*PROCESO DE DATOS*

*Mail Input*

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Autor** | **Fecha** | **Detalle** |
| 1.0 | Leonardo Lagostena | 19/09/2016 | Primera versión |
|  |  |  |  |

## Especificación de Diseño para Mail Input

Objetivo

Disponer de una herramienta que permita procesar mails existentes en una casilla de correo específica, pudiendo especificar un intervalo de tiempo y un criterio de selección.

Descripción de la funcionalidad

La herramienta permitirá procesar mails sobre una cuenta específica usando criterios y scops preestablecidos, con la finalidad de extraer la información que estos pudieran contener, ya sea en sus adjuntos o en su contenido. Para lograr su adecuado proceso, será necesario especificar qué parte o partes deben procesarse, y en qué repositorio se volcará la información encontrada. Salvo en el caso de los adjuntos, que si se deben recuperar sólo se procede a almacenarlos en el repositorio indicado, en el resto, el asunto y cuerpo del o los mails son guardados en un archivo XML específico, o en su defecto en el campo RESULT del Dispatcher.

Para realizar esto, se dispondrá de un mecanismo de parametrización basado en XML, que permitirá configurar el comportamiento de la herramienta a los efectos de concretar la tarea.

<?xml version="1.0" encoding="UTF-8"?>

<Parameters>

<Source>

<Provider Host='' Port='' SSL='' Username='' Password='' />

<Input TimeFrame='{all|today|yerterday|lastweek|lastmonth}' Content='{all|attach|subject|body}'>

<Criteria Subject='' Body='' BodyRegEx='' From='' Attach='' />

</Input>

</Source>

<Target>

<Output Log='' AttachFolder='' AttachMask='' MailFolder='' MailMask='' Action='{delete|moveTo:readfolder}' />

</Target>

</Parameters>

Básicamente el XML consta de dos secciones centrales que indican la entrada y salida del proceso, es decir, Source y Target.

La sección Source se divide en dos partes principales, Provider e Input. El Provider indica el tipo de servidor a utilizar y los atributos necesarios para acceder la cuenta a procesar. Mientras que Input permite especificar el criterio de selección que se empleará, junto con el intervalo temporal sobre el que tendrá efecto dicha selección, y que parte o partes del mail interesa trabajar.

Por otro lado, el nodo Target consta también de dos partes, el output en el que se debe indicar dónde volcar la información que pueda ser extraída, y finalmente, si se debe realizar alguna acción sobre el mail procesado antes de continuar con el siguiente.

Este componente debe soportar ejecución concurrente y para su correcto funcionamiento, sólo será necesario proveer los parámetros adecuados dentro del campo Parameter del Dispatcher destinado a tal fin. Como sucede con cualquier otro componente, podrá ser ejecutado en demanda o programado con el módulo Scheduler del Dispatcher. Podrá contar con un archivo de configuración local propio, pero sólo para aspectos que no afecten su universalidad.

Parámetros

Basado en XML, especifica los distintos comandos que el componente recibirá y que le permitirá realizar el trabajo. Constará de dos secciones principales denominadas Source y Target.

Nodo Source

La sección Source se divide en dos partes principales, Provider e Input. El Provider indica el tipo de servidor a utilizar (en este caso siempre será IMAP), junto con los atributos necesarios para acceder a la cuenta a procesar, es decir: Host, Port, SSL, Username y Password. Por otro lado Input permite especificar el criterio de selección que se empleara, junto con el intervalo temporal sobre el que tendrá efecto dicha selección, y qué parte o partes del mail interesa procesar. Dentro del nodo Criteria, se puede especificar distintas secuencias de caracteres a buscar para Subject, Body, From y Attach. Los tres primeros atributos se usan para determinar elegibilidad o selección. Los mismos se pueden combinar para complementar la localización del mail a procesar. Si se indica más de un atributo, se evalúan con el operador AND. Una vez identificado los mails que cumplen con el criterio (dentro del TimeFrame), se aplica sobre cada adjunto lo indicado por Attach pero en este caso actuando como un filtro. Pueden además usarse Wildcards (\* ó ?) para filtrar nombres de adjuntos.

Como opción, el tributo Body puede usarse con expresiones regulares, usando el atributo con la extensión BodyRegEx. Lógicamente no se puede usar al mismo tiempo ambas expresiones, solo una.

Consecuentemente, si no se indican atributos en el nodo Criteria, se asume que todos los mails dentro del TimeFrame indicado y con todos sus adjuntos, son elegibles. Determinada la selección de los mails a procesar, se extrae lo indicado por el nodo Content. Si este no se especifica, se asume el parámetro All es decir mensaje (Subject + Body) y adjunto, de lo contrario se puede especificar uno u otro según se requiera. Ejemplo:

<Input TimeFrame="yesterday" />

<Criteria Subject='[BASE]' From='test@mail.com' Attach='\*.zip' Content="attach" />

</Input>

En este caso se seleccionarán todos los mails cuyo asunto contenga el texto [BASE] y que el From sea [test@mail.com](mailto:test@mail.com), y que se hayan recibido en algún momento del día anterior. El atributo Attach indica que, de esos mails seleccionados, solo interesa los adjuntos que tengan extensión zip. Ahora bien, esto último tiene sentido, si y solo si, Content indica que se haga algo con los adjuntos. En el ejemplo, específicamente indica que se debe procesar únicamente los adjuntos, por lo que se descargarán los que tengan extensión zip, según indica el filtro. También se podría especificar un nombre completo de archivo en caso de conocerse, por ejemplo: Attach="base.zip".

Nodo Target

El nodo Target tiene un único nodo Output en el que se debe indicar dónde volcar la información que pueda ser extraída y Action que indica si se debe realizar alguna acción sobre el mail procesado, antes de continuar con el siguiente.

<Target>

<Output Log="" AttachFolder="" AttachMask="" MailFolder="" MailMask="" Action="{delete|moveTo:readfolder}" />

</Target>

Ejemplos

<Target>

<Output Log="\\BUE\Mails\Logs\Log[##].txt"

AttachFolder="\\BUE\Mails\Attach"

AttachMask="AttFile[YYYY]-[##].\*"

MailFolder="\\BUE\Mails"

MailMask="MailsFile[YYYY]-[##].xml "

Action="moveTo:OldFolder" />

</Target>

<Target>

<Output Log="\\BUE\Mails\Logs\Log[##].txt"

AttachFolder=""

AttachMask=""

MailFolder=""

MailMask=""

Action="" />

</Target>

En el primer ejemplo se especifican todos los atributos posibles con un valor concreto. Sin embargo el resultado dependerá de lo especificado en el nodo Source. Es decir, especificar un folder para los Attach tiene sentido si en Content se especificó All o Attach, de lo contrario no tendría efecto. Lo mismo para el caso del folder para guardar el mensaje. En general, cada atributo se rige por las siguientes reglas.

* **AttachFolder**: Si no se especifica folder de salida, no procesa los adjuntos, aunque se haya expecificado en Content.
* **AttachMask**: Si no se indica, se guarda con el nombre original cualquiera que sea. Si ese nombre existe, renombra con el mismo patrón que usa el SO (no pisa). En cambio si se especifica una máscara, se siguen las reglas de las mismas.
* **MailFolder**: Si es vacío y está en Content, el contenido del mail se vuelca en formato XML en el campo RESULT del Dispatcher. Debe poder guardarse más de un mail.
* **MailMask**: Si no se indica pero se especificó folder, se guarda con este formato: eMails[YYYYMMDD]-[##].xml

En Action se puede especificar una operación a realizar después de procesar el mail. Si está vacío no se realizará ninguna acción, sino se procederá a eliminar el mansaje si contiene Delete o moverlo al folder que se indique MoveTo:<folder>

Para el caso del almacenamiento del mensaje, todo lo que se procese (es decir todos los mails que sean elegibles), se volcarán en un único archivo en formato XML, conteniendo en cada mensaje, el conjunto de atributos que les corresponde. Como se indicó, ese archivo puede ser uno explícito, o generado a partir del uso de una máscara específica. A continuación se muestra un ejemplo de la estructura de ese archivo:

<?xml version="1.0" encoding="UTF-8"?>

<Emails>

<Email

From="source@source.com"

To="someone@example.com"

CC="someoneelse@example.com”

Subject="This is the subject"

Body="This is the body of an email message."

AttachFolder="\\UNC\SavedAttachFolder"

AttachFiles="attach1.img;attach2.jpg;attach3.zip" />

<Email

From="source2@source.com"

To="someone2@example.com"

CC="someoneelse2@example.com"

Subject="This is the second message subject"

Body="This is second body of an email message."

AttachFolder="\\UNC\SavedAttachFolder"

AttachFiles="attachA.xml;attachB.txt;attachC.rar" />

</Emails>

En un mismo archivo se volcará el contenido de los mensajes que se hayan procesado, describiendo así una colección que luego podrá ser a su vez operada con las herramientas usuales.

Es muy importante notar que el nombre de los adjuntos que queden guardados en el archivo, se debe corresponder con lo almacenado en primera instancia. Esto significa que si al momento de almacenar el adjunto, el nombre cambió del original por algún motivo, entonces ese nuevo nombre es el que debe quedar registrado. Si se descargaron adjuntos, el campo AttachFolder agrega el dato del directorio en el cual se efectuó el almacenamiento, sino debe quedar vacío. Este conjunto de campos así dispuestos permite, no sólo recuperar los mensajes completos, sino que también posibilitan localizar y relacionar los adjuntos descargados con su mensaje original.

Finalmente, si se especifica que se debe procesar el mensaje Content="(message|all)" pero no se indica MailFolder entonces todo el XML que se genere se almacenará en el campo RESULT del Dispatcher. Si el mismo no fuera suficiente, debe truncarse el contenido y el resultado erróneo debe notificarse por el canal usual.

Log de errores

Como se indicó al describir la estructura Target de los parámetros, se contempla la posibilidad de reportar errores específicos en un archivo creado a tal efecto. Para ello se dispone, dentro del nodo Output, del atributo Log. En dicho atributo se puede especificar el path y nombre completo del archivo que se generará. Normalmente, todos los eventos que deban ser notificados tienen siempre “por default” su salida hacia el campo RESULT del dispatcher, junto con el código de retorno apropiado. Sin embargo no será posible utilizar este campo si el mismo se está usando ya para la salida de información. En estos casos concretos, cualquier error producido no podrá ser descripto en detalle (solo se sabrá que el componente termino con errores, pero en el campo RESULT se encontrará la información que se pudo efectivamente procesar). Por ese motivo se prevé la posibilidad de guardar esos datos directamente en un archivo específico.

Disponiendo entonces de un archivo de Log, es posible almacenar información exhaustiva del proceso de pasaje de datos, a los fines de detectar problemas. Es importante notar, que salvo que se produzca un fallo general en el componente, si la lectura o escritura de un registro falla, el proceso continúa con el registro siguiente hasta finalizar el proceso de la totalidad del archivo.

Ante la eventualidad de un registro de error, es importante incluir toda la información que sea relevante, pero especialmente se indicará el número de registro en el que se produzco el fallo y la fuente (source o target). Estos archivos además, soportarán la utilización de máscaras en su nombre, que tendrán el mismo uso que lo especificado para los archivos de salida.

Máscara para nombres de archivos dinámicos

Se prevé el uso de máscaras como parte del nombre de archivo a los efectos de facilitar la construcción dinámica de dichos nombres. Para distinguirlos de otros literales serán especificados entre corchetes []

Existirán dos tipos, de fecha y de secuencia. En el caso de la Fecha, se usarán los siguientes descriptores:

YYYY Año actual, puede abreviarse: YY mostraría 16, YYY sería 016, Y = 6

MM Mes actual, no puede abreviarse: 05

DD Día actual, no puede abreviarse: 26

HH Hora, siempre en formato 24hs, ej: 11 serían las once de la mañana

mm Minutos, de 00 - 59

ss Segundos, de 00 - 59

Para la secuencia se usará el #, también encerrado entre corchetes.

### Secuencia de 001 a 999.

Alcanzado el máximo vuelve a empezar pisando el archivo existente, si ese fuera el caso. Para operar con secuencias correctamente, es necesario determinar la última existente y desde ahí proceder con el siguiente paso en la numeración. Es decir, se busca el último archivo creado (según su fecha de creación) y se procede a leer el nombre, identificando la parte componente de la máscara. Si no hay patrón coincidente, se crea el archivo con secuencia inicial, caso contrario se toma la secuencia encontrada y se incrementa en 1 la misma, para crear el archivo requerido.

Ejemplos:

File[YY]-[MM].xls File06-05.xls

File[#####].xls File00001.xls

File[YYYY]-[##].xml File2016-01.xml

File[##]-[###].xml File01-[###].xml (sólo una secuencia por archivo)

[YYYYMM][[##]].xml 201605[01].xml

File201[Y]-[MMDD].xml File2016-0526.xml