apéndice B – Estructura del vector sobre datos del layout

Para adapters de rutina automática (MSExecAuto)

- [4] Datos de Salida. Vea Apéndice C – Estructura del vector sobre datos de salida
- [5] Vectores de rutina automática (MSExecAuto). Vea Apéndice D – Estructura del Vector de datos para Rutina Automática (MSExecAuto)

Para adapters en MVC

- [4] Modelo de datos completado.



El retorno de esta función deberá ser .T. Continúa con la importación de la unidad de información / .F. no ejecuta la importación de la unidad de información

1.7. Límites de la herramienta

- Para los adapters que son una rutina automática (MSExecAuto), son compatibles solamente los que poseen la estructuración de Modelo1, Modelo2 y Modelo3.
- Para los adapters que son rutina automática (MSExecAuto) se realizan solo operaciones de **INCLUSIÓN**.
- Para los adapters que son una función específica solo es compatible en la salida una estructura de encabezado (MASTER) / ítem (DETAIL).
- Las exportaciones se realizan solo para adapters en MVC y se basan en el modelo de datos del mismo.
- Para importaciones, el tamaño máximo de una línea del archivo texto es de 1022 caracteres.
- El final de línea se determina por los caracteres CR y LF (Chr(13) / Chr(10) o en hexadecimal 0D / 0A)
- La herramienta no tiene control de actualización de versiones para los layouts. A pesar de tener un campo versión, este es solo un campo libre para control manual de la versión.
- Disponible para la línea de productos Microsiga Protheus 2011 a partir del release 11.80



2. Funcionalidades especiales archivo de layouts

La aplicación de mantenimiento de layouts (CFGA600) que se encuentra en el módulo del Configurador (SIGACFG) en las opciones Entorno / Aceleradores / MILE / Layouts, posee algunas funcionalidades especiales.

2.1. Layouts Exportar

Esta funcionalidad permite exportar un layout creado dentro de la herramienta. Se genera un archivo XML que contiene el layout. Esta funcionalidad es útil para realizar copias de seguridad de un layout o transportarlo de un entorno/servidor hacia otro.

2.2. Layouts Importar

Esta funcionalidad permite importar un layout que anteriormente se exportó por la funcionalidad **Exportar**. En caso de que el layout a importar ya exista, este no será superpuesto.

2.3. Procesar TXT

Esta funcionalidad permite procesar un archivo texto utilizando el layout que está posicionado en el Browse.

2.4. LOG

Esta funcionalidad es un facilitador para la visualización de la rutina de log de la herramienta. Se muestran las ocurrencias de log del layout posicionado en el Browse. Vea el ítem 3 - LOG de Procesamiento.

Al seleccionarse, esta opción presentará una pantalla con las opciones de visualización, donde:

Solamente la última operación: Presenta los LOG de la última operación, en caso de que no haya, presentará una pantalla vacía.

Solamente la última operación con LOG: Presenta los LOG de la última operación que generó algún LOG.



Todos los LOG del layout: Presenta todos los LOG de ese layout.

2.5. Activar/Desactivar

Esta funcionalidad activa o desactiva un layout. Un layout desactivado no podrá ser usado en una importación o exportación.

2.6. Documentación

Esta funcionalidad genera un informe que documenta el layout posicionado en el Browse.



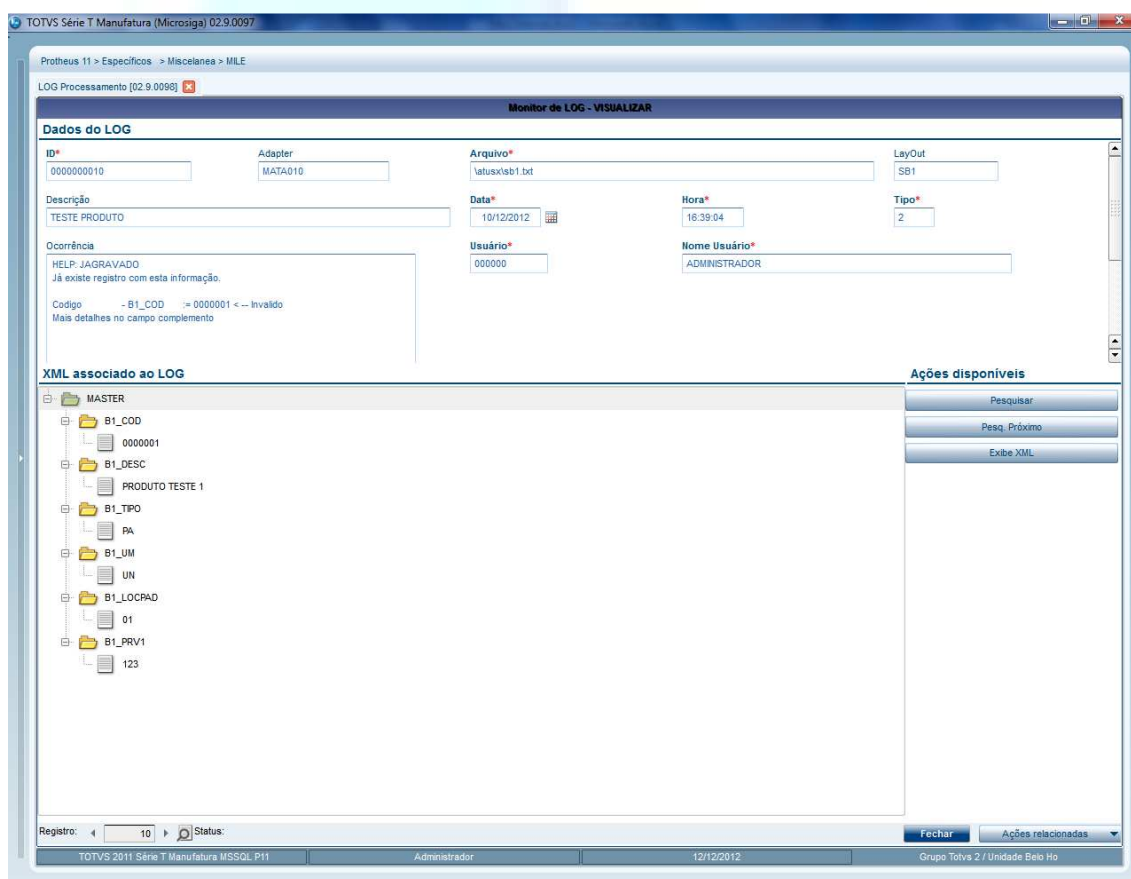


3. LOG de Procesamiento

Al ejecutarse una importación o exportación pueden ocurrir errores o advertencias tanto relacionados con la regla de negocio del contexto que se está trabajando (clientes, pedidos, etc.) como una posible mala construcción del adapter que ocasione errores fatales.

Estas ocurrencias se registran en una tabla de LOG. Para cada error se registra una ocurrencia individualmente.

Estas pueden visualizarse en la **aplicación Log de Procesamiento (CFGA650)** que se encuentra en el módulo del Configurador (**SIGACFG**) en las opciones **Entorno / Aceleradores / MILE / Log de Procesamiento**



La ocurrencia de LOG posee:

ID: ID único de la ocurrencia

Adapter: Adapter que se estaba utilizando en el procesamiento.

Archivo: Nombre del archivo texto que se estaba procesando

Layout: Layout que se estaba utilizando en el procesamiento.



Descripción: Descripción del layout usado en el procesamiento

Fecha: Fecha de la ocurrencia

Hora: Hora de la ocurrencia

Tipo: Tipo de ocurrencia

0 = **Mensaje**

1 = **Aviso**

2 = **Error**

Ocurrencia: Mensaje de error generado.

Usuario: Usuario inició sesión en el momento de la ocurrencia

Nombre del Usuario: Nombre del usuario

Complemento: Mensaje complementario, si existe.

Origen: Indica las líneas inicial y final leídas del archivo texto que se estaban procesando.

ID Operación: ID de la Operación importación o exportación. Por ejemplo, si estamos importando un archivo texto con 10 clientes y 4 de ellos generan un problema, cada ocurrencia tendrá un ID, pero todos ellos tendrán el mismo ID de Operación. El ID de la Operación solo cambiará cuando haya un nuevo procesamiento, ya sea de importación o exportación.

XML Asociado al LOG: XML que contienen los datos enviados al adapter.

Es importante destacar que la herramienta solo registra las ocurrencias, los mensajes y sus contenidos vienen directamente de los adapters. De esta manera, cuanto más claro y preciso el mensaje se haya creado en el adapter, de esta forma quedará registrada.



4. Integración con el Browse

Existe una integración de la herramienta con el Browse de las rutinas del sistema que permite que, al ser registrado un layout para una determinada rutina del sistema (adapter) automáticamente el Browse presente una opción más en su menú que permita realizar la importación o exportación utilizando este layout.

Por ejemplo, imaginemos la creación de un layout para la importación de productos en el sistema la aplicación que realiza, ello es MATA010 que posee el tratamiento para rutina automática (MSExecAuto).

Realizamos la creación del layout:

The screenshot displays the 'Manutenção Layouts MILE - IMPORTACAO' window. The 'Geral' tab is active, showing the following fields:

- Layout*:** PRODUTOS
- Descrição*:** TESTE PARA IMPORTACAO DE PRODUTOS
- Tipo Adapter*:** MSExecAuto (indicated by a red arrow pointing to the Adapter field)
- Adapter*:** MATA010
- Tabela Principal*:** SB1
- Desc. Tabela:** DESCRICAO GENERICA DO PRODUTO
- Ordem*:** 1
- Tipo Layout*:** Importação
- Versão do Layout:** 1.0

The 'Formatação do Arquivo' section shows the 'Canais' list with 'Canal SEMCANAL' selected. The 'Detalhes do Canal' section displays:

- Canal:** SEMCANAL
- Descrição*:** CANAL SEMCANAL
- ID Saída:** MASTER
- Ocorrência*:** Única
- Pós Execução:** (empty field)

The 'Campos' section shows a table of fields:

Seq.	Campo	Descrição	Tipo	Origem Dado	Execução
001	B1_COD	CODIGO DO PRODUTO		0002	
002	B1_DESC	DESCRICAO DO PRODUTO		0003	
003	B1_TIPO	TIPO DE PRODUTO (MPRA...)		0004	
004	B1_UM	UNIDADE DE MEDIDA		0005	

The bottom status bar indicates 'Registro: 5108', 'Status: Em Edição...', and '11/12/2012'.

Luego de la creación del layout al ingresar en la pantalla de la aplicación de **Archivo de Productos**, ya estará disponible una opción para importación en **OPCIONES / IMPORTAR**.



TOTVS Série T Manufatura (Microsiga) 02.9.0097

Protheus 11 > Especificos > Miscelanea > MILE

Produto Normal [02.9.0098]

Detalhes

Filial: L MG 01 -Unidade BELO HO Código: 0000001 Descrição: PRODUTO TESTE 1 Tipo: PA

Unidade: Grupo: Pos.IP/NCM:

Atualizacao de Produtos

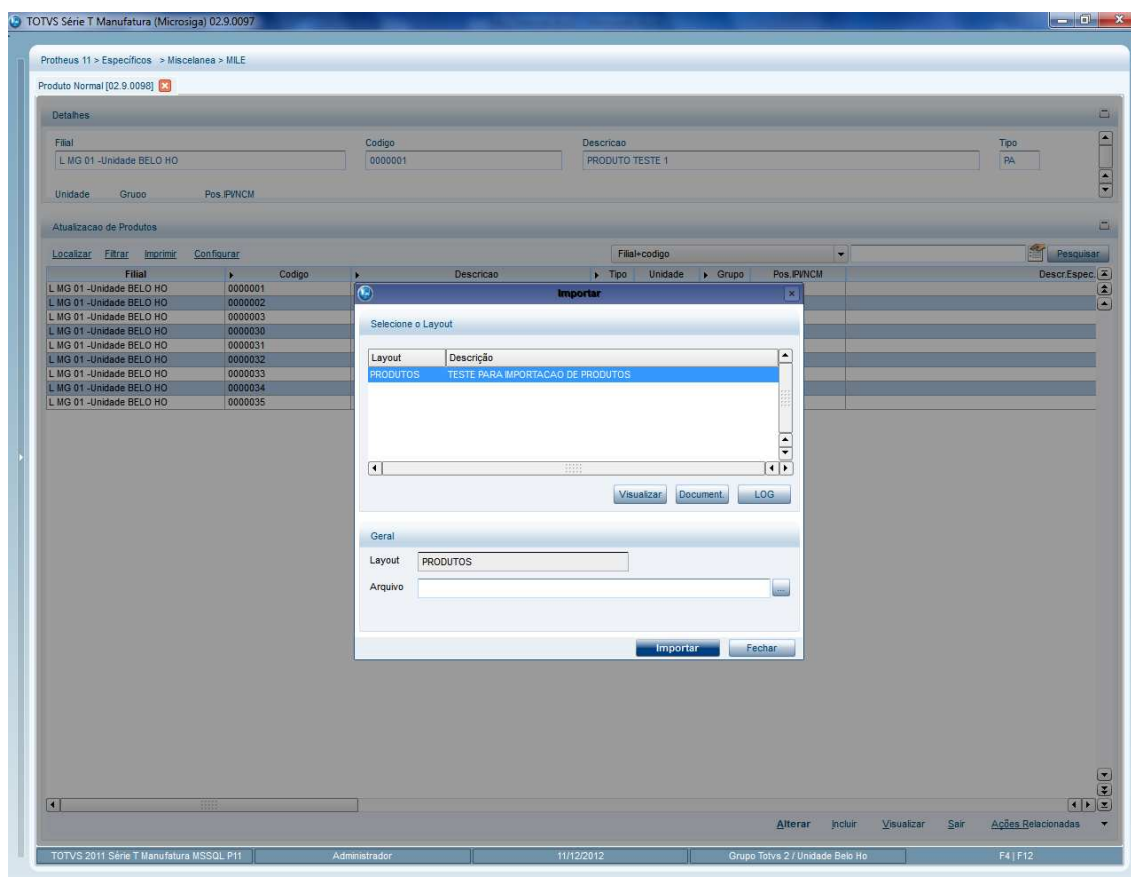
Localizar Filtrar Imprimir Configurar Filial-codigo Pesquisar DescrEspec

Filial	Codigo	Descricao	Tipo	Unidade	Grupo	Pos.IP/NCM
L MG 01 -Unidade BELO HO	0000001	PRODUTO TESTE 1	PA	UN		
L MG 01 -Unidade BELO HO	0000002	PRODUTO TESTE 2	PA	UN		
L MG 01 -Unidade BELO HO	0000003	PRODUTO TESTE 3	PA	UN		
L MG 01 -Unidade BELO HO	0000030	PRODUTO ACABADO P.A. - VENDA	PA	UN		
L MG 01 -Unidade BELO HO	0000031	REVENDA - COMPRA	PA	UN		
L MG 01 -Unidade BELO HO	0000032	INDUSTRIALIZACAO - COMPRA	PA	UN		
L MG 01 -Unidade BELO HO	0000033	CONSUMO - COMPRA	PA	UN		
L MG 01 -Unidade BELO HO	0000034	ATIVO FIXO - COMPRA	PA	UN		
L MG 01 -Unidade BELO HO	0000035	TESTE	PA	UN		

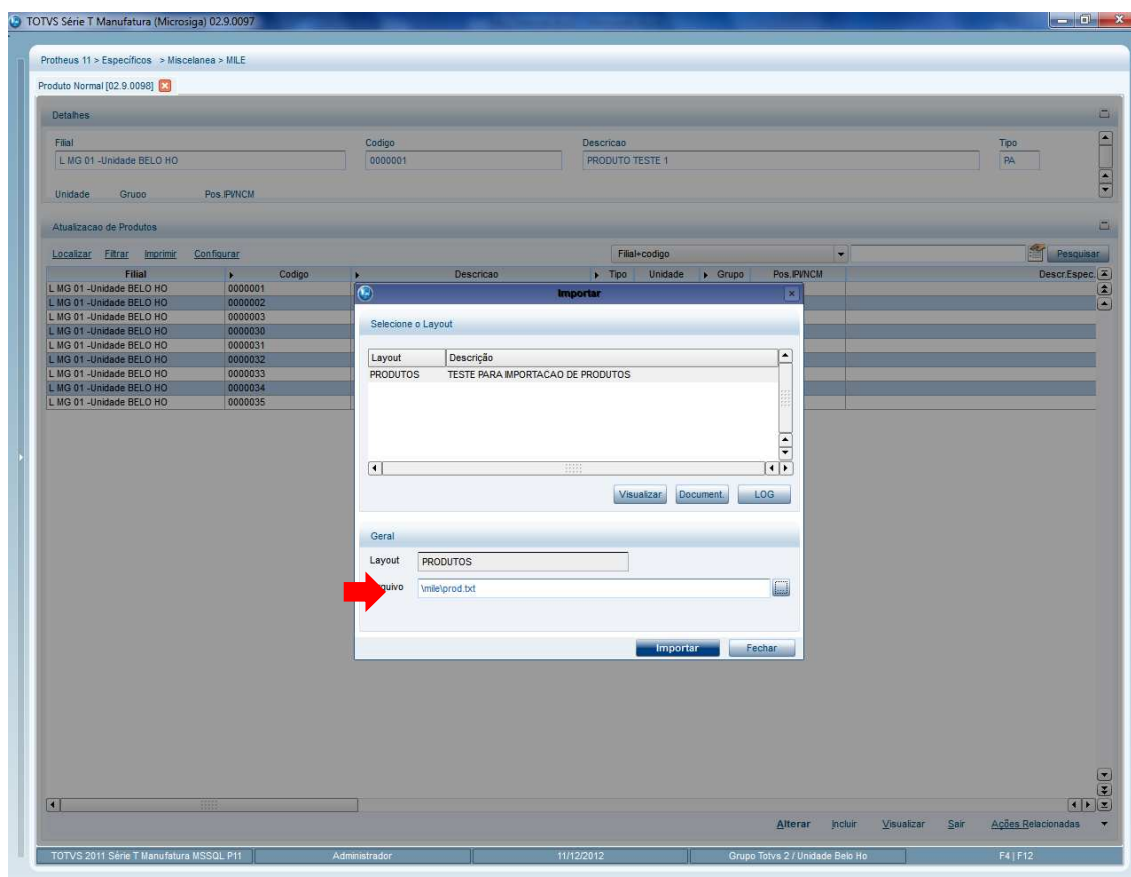
Alterar Incluir Visualizar Sair **Importar** Ações Relacionadas

TOTVS 2011 Série T Manufatura MSSQL P11 Administrador 11/12/2012 Grupo Totvs Z / Unidade Belo Ho F4 | F12

Al hacer clic en **IMPORTAR** se presenta una pantalla con los layouts disponibles para aquella rutina (adapter)

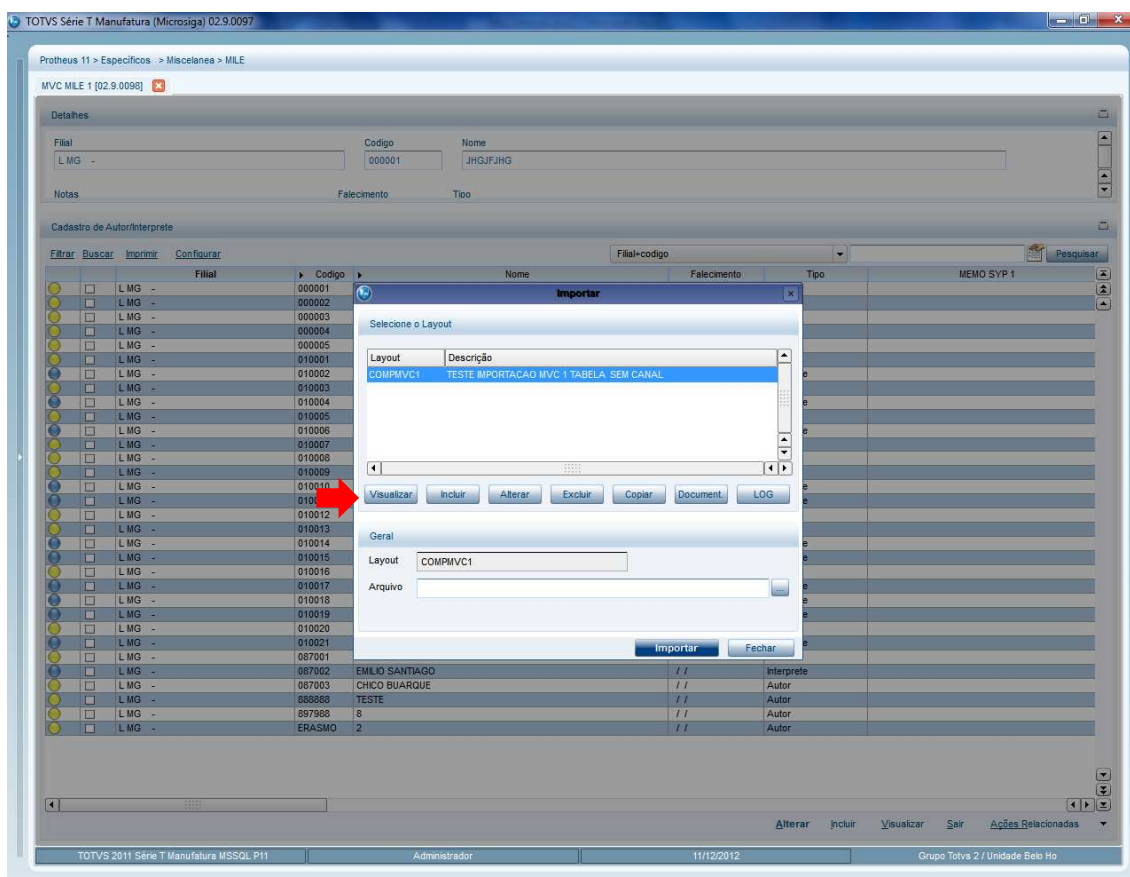


En esta pantalla el usuario que posee acceso a la aplicación de **Archivo de Productos** podrá realizar la importación de un archivo texto utilizando el layout creado.



En esta pantalla también podrá visualizarse el layout, imprimir la documentación o visualizar los LOG generados.

Si la aplicación es en MVC aparecerán algunas opciones más:



Se apresentarán las opciones de Visualizar, Incluir, Modificar, Borrar, Copiar, Documentación y LOG.

Eventualmente el administrador del sistema podrá modificar esos permisos a través del **Archivo de Privilegios** del sistema.

IMPORTANTE: Como no es posible identificar si una rutina está o no preparada para trabajar como rutina automática (MSExecAuto), si se registra un layout para una rutina que no posea esta característica, se presentará la opción de IMPORTAR. Sin embargo, esta opción generará un error al ejecutarse.



5. Características de las exportaciones

Las exportaciones se realizan solo para adapters en MVC y se basan en el modelo de datos del mismo.

Si el componente del modelo es un formulario (FORMFIELD) se generará una línea en el archivo de texto de exportación.

Si el componente del modelo es un grid (FORMGRID) se generará una línea para cada línea del grid en el archivo de texto de exportación.

El layout de exportación hace referencia al alias de la tabla que se exportará, si hay un filtro aplicado a esta tabla, este filtro se respetará.





6. Utilización de la herramienta directamente en aplicaciones AdvPL

Una de las características de la herramienta es que se la pueda utilizar directamente en otras aplicaciones AdvPL. Esto permite su uso en procesos más complejos donde la importación o exportación de un archivo texto sea una de las etapas.

La configuración del layout aun se dará por la aplicación de mantenimiento de layout (vea ítem 1.6 – Composición del layout), pero los procesos de importación o exportación ser realizarán por código AdvPL.

Los log generados en el proceso se grabarán en la pantalla log y podrán visualizarse en la aplicación de **Log de Procesamiento** (vea ítem 3 - LOG de Procesamiento).

Debemos proceder de la siguiente manera:

Estanciamos la clase de FWMILE, a través del método New()

Ej. oFWMILE := FWMILE():New()

Definimos el layout que se utilizará a través del método SetLayout(). El mismo ya debe haber sido creado anteriormente para que pueda utilizarse.

Ej. oFWMILE:SetLayout("TESTE")

Definimos cuál será el archivo texto que se procesará a través del método SetTXTFile().

oFWMILE:SetTXTFile("archivo.txt")

Activamos la clase, a partir de este momento podremos ejecutar las operaciones de importación.

oFWMILE:Activate()

Si el layout es de importación para ejecutar el proceso utilizamos el método Import() si es una exportación el método Export()

oFWMILE:Import()



Para identificar si hubo algún error en el proceso utilizado el método Error()

oFWMILE:Error()

Al final debemos desactivar la clase.

oFWMILE:DeActivate()

6.1. Ejemplo de uso de la herramienta directamente en aplicaciones AdvPL

```
Function MILETest()  
Local aArquivos := { "arquivo1.txt", "arquivo2.txt", "arquivo3.txt" }  
Local cLayout   := "TESTE"  
Local nX        := 0  
Local oFWMILE  
  
oFWMILE := FWMILE():New()  
  
For nX := 1 To Len( aArquivos )  
    oFWMILE:SetLayout( cLayout )  
    oFWMILE:SetTXTFile( aArquivos[nX] )  
  
    If oFWMILE:Activate()  
        oFWMILE:Import()  
  
        If oFWMILE:Error()  
            ApmMsgInfo( I18N( "Processamento terminado. Arquivo #1.", ;  
                { aArquivos[nX] } ) )  
        Else  
            ApmMsgStop( I18N( "Processamento terminado. Arquivo #1." ;  
                "Houve erros no processamento. Verifique as ocorrências no LOG.", ;  
                { aArquivos[nX] } ) )  
        EndIf  
  
        oFWMILE:Deactivate()  
  
    Else  
        ApmMsgStop( I18N( "Problemas para importar. " + oFWMILE:GetError(), {} ) )  
    EndIf  
Next
```



Return NIL

6.2. Métodos de la Clase FWMILE

A continuamos mostramos los principales métodos de la clase FWMILE. Consulte siempre el TDN sobre actualizaciones.

6.2.1. Método New()

Constructor de la clase

Sintaxis:

New(IOpenTable)

Parámetros:

IOpenTable Efectúa o no la apertura de tablas de la clase

Retorno:

No hay retorno esperado (NIL)

6.2.2. Método Activate()

Activación de la clase

Sintaxis

Activate()

Retorno:

.T. Si la clase se activó y .F. en caso contrario

6.2.3. Método Deactivate()

Desactiva la clase

Sintaxis:



Deactivate()

Retorno:

No hay retorno esperado (NIL)

6.2.4. Método HasInterface()

Informa si a está usando interfaz

Sintaxis:

HasInterface()

Retorno:

.T. Si hay interfaz y .F. en caso contrario

6.2.5. Método IsActive()

Informa si la clase está activa

Sintaxis:

IsActive()

Retorno:

.T. Si la clase está activa y .F. en caso contrario

6.2.6. Método IsSimulation()

Informa si está configurado para simular el procesamiento. Solo para adapters en MVC

Sintaxis:

IsSimulation()



Retorno:

.T. Si la clase está configurada como simulación y .F. en caso contrario

6.2.7. Método ClassName()

Nombre de la clase

Sintaxis:

ClassName()

Retorno:

Nombre de la Clase

6.2.8. Método SetLayout()

Define el layout que se utilizará

Sintaxis:

SetLayout(cLayout)

Parámetros:

cLayout Nombre del layout que se usará en el procesamiento

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.9. Método SetLogGeneration()

Define la generación de LOG

Sintaxis:

SetLogGeneration(lLogAuto)



Parámetros:

ILogAuto .T. Generará el LOG y .F. no generará

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.10. Método SetLogApplication()

Define la rutina generación de LOG

Sintaxis:

```
SetLogApplication( bLogApplication )
```

Parámetros:

bLogApplication Bloque de código con la función responsable de la generación de LOG

Ejemplo:

```
oFWMILE:SetLogApplication( { |oObj, xParam| SuaFuncao( oObj, xParam ) } )
```

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.11. Método SetSimulation()

Define si es para apenas simular el procesamiento. Solo para adapters en MVC.

Sintaxis:

```
SetSimulation( ISimulation )
```




Parámetros:

ISimulation Define si será una simulación o no

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.12. Método SetTXTFile()

Define qué archivo texto se procesará

Sintaxis:

SetTXTFile(cTXTFile)

Parámetros:

cTXTFile Nombre del archivo texto que se procesará

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.13. Método SetInterface()

Define si habrá interfaz

Sintaxis:

SetInterface(IHasInterface)

Parámetros:

IHasInterface .T. Define el proceso posee interfaz y .F. en caso contrario

Retorno:



.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.14. Método SetInitLine()

Define cuál es la línea inicial del archivo texto para procesamiento

Sintaxis:

SetInitLine(nInitialLine)

Parámetros:

nInitialLine Número de la línea

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.15. Método SetOperation()

Define cuál es la operación (Importación/Exportación)

Sintaxis:

SetOperation(cOperation)

Parámetros:

cOperation Operación. Que puede ser:

1=Importación #DEFINE MILE_IMPORT

2=Exportación #DEFINE MILE_EXPORT

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario



6.2.16. Método SetAlias()

Define el alias para exportación

Sintaxis:

Método SetAlias(cAlias)

Parámetros:

cAlias Nombre del Alias para exportación

Retorno:

.T. Si la definición se ejecutó con éxito y .F. en caso contrario

6.2.17. Método Error()

Informa si hubo error

Sintaxis:

Error()

Retorno:

.T. Si hubo error y .F. en caso contrario

6.2.18. Método GetError()

Obtiene el último mensaje de error

Sintaxis:

GetError()

Retorno: Último mensaje de error



6.2.19. Método GetLayout()

Obtiene el código del layout definido

Sintaxis:

GetLayout()

Retorno:

Código del Layout

6.2.20. Método GetOperation()

Obtiene la operación

Sintaxis:

GetOperation()

Retorno:

Código de la operación

1=Importación #DEFINE MILE_IMPORT

2=Exportación #DEFINE MILE_EXPORT

6.2.21. Método GetTXTFile()

Obtiene el nombre del archivo texto definido

Sintaxis:

GetTXTFile()

Retorno:

Nombre del archivo texto definido



6.2.22. Método CanActivate()

Verifica si la clase puede ser activada

Sintaxis:

CanActivate()

Retorno:

.T. Si la clase puede activarse y .F. en caso contrario

6.2.23. Método Import()

Ejecuta el bloque de importación definido

Sintaxis:

Import()

Retorno:

No hay retorno esperado (NIL)

6.2.24. Método Export()

Ejecuta el bloque de exportación definido

Sintaxis:

Export()

Retorno:

No hay retorno esperado (NIL)

6.2.25. Método Dialog()

Ejecuta el bloque de DIALOG definido



Sintaxis:

Dialog(cAdapter)

Parámetros:

cAdapter Código del Adapter

Retorno:

No hay retorno esperado (NIL)

6.2.26. Método MakeLog()

Ejecuta rutina de LOG

Sintaxis:

MakeLog(xParam1, xParam2, xParam3, xParam4, xParam5)

Parámetros:

xParam1 Parámetro transferido a la rutina de LOG

xParam2 Parámetro transferido a la rutina de LOG

xParam3 Parámetro transferido a la rutina de LOG

xParam4 Parámetro transferido a la rutina de LOG

xParam5 Parámetro transferido a la rutina de LOG

Retorno:

No hay retorno esperado (NIL)

6.2.27. Método GetIDOperation()

Obtiene el ID de la Operación



Sintaxis:

GetIDOperation()

Retorno:

ID de la Operación

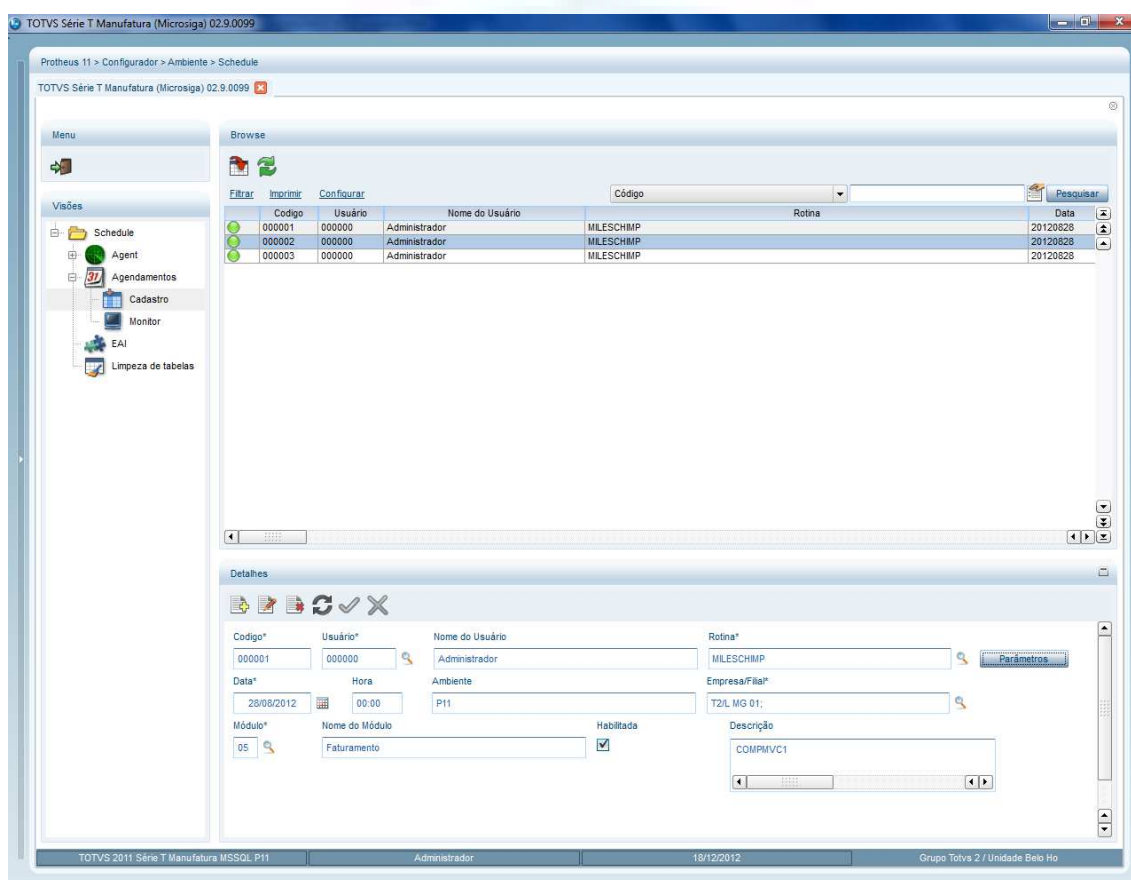




7. Agendando importaciones

La herramienta permite la ejecución de importaciones de forma agendada a través de la herramienta Scheduler del sistema (para más información vea la documentación del Scheduler). Para ello se registra en el Scheduler la aplicación **MILESCHIMP** para la empresa/sucursal que se desea ejecutar las importaciones.

La idea básica es procesar los archivos texto de una carpeta utilizando un determinado layout.



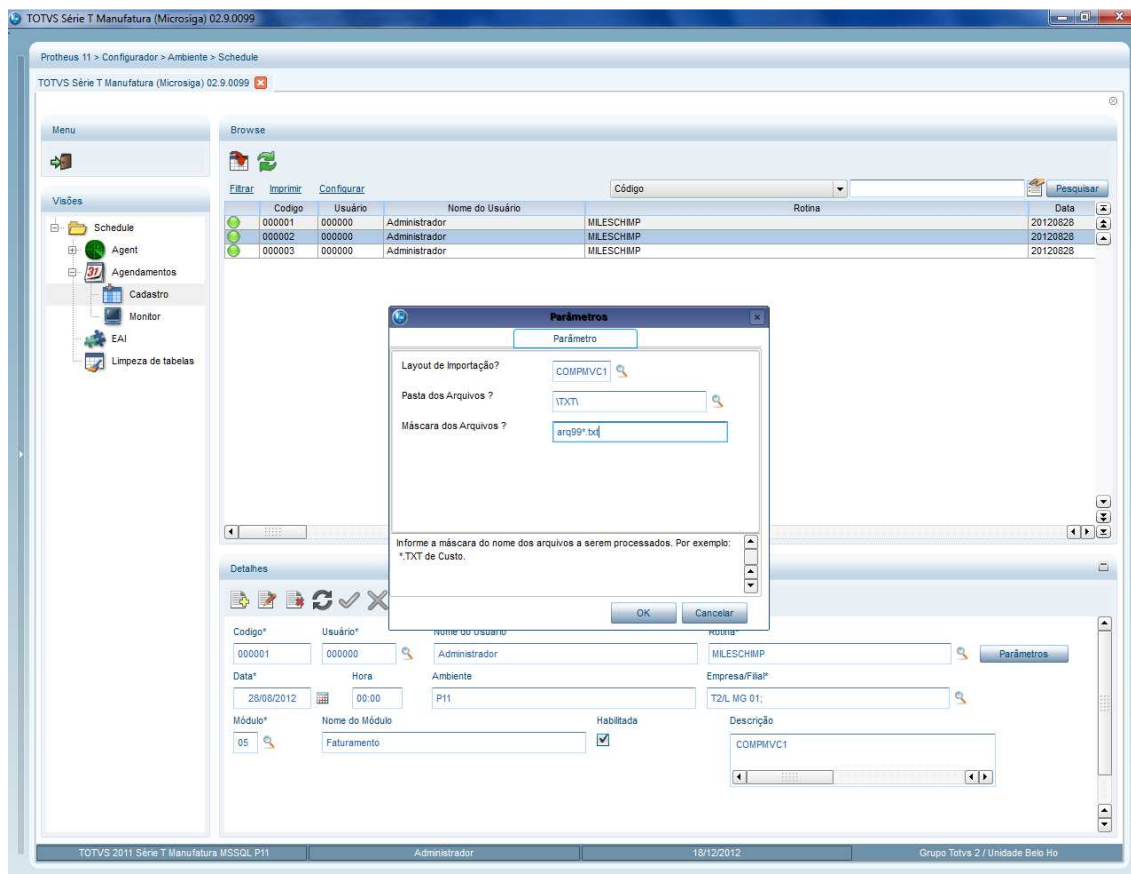
La aplicación **MILESCHIMP** recibe como parámetros:

Layout de Importación: Código del layout de importación que se utilizará en el procesamiento de los archivos texto.

Carpeta de los Archivos: Carpeta donde se encuentran los archivos que se procesarán. Se procesarán todos los archivos de la carpeta de acuerdo con la máscara informada.



Máscara de los Archivos: Máscara del nombre de los archivos texto que se procesarán. Se puede usar el carácter comodín “*” (asterisco), por ejemplo: *.TXT, solo los archivos que respondan a la máscara se procesarán.



Se creará en la carpeta donde se encuentran los archivos texto una subcarpeta con el nombre **\processed**. Los archivos que se procesen se moverán hacia esta subcarpeta.

Los log generados en el proceso se grabarán en la pantalla log y podrán visualizarse en la aplicación de **Log de Procesamiento** (vea ítem 3 - LOG de Procesamiento).



8. Ejemplificando algunos layouts

A continuación mostraremos ejemplos de layout

8.1. Layout de importación simple sin canal

Imaginemos la importación del archivo de productos de un archivo texto por separadores sin canales. Cada línea será un producto a importar.

```
|000001|PRODUTO TESTE 1|PA|UN|01|  
|000002|PRODUTO TESTE 2|PA|UN|01|  
|000003|PRODUTO TESTE 3|PA|UN|01|
```

En este layout:

- 1ª Posición es el Código del Producto
- 2ª Posición es la Descripción
- 3ª Posición es el Tipo
- 4ª Posición es la Unidad de Medida
- 5ª Posición es el Local Estándar

Para crear el layout, como puntos importantes tenemos:

- Registramos **0000** en el campo **Origen del Canal**, pues no hay canales:

Origem do Canal

0000

- Colocamos **NO** en el campo **Entrada MultiCanal**


Entrada MultiCanal

Não

- A pesar de que no haya canales creamos un canal ficticio con cualquier código:



Canais

 Canal NOCHANNEL

Incluir

Alterar

Excluir

- Colocamos el campo **Ocorrência** como **Única**

Ocorrência*

Única

- Como no hay canales, registramos los campos a partir de la 1ª posición.

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	B1_COD	CODIGO DO PRODUTO		0001
002	B1_DESC	DESCRICAO DO PRODUTO		0002
003	B1_TIPO	TIPO DE PRODUTO (MP,PA,..)		0003
004	B1_UM	UNIDADE DE MEDIDA		0004
005	B1_LOCPAD	ARMAZEM PADRAO P/REQUIS.		0005

- Y proceder con el resto del registro del layout

8.2. Layout de importación con 2 canales MASTER - DETAIL

Imaginemos la importación de pedidos de ventas. Cada pedido posee información que es pertinente al encabezado (MASTER) del pedido y otras pertinentes a los ítems (DETAIL)

En el archivo texto, para separar la información de encabezado (MASTER) e ítems (DETAIL) es necesario que cada una esté en un canal diferente. Por ejemplo:

```
|A|000001|000001|00|F|001|  
|B|000001|10.00|502|
```



```
|B|000002|20.00|502| | |
|A|000001|000002|00|F|001|  
|B|000003|10.00|502|  
|A|000001|000003|00|F|001|  
|B|000001|12.00|502|  
|B|000003|50.00|502|
```

En este archivo tenemos los datos del encabezado en el canal A (1ª posición) y los ítems en el canal B.

En este layout:

Canal A encabezado (MASTER)

1ª Posición es el **Canal**

2ª Posición es el **Número del Pedido**

3ª Posición es el **Cliente**

4ª Posición es la **Tienda**

5ª Posición es el **Tipo**

6ª Posición es la **Condición de Pago**

Canal B ítems (DETAIL)

1ª Posición es el **Canal**

2ª Posición es el **Código del Producto**

3ª Posición es la **Cantidad**

4ª Posición es el **Tipo Entrada/Salida**

Para crear el layout, como puntos importantes tenemos:

- Registramos **0001** en el campo **Origen del Canal**, pues los canales están en la 1ª posición:

Origen do Canal

0001

- Colocamos **SÍ** en el campo **Entrada MultiCanal**



Entrada MultiCanal

- Definimos el modelo del MSExecAuto, en este caso, modelo 3

Tipo MSExecAuto

- Registramos los 2 canales

Canais

Canais

- Canal A
 - Canal B

Incluir Alterar Excluir

- El Canal A es el encabezado, entonces como **ID Salida** definimos MASTER

Detalhes do Canal

Canal	Descrição*
<input type="text" value="A"/>	<input type="text" value="CANAL A"/>
ID Saída	Ocorrência*
<input type="text" value="MASTER"/>	<input type="text" value="Única"/>

Pós Execução

- Colocamos el campo **Ocorrência** como **Única**

Ocorrência*

- Como hay canales, registramos los campos a partir de la 2ª posición



Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	C5_NUM	NUMERO DO PEDIDO		0002
002	C5_CLIENTE	CODIGO DO CLIENTE		0003
003	C5_LOJACLI	LOJA DO CLIENTE		0004
004	C5_TIPOCLI	TIPO DO CLIENTE		0005
005	C5_CONDPAG	CONDICAO DE PAGAMENTO		0006

- El Canal B son los ítems, entonces como **ID Salida** definimos DETAIL

Detalhes do Canal

Canal	Descrição*
<input type="text" value="B"/>	<input type="text" value="CANAL B"/>
ID Saída	Ocorrência*
<input type="text" value="DETAIL"/>	<input type="text" value="Várias"/>

- Colocamos el campo **Ocorrência** como **Várias**, pues para cada pedido existen varias líneas del canal B (una para cada ítem)

Ocorrência*
<input type="text" value="Várias"/>

- Como hay canales, registramos los campos a partir de la 2ª posición

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	C6_PRODUTO	CODIGO DO PRODUTO		0002
002	C6_QTDVEN	QUANTIDADE VENDIDA		0003
003	C6_TES	TIPO DE SAIDA DO ITEM		0004

- Y proceder con el resto del registro del layout

8.3. Layout de importación 2 canales para 1 salida

Imaginemos la importación del archivo de productos de un archivo texto por separadores donde existen 2 canales (A y B), donde A tiene algunos datos del producto y B otros. De esta



manera, la combinación de datos del canal A con el canal B generará 1 producto, por lo que tendremos 2 canales en la entrada que convergirán en una sola salida (MASTER)

```
|A|000001|PRODUTO  TESTE  1| |
|B|PA|UN|01|  
|A|000002|PRODUTO  TESTE  2|  
|B|PA|UN|01|  
|A|000003|PRODUTO  TESTE  3|  
|B|PA|UN|01|
```

En este layout:

Para el canal A:

1ª Posición es el **Canal**

2ª Posición es el **Código del Producto**

3ª Posición es la **Descripción**

Para el canal B:

1ª Posición es el **Canal**

2ª Posición es el **Tipo**

3ª Posición es la **Unidad de Medida**

4ª Posición es el **Local Estándar**

Para crear el layout, como puntos importantes tenemos:

- Registramos **0001** en el campo **Origen del Canal**, pues los canales están en la 1ª posición:

Origem do Canal

0001

- Colocamos **Sí** en el campo **Entrada MultiCanal**

Entrada MultiCanal

Sim

- Registramos los 2 canales



Canais

Canais

Canal A

Canal B

Incluir Alterar Excluir

- Para el Canal A definimos como **ID Salida** definimos MASTER

Detalhes do Canal

Canal

A

Descrição*

CANAL A

ID Saída

MASTER

Ocorrência*

Única

- Colocamos el campo **Ocorrência** como **Única**, pues ocurre 1 vez para cada unidad de información (producto)

Ocorrência*

Única

- Como hay canales, registramos los campos a partir de la 2ª posición

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	B1_COD	CODIGO DO PRODUTO		0002
002	B1_DESC	DESCRICAO DO PRODUTO		0003

- Para el Canal B definimos como **ID Salida** definimos MASTER también, pues sus datos irán hacia la misma salida



Detalhes do Canal

Canal	Descrição*
<input type="text" value="B"/>	<input type="text" value="CANAL B"/>
ID Saída	Ocorrência*
<input type="text" value="MASTER"/>	<input type="text" value="Única"/>

- Colocamos el campo **Ocorrência** como **Única**, pues ocurre 1 vez para cada unidad de información (producto)

Ocorrência*

- Como hay canales, registramos los campos a partir de la 2ª posición

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	B1_TIPO	TIPO DE PRODUTO (MP,PA,...)		0002
002	B1_UM	UNIDADE DE MEDIDA		0003
003	B1_LOCPAD	ARMAZEM PADRAO P/REQUIS.		0004

- Y proceder con el resto del registro del layout

8.4. Layout de importación con ocurrencia única-única

Imaginemos un archivo texto para la importación de archivo de bancos, donde hay un canal para definir los datos del banco, un canal para definir los datos de la agencia y otro para definir los de la cuenta corriente, con el agravante de que el canal del banco puede tener varios canales de agencia y este, a su vez, puede tener varios canales de cuenta corriente. Por ejemplo:

```
|BC|341|ITAU|
|AG|0001|AGÊNCIA 001|
|CC|111111|CONTA 1|
|CC|222111|CONTA 2|
|AG|0002|AGÊNCIA 002|
|CC|222112|CONTA 3|
|AG|0003|AGÊNCIA 003|
|CC|333111|CONTA 4|
```



La 1ª posición es el canal, por lo tanto, tenemos BC como el canal del banco, AG como el canal de agencia y CC como el canal de cuenta corriente. Observen que el canal BC sucedió una vez y tenemos canales AG con uno o dos canales CC.

El archivo de bancos es una rutina automática (MSExecAuto) de modelo1, o sea, toda la información tendrá que converger en una única salida.

El agravante en este layout es que estructuralmente se asemeja a un modelo3, pues podríamos, hipotéticamente, decir que el canal de banco sería el encabezado (MASTER) y los ítems (DETAIL),

Para procesar este archivo texto, definiremos en el layout la ocurrencia de cada canal como ÚNICA, de esta forma la herramienta procederá de la siguiente manera:

- Leerá la 1ª línea, que es un canal **BC**
- Leerá la próxima línea del archivo texto, como la ocurrencia de BC es la única, si el canal de la línea leída es **BC**, esta entenderá que es la nueva información e intentará importar la información ya leída antes de continuar.
- Pero el próximo canal es **AG**, y este continúa la lectura.
- Leerá la próxima línea del archivo texto, como la ocurrencia de **AG** también es la única, si el canal de la línea leída es **BC** o **AG**, esta entenderá que es nueva información e intentará importar la información ya leída antes de continuar.
- Pero el próximo canal es **CC**, y este continúa la lectura.
- Leerá la próxima línea del archivo texto, como la ocurrencia de **CC** también es única, si el canal de la línea leída es **BC** o **AG** o **CC**, esta entenderá que es nueva información e intentará importar la información ya leída antes de continuar.
- El próximo canal es **CC**, la herramienta entenderá que es nueva información e intentará importar la información ya leída antes de continuar y comenzará el ciclo nuevamente.

En este tenemos layout:

Para el canal BC:

1ª Posición es el **Canal**



2ª Posición es el **Código del Banco**

3ª Posición es el **Nombre del Banco**

Para el canal AG:

1ª Posición es el **Canal**

2ª Posición es el **Código de la Agencia**

3ª Posición es el **Nombre de la Agencia**

Para el canal CC:

1ª Posición es el **Canal**

2ª Posición es el **Número de Cuenta**

3ª Posición es el **Nombre de la Cuenta**

Para crear el layout, como puntos importantes tenemos:

- Registramos **0001** en el campo **Origen del Canal**, pues los canales están en la 1ª posición:

Origem do Canal

0001

- Colocamos **Sí** en el campo **Entrada MultiCanal**

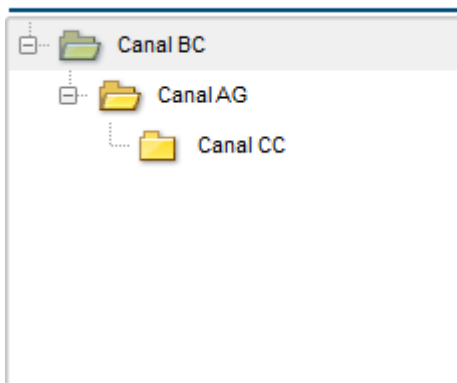
Entrada MultiCanal

Sim

- Registramos los 3 Canales



Canais



Incluir

Alterar

Excluir

- Para el canal **BC** definimos como **ID Salida** definimos MASTER

Detalhes do Canal

Canal	Descrição*
BC	CANAL BC
ID Saída	Ocorrência*
MASTER	Única

- Colocamos el campo **Ocorrência** como **Única**, puesto este ocurre 1 vez para cada unidad de información (banco)

Ocorrência*

Única

- Como hay canales, registramos los campos a partir de la 2ª posición

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	A6_COD	CODIGO DO BANCO		0002
002	A6_NOME	NOME DO BANCO		0003

- Para el canal **AG** definimos como **ID Salida** definimos MASTER también, pues los datos de este irán a la misma salida.



Detalhes do Canal

Canal	Descrição*
AG	CANAL AG
ID Saída	Ocorrência*
MASTER	Única

- Colocamos el campo Ocorrência como Única, pues este ocurre 1 vez para cada unidad de información (banco)

Ocorrência*

Única

- Como hay canales, registramos los campos a partir de la 2ª posición

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	A6_AGENCIA	AGENCIA DO BANCO		0002
002	A6_NOMEAGE	NOME DA AGENCIA		0003

- Para el Canal **CC** definimos como **ID Salida** definimos MASTER también, pues los datos de este irán a la misma salida.

Detalhes do Canal

Canal	Descrição*
CC	CANAL CC
ID Saída	Ocorrência*
MASTER	Única

- Colocamos el campo **Ocorrência** como **Única**, pues este ocurre 1 vez para cada unidad de información (banco)

Ocorrência*

Única

- Como hay canales, registramos los campos a partir de la 2ª posición



Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	A6_NUMCON	CONTA CORRENTE NO BANCO		0002

- Y proceder con el resto del registro del layout

8.5. Layout de importación con uso de variables

Imaginemos la importación de pedidos de ventas. Cada pedido posee información que es pertinente al encabezado (MASTER) del pedido y otras pertinentes a los ítems (DETAIL)

En el archivo texto, para separar la información de encabezado (MASTER) e ítems (DETAIL) es necesario que cada uno esté en un canal diferente, sin embargo, la información con relación al Tipo de Entrada/Salida en vez de venir al canal de los ítems viene en el canal del encabezado. El problema de esta situación es que esa información no tiene un correspondiente en el encabezado, es una información pertinente al ítem, al leer el canal de encabezado tendríamos que almacenar este dato en algún lugar para que pudiéramos usarlo en los ítems.

Este es un caso en el que se recomienda el uso de variables. La idea es que al leer el canal del encabezado la información de Tipo de Entrada/Salida se almacene en una variable en vez de en algún campo y que posteriormente, esa variable se use en los ítems.

Por ejemplo, el archivo texto sería:

```
|A|000001|000001|00|F|001|502|
|B|000001|10.00|
|B|000002|20.00|
|A|000001|000002|00|F|001|502|
|B|000003|10.00|
|A|000001|000003|00|F|001|502|
|B|000001|12.00|
|B|000003|50.00|
```

En este archivo tenemos los datos del encabezado en el canal A (1ª posición) y los ítems en el canal B.

En este layout:



Canal A encabezado (MASTER)

1ª Posición es el **Canal**

2ª Posición es el **Número del Pedido**

3ª Posición es el **Cliente**

4ª Posición es la **Tienda**

5ª Posición es el **Tipo**

6ª Posición es la **Condición de Pago**

7ª Posición es el **Tipo Entrada/Salida**

Canal B ítems (DETAIL)

1ª Posición es el **Canal**

2ª Posición es el **Código del Producto**

3ª Posición es la **Cantidad**

- Para crear el layout, como puntos importantes tenemos:
- Registramos 0001 en el campo Origen del Canal, pues los canales están en la 1ª posición:

Origem do Canal

0001

- Colocamos Sí en el campo **Entrada MultiCanal**

Entrada MultiCanal

Sim

- Definimos el modelo del MExecAuto, en este caso, modelo 3

Tipo MExecAuto

Modelo 3

- Registramos los 2 Canales



Canais

Canais

Canal A

Canal B

Incluir Alterar Excluir

- El Canal A es el encabezado, entonces como **ID Salida** definimos MASTER

Detalhes do Canal

Canal

A

Descrição*

CANAL A

ID Saída

MASTER

Ocorrência*

Única

Pós Execução

Colocamos el campo Ocorrência como Única

Ocorrência*

Única

- Como hay canales, registramos los campos a partir de la 2ª posición

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado
001	C5_NUM	NUMERO DO PEDIDO		0002
002	C5_CLIENTE	CODIGO DO CLIENTE		0003
003	C5_LOJACLI	LOJA DO CLIENTE		0004
004	C5_TIPOCLI	TIPO DO CLIENTE		0005
005	C5_CONDPAG	CONDICAO DE PAGAMENTO		0006

- Registramos la variable que almacenará el Tipo de Entrada/Salida. Por ejemplo, crearemos la variable MEMTES, que recibirá el campo leído del archivo texto. Así cuando se lea el canal A, se creará esa variable.



Variáveis

Seq.	Variável	Tipo	Origem Dado	Execução
001	MENTES	Caracter	0007	

- El Canal B son los ítems, entonces como **ID Salida** definimos DETAIL

Detalhes do Canal

Canal	Descrição*
<input type="text" value="B"/>	<input type="text" value="CANAL B"/>
ID Saída	Ocorrência*
<input type="text" value="DETAIL"/> 	<input type="text" value="Várias"/> ▼

- Colocamos el campo Ocorrência como Varias, pues para cada pedido existen varias líneas del canal B (una para cada ítem)

Ocorrência*

▼

- Como hay canales, registramos los campos a partir de la 2ª posición y el campo de Tipo de Entrada/Salida en vez de venir del texto de la variable MENTES. Para ello usamos el campo **EJECUCIÓN** (vea el ítem 1.6.9 - Campos)

Campos

Seq.	Campo	Descrição	Tipo	Origem Dado	Execução
001	C6_PRODUTO	CODIGO DO PRODUTO		0002	
002	C6_QTDVEN	QUANTIDADE VENDIDA		0003	
003	C6_TES	TIPO DE SAIDA DO ITEM		0000	M->MENTES

- Y proceder con el resto del registro del layout



9. Trabajando con una función específica

En la herramienta aun hay posibilidad de además de trabajar con adapters en rutina automática (MSExecAuto) o en MVC, se utilice una función específica para tratar los datos. Esta podrá ser una FUNCTION o USER FUNCTION .

Esta función será la responsable de realizar todos los tratamientos y persistencias que los datos necesiten, así como generar ocurrencias en el LOG, si es necesario.

La herramienta realizará lecturas de los datos y los pasará a través de parámetros para la función. Se pasará una unidad de información por cada vez (vea el ítem 1.3 – Qué es una Unidad de Información)

Solo es compatible en la salida de una estructura de MASTER/DETAIL. El archivo texto puede tener hasta más de un canal, sin embargo, estos deben converger hacia esa estructura.

Si es una USER FUNCTION, los parámetros se pasan vía PARAMIXB. Ejemplificando:

```
USER FUNCTION XPTO()  
  
Local aParam := PARAMIXB
```

Si es una FUNCTION los parámetros se pasan directamente como parámetros de la propia función, por lo tanto, la función debe estar en el siguiente formato para que logre recibir los parámetros.

```
FUNCTION XPTO( param1, param2, param3, param4 )
```

La función recibirá como parámetros:

- [1] .T. se está ejecutando con interfaz / .F. se está ejecutando sin interfaz
- [2] Vector con información adicional. Vea



Apéndice A – Estructura del vector sobre datos adicionales





Apéndice A – Estructura del vector sobre datos adicionales

- [3] Información de las definiciones del layout. Vea Apéndice B – Estructura del vector sobre datos del layout
- [4] Datos de Salida. Vea Apéndice C – Estructura del vector sobre datos de salida





Apéndice A – Estructura del vector sobre datos adicionales

[2] Vector con información adicional.

[2][1] Línea inicial leída en el archivo texto

[2][2] Línea final leída en el archivo texto

[2][3] nombre del archivo texto que se está procesando





Apéndice B – Estructura del vector sobre datos del layout

[3] Información de las definiciones del layout

[3][1] Datos Generales

[3][1][1]	Código	DEFINE XZ1LAYOUT
[3][1][2]	Tipo de adapter	DEFINE XZ1TYPE
[3][1][3]	Adapter	DEFINE XZ1ADAPT
[3][1][4]	Formato del archivo texto	DEFINE XZ1STRUCT
[3][1][5]	Separador	DEFINE XZ1SEPARA
[3][1][6]	Tipo MSExecAuto	DEFINE XZ1TYPEXA
[3][1][7]	Separador Inicial	DEFINE XZ1SEPINI
[3][1][8]	Separador Final	DEFINE XZ1SEPFIN
[3][1][8]	Tabla Principal	DEFINE XZ1TABLE
[3][1][10]	Orden Tab. Principal	DEFINE XZ1ORDER
[3][1][11]	Descripción	DEFINE XZ1DESC
[3][1][12]	Origen del Canal	DEFINE XZ1SOURCE
[3][1][13]	Función prejecución	DEFINE XZ1PRE
[3][1][14]	Función posejecución	DEFINE XZ1POS
[3][1][15]	Función de Tratamiento de datos	DEFINE XZ1TDATA
[3][1][16]	Tipo Fecha	DEFINE XZ1TIPDAT
[3][1][17]	Entrada Multicanal	DEFINE XZ1EMULTC
[3][1][18]	Detalles Opcional	DEFINE XZ1DETOPC
[3][1][19]	Importación/Exportación	DEFINE XZ1IMPEXP
[3][1][20]	Separador Decimal	DEFINE XZ1DECSEP
[3][1][21]	Operaciones Importación	DEFINE XZ1MVCOPT
[3][1][22]	Método de Modificación	DEFINE XZ1MVCMET



[3][1][23] Función de Validación de la Operación DEFINE XZ1CANDO

[3][2] Datos de los Canales

[3][2][x][1] Canal DEFINE XZ2CHANEL

[3][2][x][2] Canal Superior DEFINE XZ2SUPER

[3][2][x][3] Descripción DEFINE XZ2DESC

[3][2][x][4] ID Salida DEFINE XZ2IDOUT

[3][2][x][5] Ocurrencias DEFINE XZ2OCCURS

[3][2][x][6] Función de posejecución DEFINE XZ2POS

[3][2][x][7] Campos DEFINE XZ2XZ4

[3][2][x][7][y][1] Campo DEFINE XZ4FIELD

[3][2][x][7][y][2] Tipo DEFINE XZ4TYPFLD

[3][2][x][7][y][3] Tamaño DEFINE XZ4SIZFLD

[3][2][x][7][y][4] Decimal DEFINE XZ4DECFLD

[3][2][x][7][y][5] Origen DEFINE XZ4SOURCE

[3][2][x][7][y][6] Ejecución DEFINE XZ4EXEC

[3][2][x][7][y][7] Condición DEFINE XZ4COND

[3][2][x][8] Variables DEFINE XZ2XZ5

[3][2][x][8][z][1] Variables DEFINE XZ5FIELD

[3][2][x][8][z][2] Tipo DEFINE XZ5TYPFLD

[3][2][x][8][z][3] Tamaño DEFINE XZ5SIZFLD

[3][2][x][8][z][4] Decimal DEFINE XZ5DECFLD

[3][2][x][8][z][5] Origen DEFINE XZ5SOURCE

[3][2][x][8][z][6] Ejecución DEFINE XZ5EXEC

[3][2][x][8][z][7] Condición DEFINE XZ5COND

Donde:

[x] Varía para cada Canal

[y] Varía para cada Campo definido de cada Canal



[z] Varía para cada Variable definida de cada Canal

Apéndice C – Estructura del vector sobre datos de salida

[4] Datos de Salida. Apéndice C

[4][x][1] Id de Salida

[4][x][2] Control interno. No manejar.

[4][x][3] Control interno. No manejar.

[4][x][4] Datos de Salida

[4][x][4][y][1] Campo

[4][x][4][y][2] Contenido

[4][x][4][y][3] NIL

Donde:

[x] Varía para cada ID de Salida

[y] Varía para cada Campo

[4][x][5] Control interno. No manejar



Apéndice D – Estructura del Vector de datos para Rutina Automática (MSExecAuto)

Para rutina automática (MSExecAuto) Modelo 1

[5][1] Datos de los campos

[5][1][x][1] Campo

[5][1][x][2] Contenido

[5][1][x][3] NIL

Donde:

[x] Varía para cada campo

[5][2] Vacío

Para rutina automática (MSExecAuto) Modelo 2 y 3

[5][1] Datos de los campos de encabezado

[5][1][x][1] Campo

[5][1][x][2] Contenido

[5][x][3] NIL

Donde:

[x] Varía para cada campo

[5][2] Datos de los campos de ítems

[5][2][x][1][y][1] Campo

[5][2][x][1][y][2] Contenido

[5][2][x][1][y][3] NIL

Donde:

[x] Varía para cada línea



[y] Varía para cada campo

