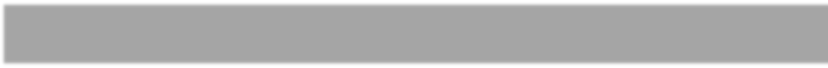
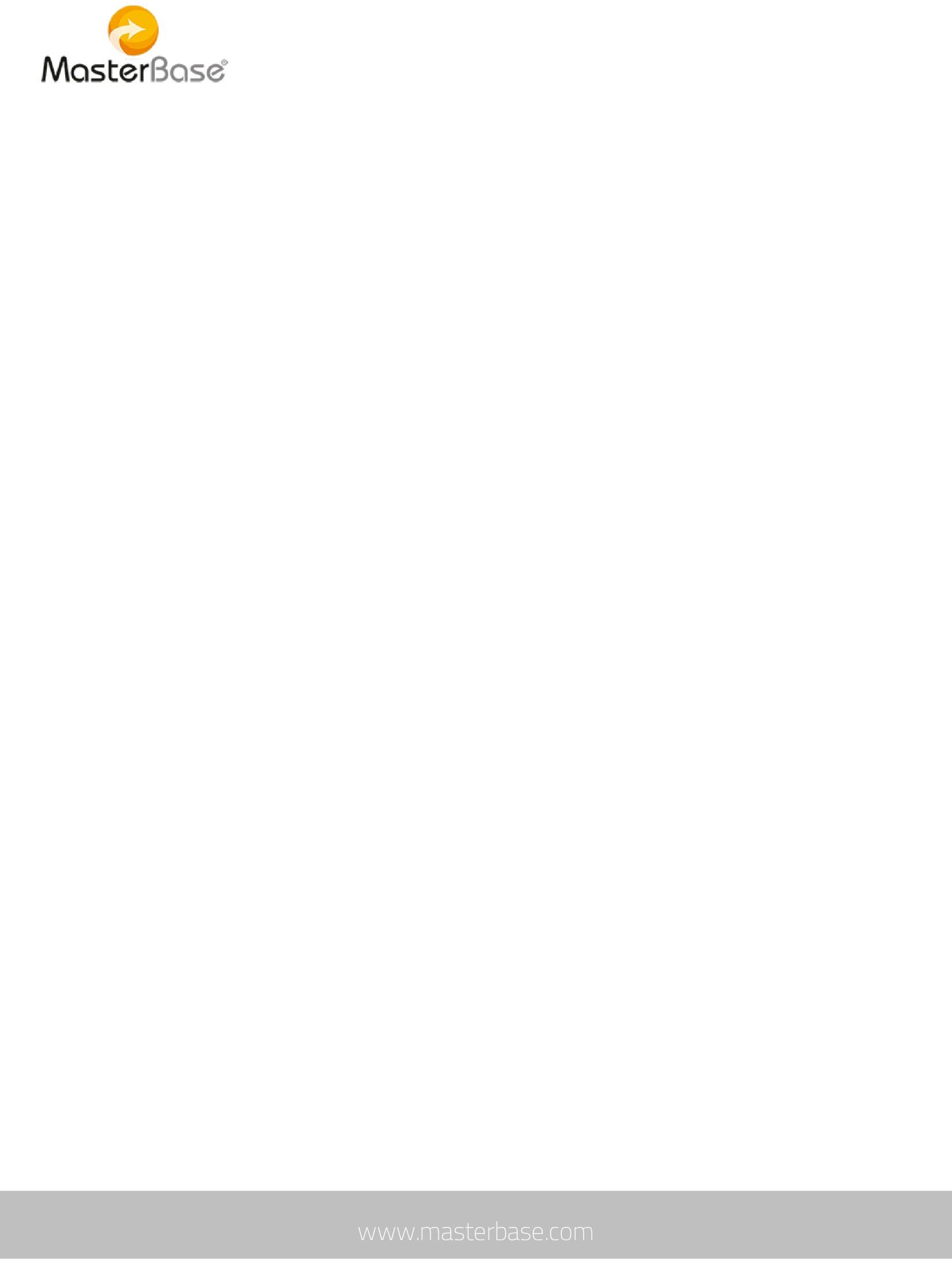


MasterBase®

Configuración de Autómatas Virtuales



Configuración Autómatas Virtuales

# Objetivo del documento

El presente documento tiene como objetivo definir la configuración virtual de un *Autómata A1*, para que pueda procesar la información contenida en un Dato Fuente especifico y generar un conjunto de Etiquetas de Contacto asociados a esta información.

# Conceptos básicos

A continuación, se describen algunos conceptos básicos a tener en cuenta para el entendimiento del proceso de generación de Etiquetas de Contacto realizado por el *Autómata A1.*

*Dato Fuente:*

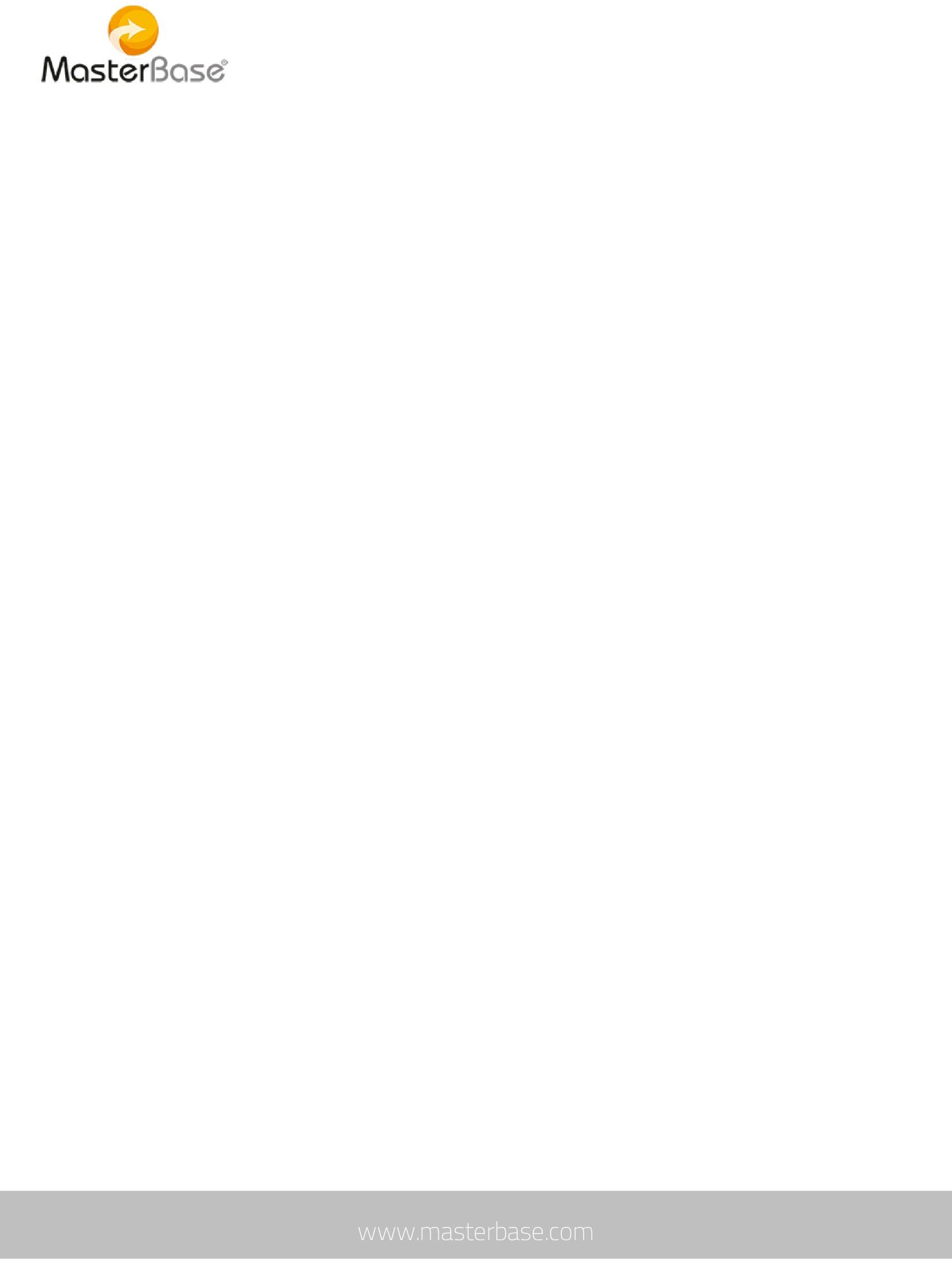
Se entiende por Dato fuente (Feed) a cualquier conjunto de información de tipo Nombre:Valor representada en archivos del tipo XML, JSON o CSV.

*Dato fuente de Sistema:*

Corresponde a Datos Fuentes generados en base a la información contenida en las tablas de Logs de MasterBase®. Esta información está compuesta por los envíos, aperturas, clicks, rebotes y desuscripciones. Los Datos Fuentes de sistema son generados automáticamente y no pueden ser modificados por el cliente.

*Contenedor:*

Corresponde al lugar donde se almacenan físicamente los Datos Fuentes, Existen 20 Contenedores donde se pueden almacenar Datos Fuentes de los cuales los primeros 5 están reservados para los Datos Fuentes de sistema.

La siguiente tabla detalla los identificadores de los Contenedores de sistema:

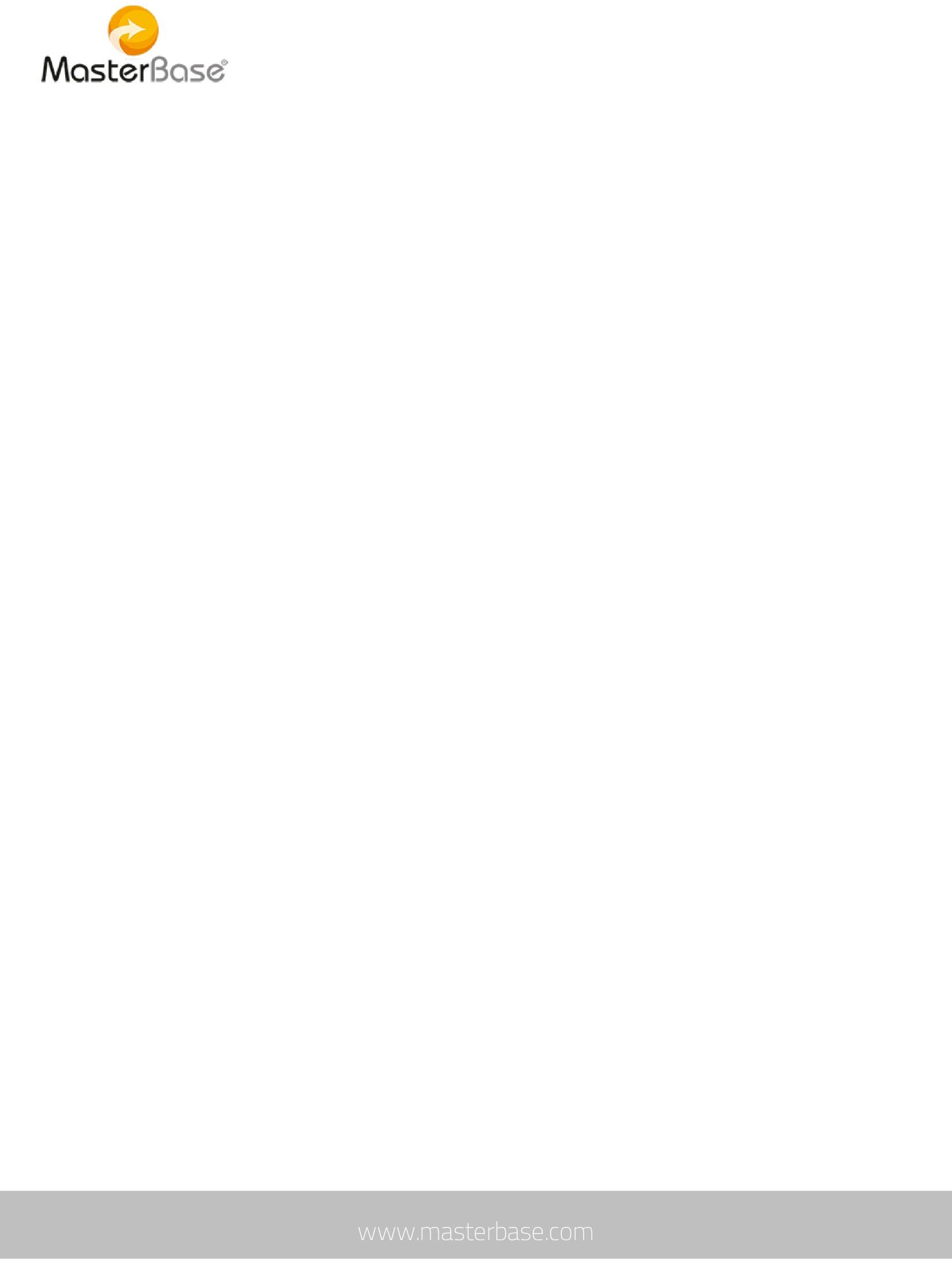
|  |  |
| --- | --- |
| ID del Container | Descripción |
| CFC01 | Contenedor de Sistema – Datos Fuentes de Envíos |
| CFC02 | Contenedor de Sistema – Datos Fuentes de Aperturas |
| CFC03 | Contenedor de Sistema – Datos Fuentes de Clicks |
| CFC04 | Contenedor de Sistema – Datos Fuentes de Rebotes |
| CFC05 | Contenedor de Sistema – Datos Fuentes de Desuscripciones |

*Autómata A1:*

Un autómata A1 es un servicio Windows encargado de procesar los Datos Fuentes almacenados en un Contenedor especifico y en base a la información contenida en estos Datos Fuentes generar un conjunto de Etiquetas de Contacto en un formato predeterminado.

*XML de configuración virtual:*

Se trata de un archivo XML que describe de qué forma un Autómata A1 debe interpretar los Datos Fuentes (en formato XML, JSON o CSV) almacenados en un Contenedor especifico y cómo debe procesar esta información para generar un conjunto de Etiquetas de Contacto predefinidas.



# Generación de XML de configuración virtual.

A continuación, se describe cómo generar un XML de configuración virtual para procesar un Dato Fuente específico, así como también los XML de configuración virtual correspondientes a los Datos Fuentes de sistema.

## Estructura base de configuración virtual

El XML de configuración virtual tiene la siguiente estructura base:

<VirtualAutomataConfig>

<Format Type="{TYPE}" Separator=”{SEPARATOR}”/>

<ContactField CreateContact="{CREATE\_CONTACT}">

<ListId>{LIST\_ID}</ListId>

<Name>{NAME}</Name>

<Column>{COLUMN}</Column>

<Path>{CONTACT\_PATH}</Path>

</ContactField>

<Tags Regexp=”{REGEXP}”>

<Tag Template="{TAG\_TEMPLATE}" Ttl=”{TTL}”>

<Paths>

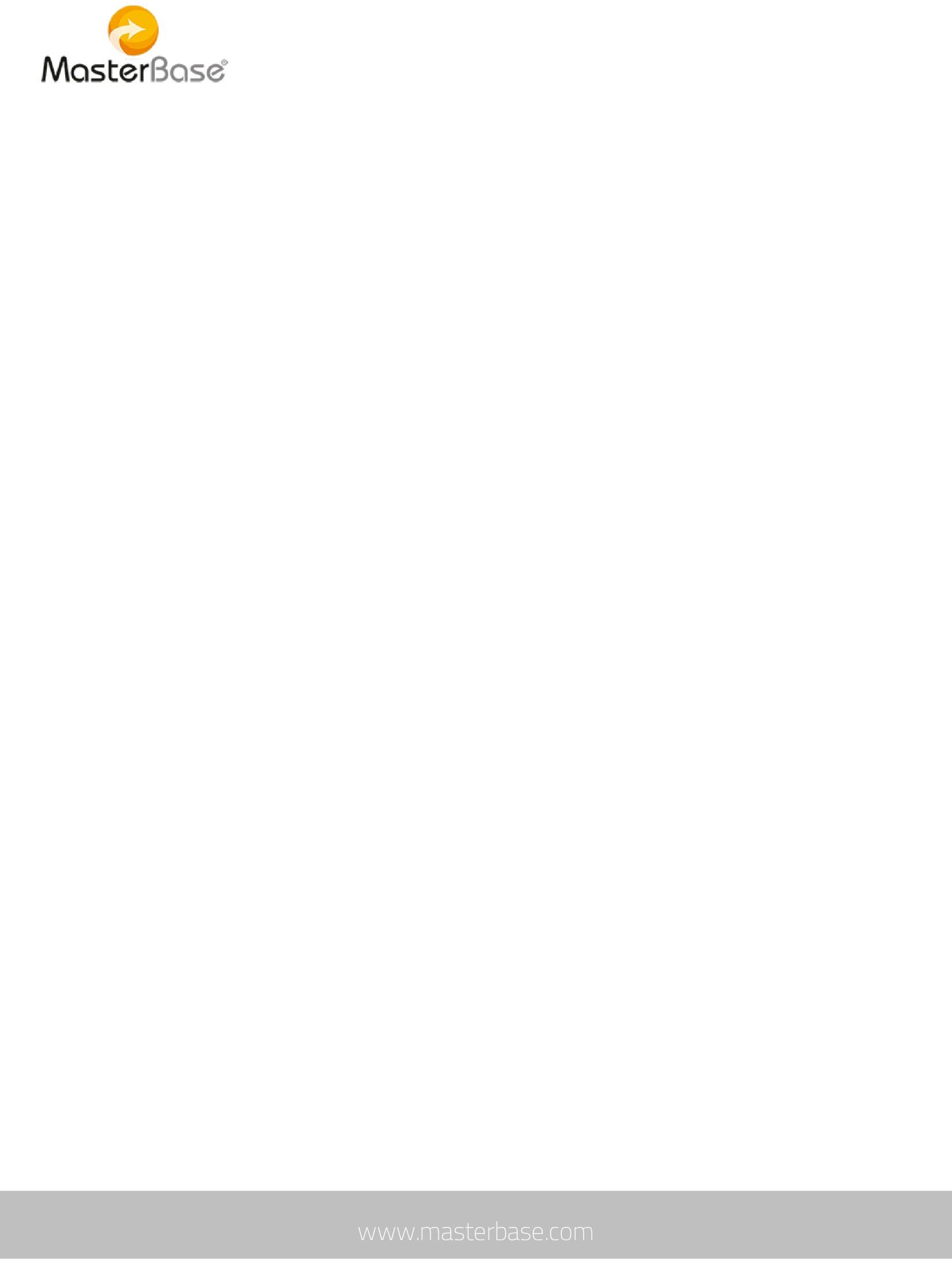
<Path id=”{ID\_PATH}”>{TAG\_PATH}</Path>

</Paths>

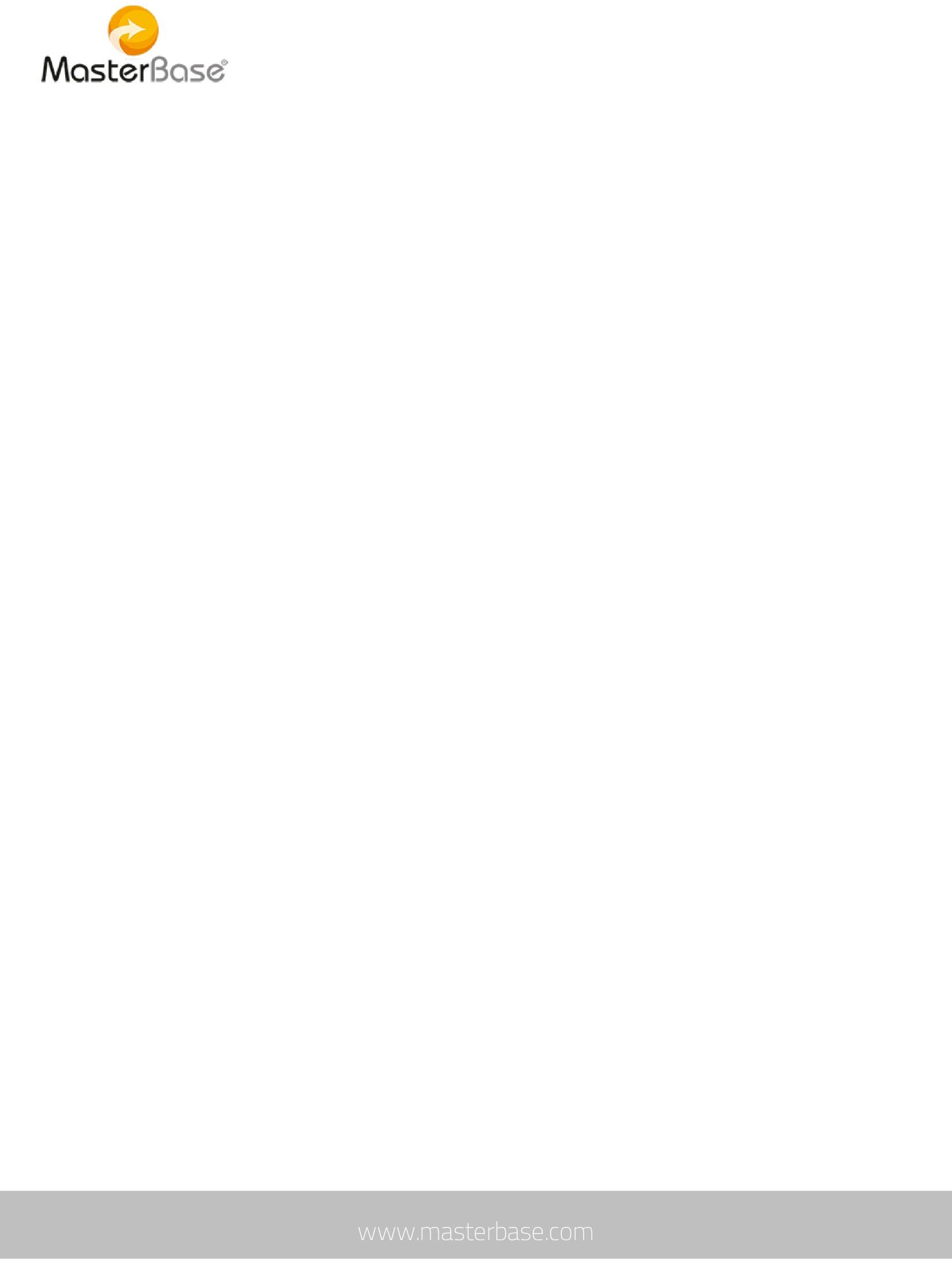
</Tag>

</Tags>

</VirtualAutomataConfig>

La siguiente tabla detalla cada uno de los elementos y atributos a considerar:

|  |  |
| --- | --- |
| Elemento/atributo | Descripción |
| {ID} | Valor obligatorio que corresponde al identificador del autómata  virtual. |
| {CFC} | Valor obligatorio que corresponde al identificador del Contenedor. |
| {TYPE} | Valor obligatorio que corresponde al tipo de formato de archivo del Dato Fuente que va a procesar el autómata. Los posibles valores son  xml, json o csv. |
| {SEPARATOR} | Valor obligatorio cuando {TYPE} corresponde a csv. Define el caracter  que se utiliza como separador de los valores del archivo csv. |
| {CREATE\_CONTACT} | Valor booleano (true o false) que indica si el contacto debe ser creado en caso de no existir. El atributo CreateContact es de carácter  opcional y de no ser colocado su valor por omisión es false. |
| {LIST\_ID} | Valor obligatorio que corresponde al identificador de la lista donde se  encuentra el campo con el que se identificará el registro. |
| {NAME} | Valor obligatorio que identifica el nombre del campo contenido en la  lista especificada con el cual se identificará el registro. |
| {COLUMN} | Valor obligatorio cuando el Type es csv. Identifica la columna del archivo csv de la cual se obtendrá el valor del campo identificador del  registro desde el Dato Fuente. |
| {CONTACT\_PATH} | Valor obligatorio cuando el Type es xml o json. Identifica el path xml  o json del cual se obtendrá el valor del campo identificador del registro desde el Dato Fuente. |
| {REGEXP} | Valor obligatorio cuando el Type es csv. Contiene una expresión  regula que hoy no se utiliza. |
| {TAG\_TEMPLATE} | Valor obligatorio que especifica el formato en el cual se generara el  tag. |
| {TAG\_PATH} | Valor obligatorio cuando Type es xml o json. Identifica el Path xml o json para obtener un valor desde el Dato Fuente que será utilizado para generar un Tag. En formato JSON, el valor {TAG\_PATH} va  antecedido siempre por: “$.” |
| {ID\_PATH} | Valor obligatorio cuando Type es xml o json. Identifica un índice único para el TAG\_PATH especificado. Este índice es utilizado en TAG\_TEMPLATE para generar el Tag con el valor remplazado desde  el Dato Fuente. |
| {TTL} | Valor obligatorio que especifica el tiempo de validez del tag. Se trata de un valor compuesto por un entero mayor a 1 y menor a 36500 seguido de una letra minúscula (m, h, d) que identifica minutos, horas,  días respectivamente. No debe incluir espacios. |



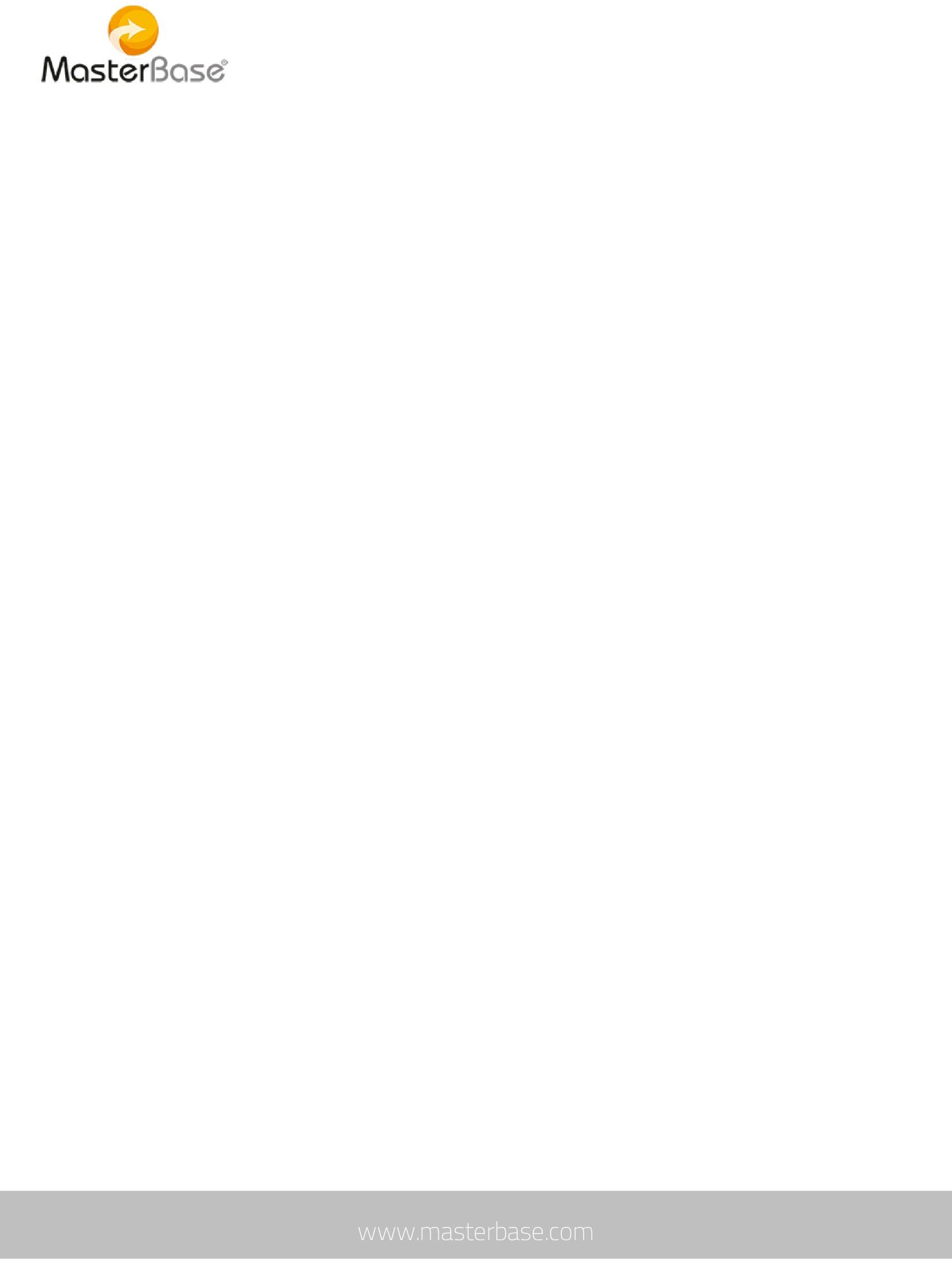
## Ejemplos de XML de configuración virtual

A continuación, se detallan 3 configuraciones virtuales basadas en Datos Fuentes de diferente tipo (XML, JSON y CSV), para la generación de Etiquetas de Contacto específicas. Esta configuración virtual se basa en la estructura de Datos Fuentes almacenados en 3 diferentes Contenedores, simulando el ejemplo de una Autopista registrando el detalle de patentes que pasan bajo los pórticos de cobro.

La siguiente tabla detalla los Contenedores y los formatos de los Datos Fuentes almacenados en cada uno de ellos que se utilizarán para el ejemplo:

|  |  |  |
| --- | --- | --- |
| Id de Container | Tipo de  Feed | Tags a Generar |
| CFC06 | XML | Pórtico/{Patente} con expiración en 7 días. |
| CFC07 | JSON | Pórtico/{Patente} con expiración en 7 días. |
| CFC08 | CSV | Pórtico/{Patente} con expiración en 7 días. |

A continuación, la estructura de los Datos Fuentes almacenados en cada uno de los Contenedores detallados anteriormente



CFC06: (Formato: XML)

<Feeds>

<Feed>

<Fecha>01/01/2016</Fecha>

<Portico>38</Portico>

<Patente>WF0101</Patente>

</Feed>><Feed>

<Fecha>01/01/2016</Fecha>

<Portico>38</Portico>

<Patente>WF0101</Patente>

</Feed>><Feed>

<Fecha>01/01/2016</Fecha>

<Portico>38</Portico>

<Patente>WF0101</Patente>

</Feed>

</Feeds>

Configuración AV1 para CFC06: (XML)

<VirtualAutomataConfig>

<Format Type="xml" />

<ContactField CreateContact=”true”>

<ListId>1</ListId>

<Name>Patente</Name>

<Path>/Feeds/Feed/Patente</Path>

</ContactField>

<Tags>

<Tag Template="/portico/{1}" Ttl=”7d”>

<Paths>

<Path id=”1”> /Feeds/Feed/@Patente</Path>

</Paths>

</Tag>

</Tags>

</VirtualAutomataConfig>

CFC07: (Formato: JSON)

{

"Feeds": {

"Feed": [{

}, {

}, {

}

"Fecha": "01/01/2016",

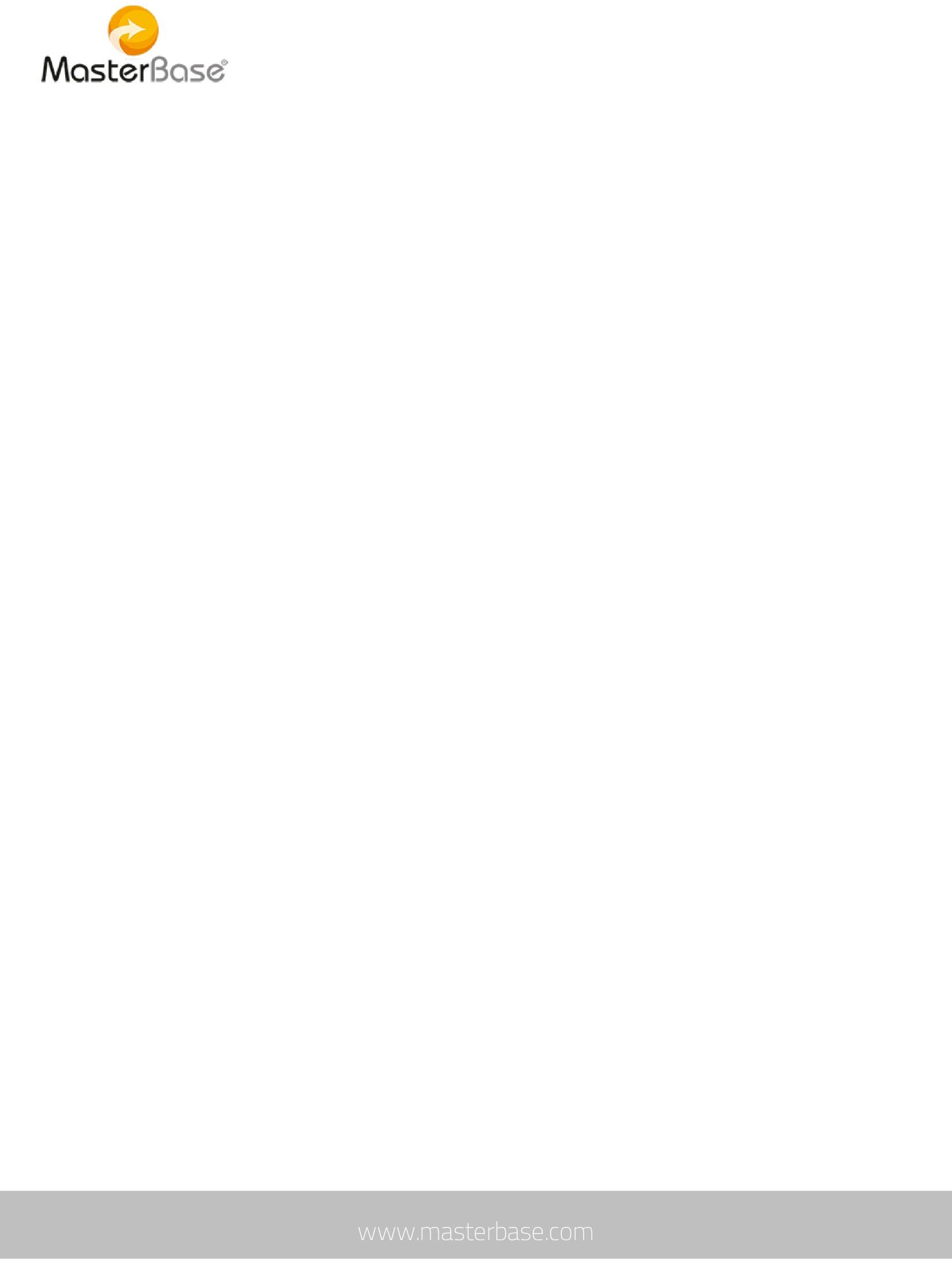
"Portico": "38", "Patente": "WF0101"

"Fecha": "01/01/2016",

"Portico": "38", "Patente": "WF0101"

"Fecha": "01/01/2016",

"Portico": "38", "Patente": "WF0101"

]

}

}

Configuración AV1 para CFC07: (JSON)

<VirtualAutomataConfig>

<Format Type="json" />

<ContactField CreateContact=”true”>

<ListId>1</ListId>

<Name>Patente</Name>

<Path>$.Feeds.Feed.Patente</Path>

</ContactField>

<Tags>

<Tag Template="/portico/{1}" Ttl=”7d”>

<Paths>

<Path id=”1”> $.Feeds.Feed.Patente</Path>

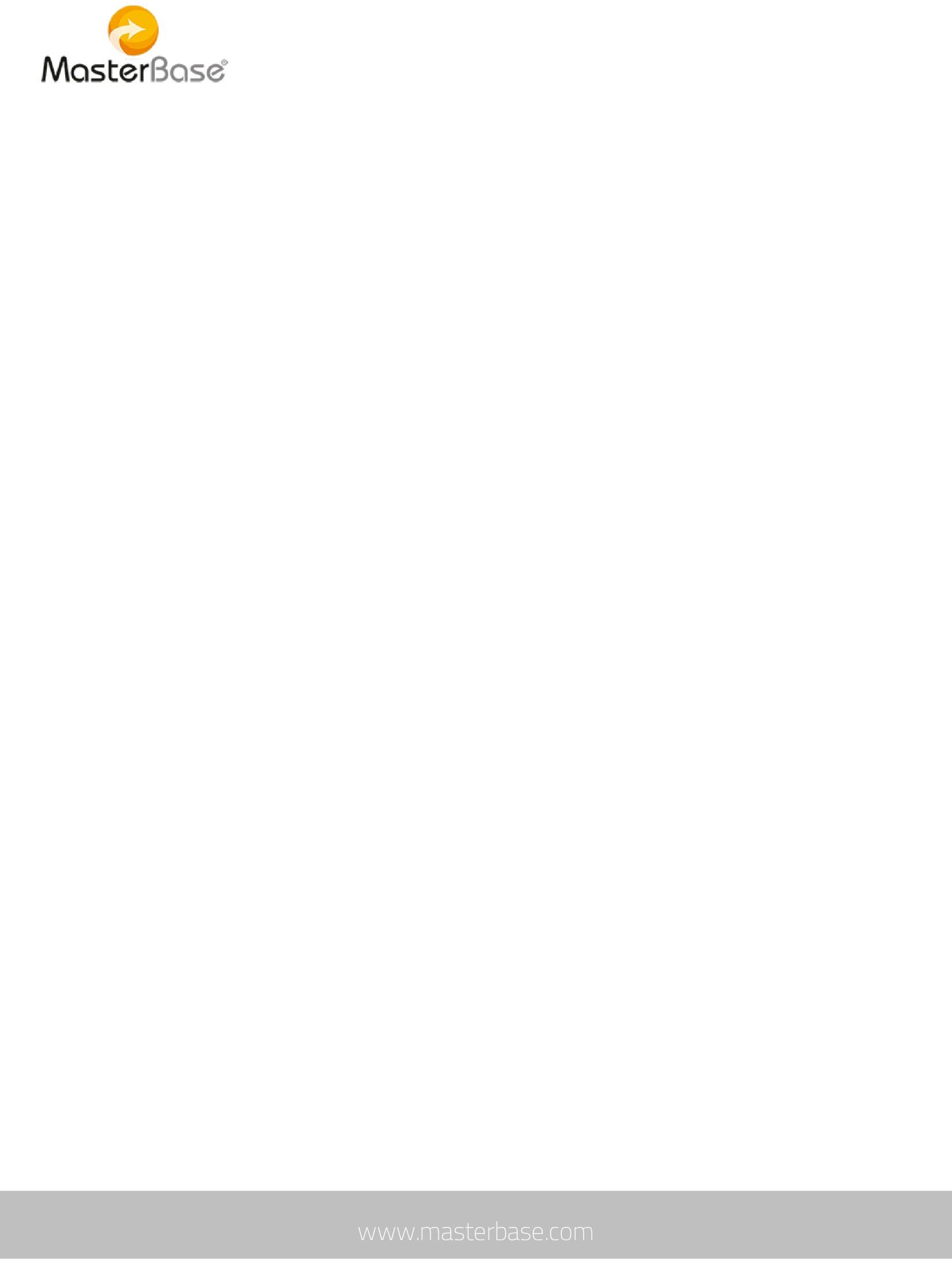
</Paths>

</Tag>

</Tags>

</VirtualAutomataConfig>

En formato JSON, el valor {TAG\_PATH} va antecedido siempre por: “$.”



CFC08: (Formato: CSV) 01/01/2016, 38, WF0101,

01/01/2016, 38, WF0101,

01/01/2016, 38, WF0101,

01/01/2016, 38, WF0101,

Configuración AV1 para CFC08: (CSV)

<VirtualAutomataConfig>

<Format Type="csv" />

<ContactField CreateContact=”true”>

<ListId>1</ListId>

<Name>Patente</Name>

<Colum>3</Column>

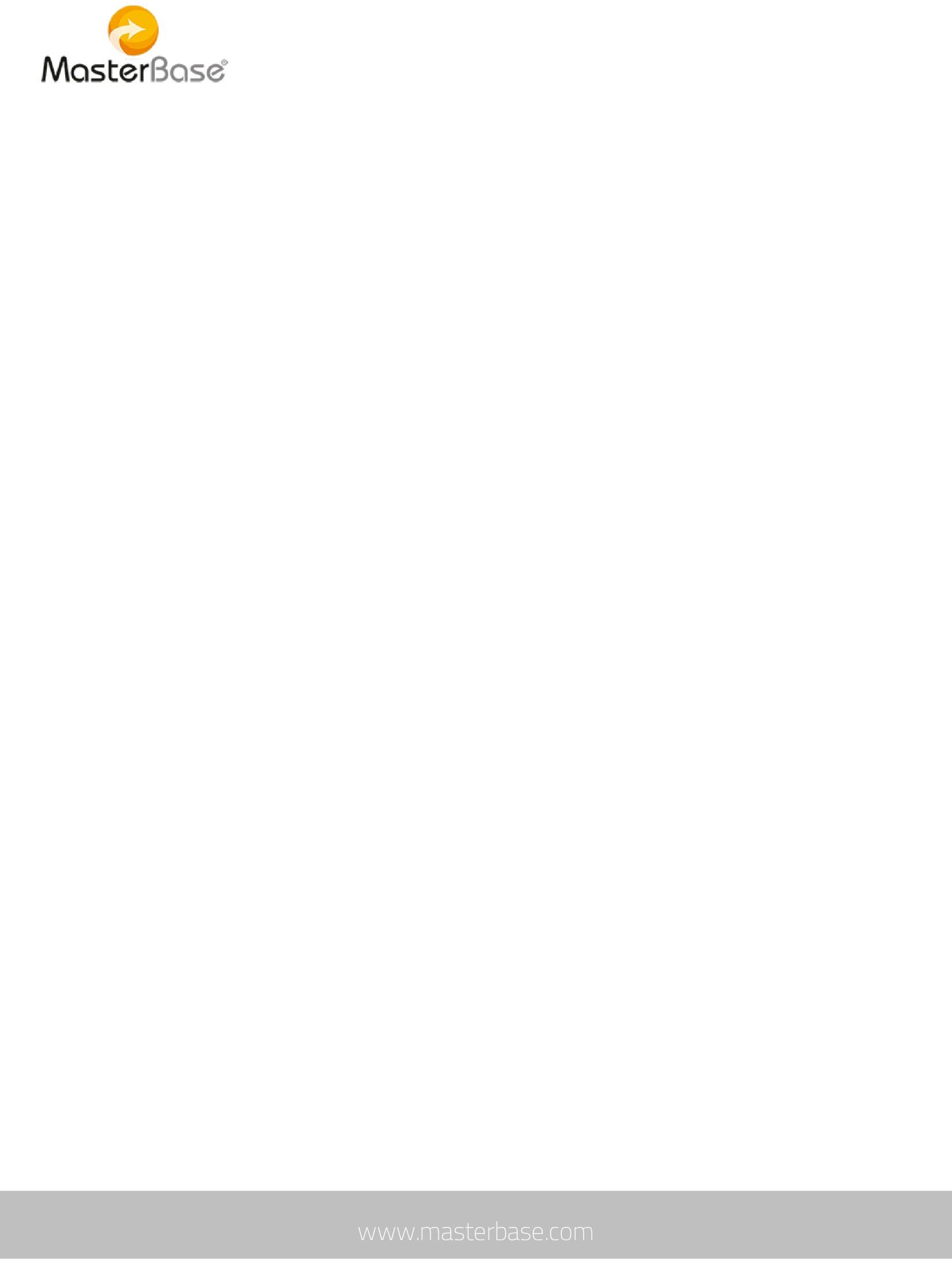
</ContactField>

<Tags Regexp=”.\*”>

<Tag Template="/portico/{3}" Ttl=”7d”/>

</Tags>

</VirtualAutomataConfig>



## Consideraciones especiales:

#### Variables con espacios

Una variable, en una estructura JSON, podría venir con espacios, frente a esto casos el elemento

{TAG\_PATH} debe contener una estructura especial para que sea procesable por el autómata Virutal, esta estructura es la siguiente: $.[‘variable con espacio’]

Ejemplo de Dato fuente en el CFC (Variable con espacios en naranjo):

{

"Rut": "11111111-1",

"Facultad": "Humanidades", "area disciplinaria": "Derecho",

"programa": "Magíster en Planificación y Dirección Tributaria (Santiago)", "sede": "",

"modalidad": "",

"Nombre": "Juan",

"Sexo": "Masculino", "Nacionalidad": "CL",

"mail": "[jcalvo@masterbase.com"](mailto:jcalvo@masterbase.com)

}

Elemento en el XML de Configuración Virtual (destacado en naranjo):

<VirtualAutomataConfig>

<Format Type="json" />

<ContactField>

<ListId>1</ListId>

<Name>Email</Name>

<Path>mail</Path>

</ContactField>

<Tags>

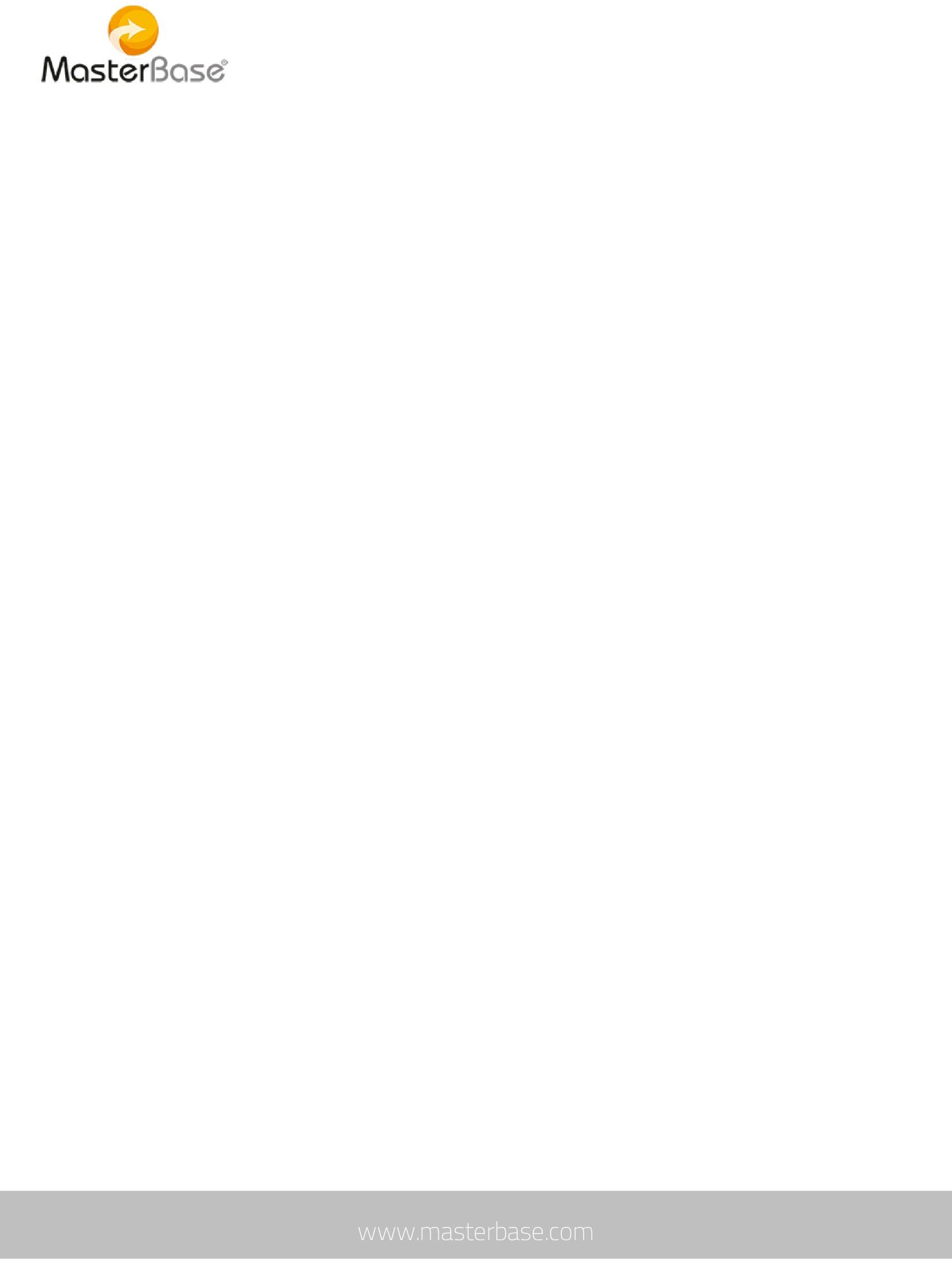
<Tag Template="/MATRICULADOS/AREA DISC/{1}" Ttl="720d">

<Paths>

<Path id="1">$.['area disciplinaria']</Path>

</Paths>

</Tag>



<Tag Template="/MATRICULADOS/PROGRAMA/{2}" Ttl="720d">

<Paths>

<Path id="2">programa</Path>

</Paths>

</Tag>

<Tag Template="/MATRICULADOS/SEDE/{3}" Ttl="720d">

<Paths>

<Path id="3">$.sede</Path>

</Paths>

</Tag>

<Tag Template="/MATRICULADOS/MODALIDAD/{4}" Ttl="720d">

<Paths>

<Path id="4">$.modalidad</Path>

</Paths>

</Tag>

</Tags>

</VirtualAutomataConfig>

#### Multitags

El elemento Tags puede contener uno o más elementos Tag para la generación de múltiples Tags basados en la información contenida en el Dato Fuente.

<Tags>

<Tag Template="/Ejemplo1/{1}" Ttl="720d">

<Paths>

<Path id="1">$.Ejemplo1</Path>

</Paths>

</Tag>

<Tag Template="/Ejemplo2/{2}" Ttl="720d">

<Paths>

<Path id="2">$.Ejemplo2</Path>

</Paths>

</Tag>

<Tag Template="/Ejemplo3/{3}" Ttl="720d">

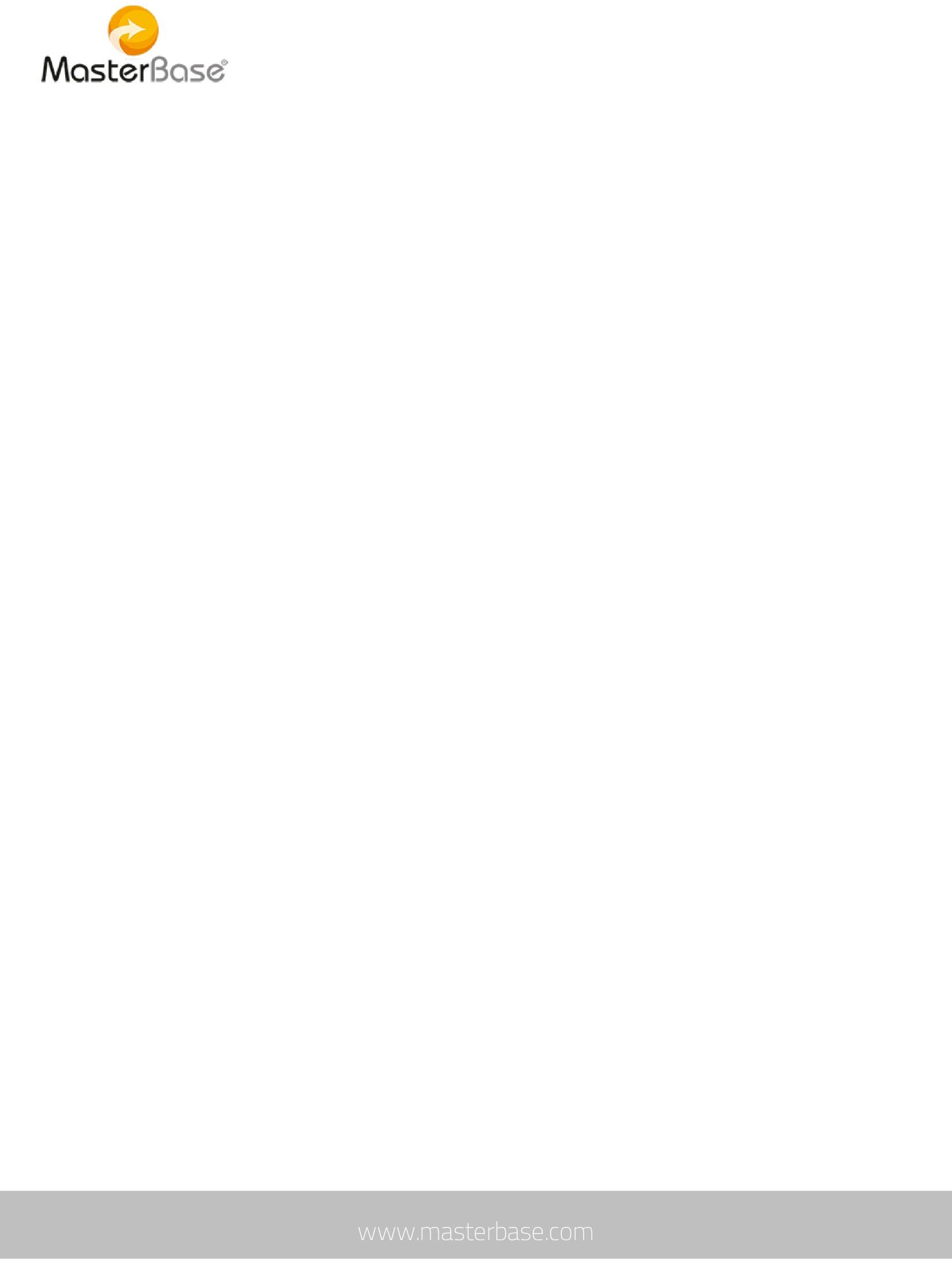
<Paths>

<Path id="3">$.Ejemplo3</Path>

</Paths>

</Tag>

</Tags>



#### Estructura Dato Fuente

Antes de poder generar un archivo de configuración virtual es necesario identificar el Contenedor que contiene los Datos Fuentes que se quieren procesar, así como la estructura especifica de estos Datos Fuentes. Si no se conoce la estructura especifica del Dato Fuente no es posible generar un archivo de configuración virtual para que procese esta información.

#### Respetar la estructura jerárquica del dato fuente

En ocasiones, el dato fuente puede tener varios niveles jerárquicos, lo cual debe considerarse a la hora de confiogurar el XML del correspondiente AV. En el siguiente ejemplo podemos ver un dato fuente en formato JSON el cual es obtenido desde WebBehavior capturando los datos de un sitio web. Podran ver en negro los datos asociados al primer nivel de datos y en “azul” los datos asociados a un segundo nivel de datos contenidos dentro de “BehaviorData”:

{

"IdDevice": "acecc15c727c45adb371bf2289adaf55", "IdClient": "5121",

"IdContacto": "8944bc4463cb4898b99ecd7df18bd519", "IdContainer": "CFC11",

"Email": "",

"IdMail": "",

"ClientURI": "<http://marketingdigitalgroup.com/umayor/postgrado/>", "ClientDomain": "marketingdigitalgroup.com",

"BehaviorData": {

"Ip": "191.102.32.14",

"CountryCode": "CL",

"title": "Buscador Postgrados - Universidad",

"description": "El Buscador Postgrado te permite encontrar, de manera fácil, cuál de los programas de la Universidad es el que tú necesitas.",

"keywords": "postgrados, universidad, comunicación, arquitectura, arte, biotecnología, salud, ingeniería, derecho, educación",

"viewport": "width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0",

"sede": "Temuco",

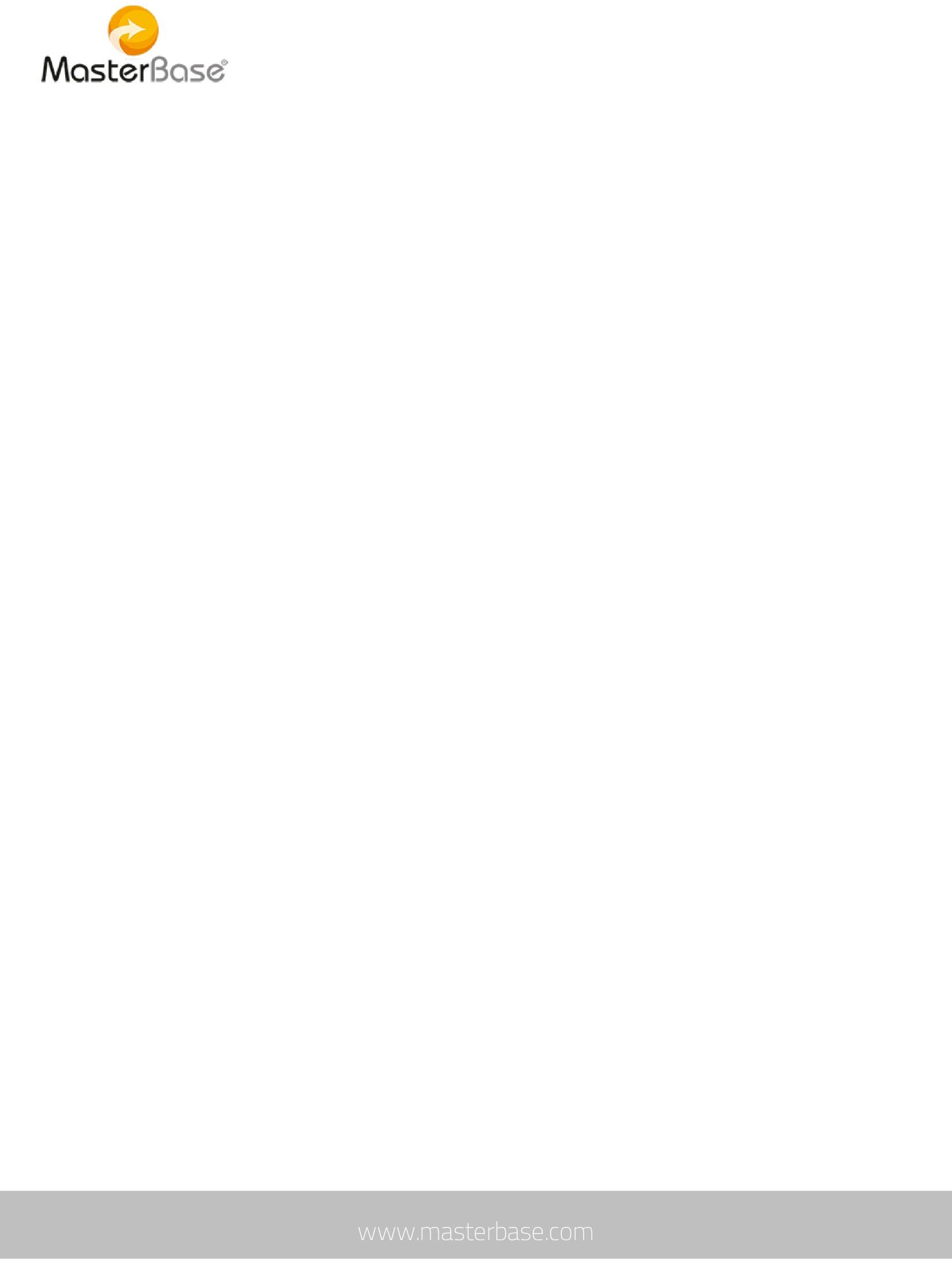
"area": "Derecho", "subarea": "Derecho",

"tipoprograma": "Diplomado",

"programa": "Diplomado en Derecho del Trabajo", "rut": "15642544-3",

"nombres": "Juan",

"apellidos": "Calvo", "email": "[nombre.apellido@mail.com"](mailto:nombre.apellido@mail.com), "celular": "988378209", "MobileDevice": "False",



"Browser": "Chrome",

"OS": "Windows 10"

}

}

En este caso, el correspondiente XML de configuración debe tener la siguiente estructura. Destacamos el campo “email” ya que será la llave en el ejemplo:

<VirtualAutomataConfig>

<Format Type="json" />

<ContactField CreateContact="true">

<ListId>1</ListId>

<Name>Email</Name>

<Path>$.BehaviorData.email</Path>

</ContactField>

<Tags>

<Tag Template="WEB/BUSCADOR POSTGRADO/SEDE/{1}" Ttl="90d">

<Paths>

<Path id="1">$.BehaviorData.sede</Path>

</Paths>

</Tag>

<Tag Template="WEB/BUSCADOR POSTGRADO/AREA/{2}" Ttl="90d">

<Paths>

<Path id="2">$.BehaviorData.area</Path>

</Paths>

</Tag>

<Tag Template="WEB/BUSCADOR POSTGRADO/SUBAREA/{3}" Ttl="90d">

<Paths>

<Path id="3">$.BehaviorData.subarea</Path>

</Paths>

</Tag>

<Tag Template="WEB/BUSCADOR POSTGRADO/TIPO PROGRAMA/{4}" Ttl="90d">

<Paths>

<Path id="4">$.BehaviorData.tipoprograma</Path>

</Paths>

</Tag>

<Tag Template="WEB/BUSCADOR POSTGRADO/PROGRAMA/{5}" Ttl="90d">

<Paths>

<Path id="5">$.BehaviorData.programa</Path>

</Paths>

</Tag>

</Tags>

</VirtualAutomataConfig>

En donde $. Se asigna sólo al comienzo para identificar la Raíz de la estructura, luego del “.” se indica el “objeto” o llave del nivel correspondiente y así sucesivamente dependiendo los niveles que tenga la estructura.

#### Respetar Mayúsculas y minúsculas:

Para evitar errores, se debe respetar las mayúsculas y minúsculas según como vengan en el dato fuente.

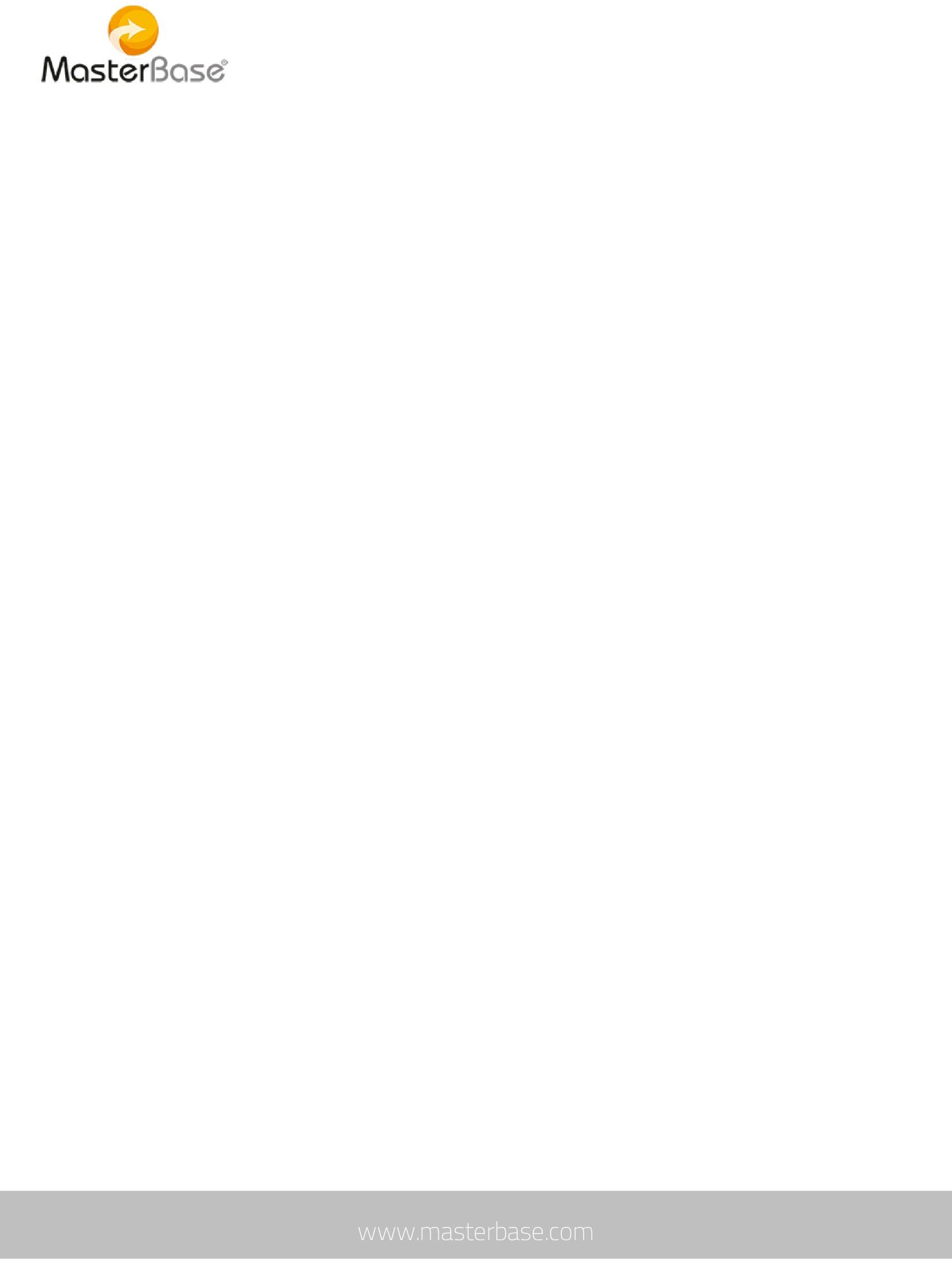
#### Etiquetas con varias dimensiones asociadas

Una plantilla de etiqueta puede contener hasta 10 dimensiones, las cuales puedes estar asociadas a partir de la información capturada en el dato fuente, en ese caso la configuración del autómata virtual debe considerar la siguiente estructura:

</Tag>

<Tag Template="/WEB/{6}/{7}/{8}" Ttl="70d">

<Paths>



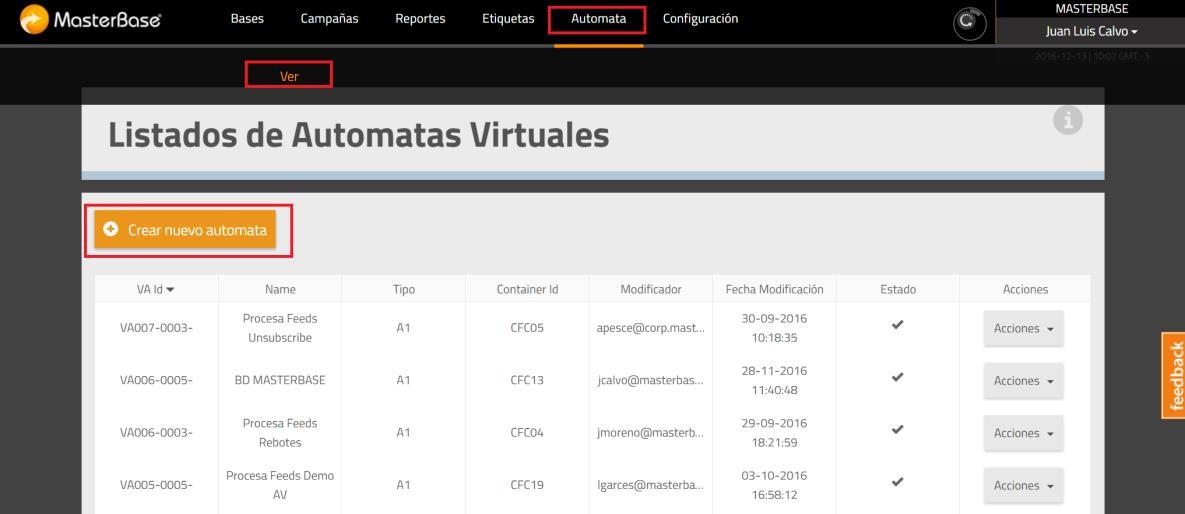
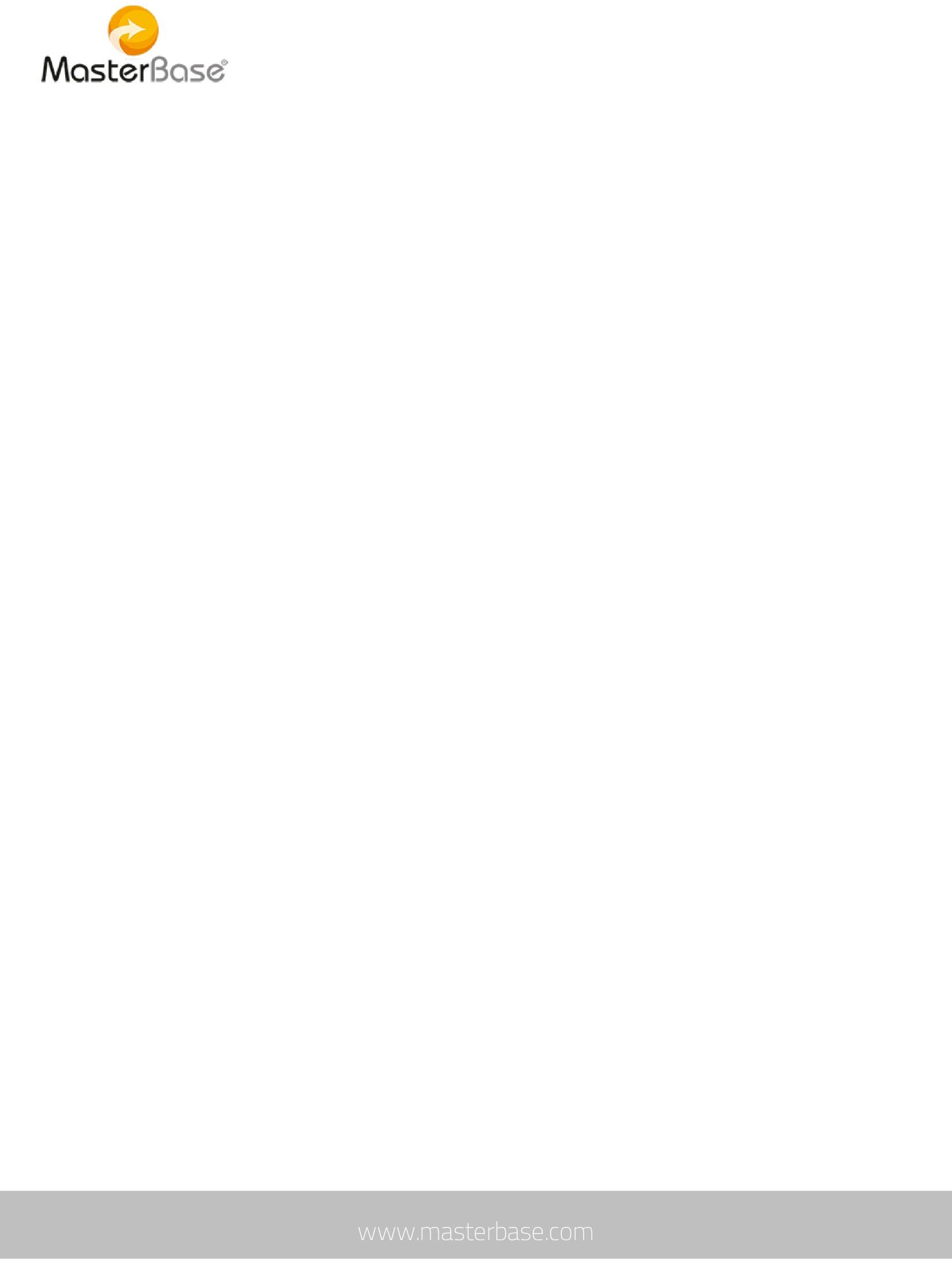
</Paths>

</Tag>

<Path id="6">$.BehaviorData.subjetc</Path>

<Path id="7">$.BehaviorData.CountryCode</Path>

<Path id="8">$.BehaviorData.Location</Path>



## Creación de un Autómata Virtual desde la interfaz web:

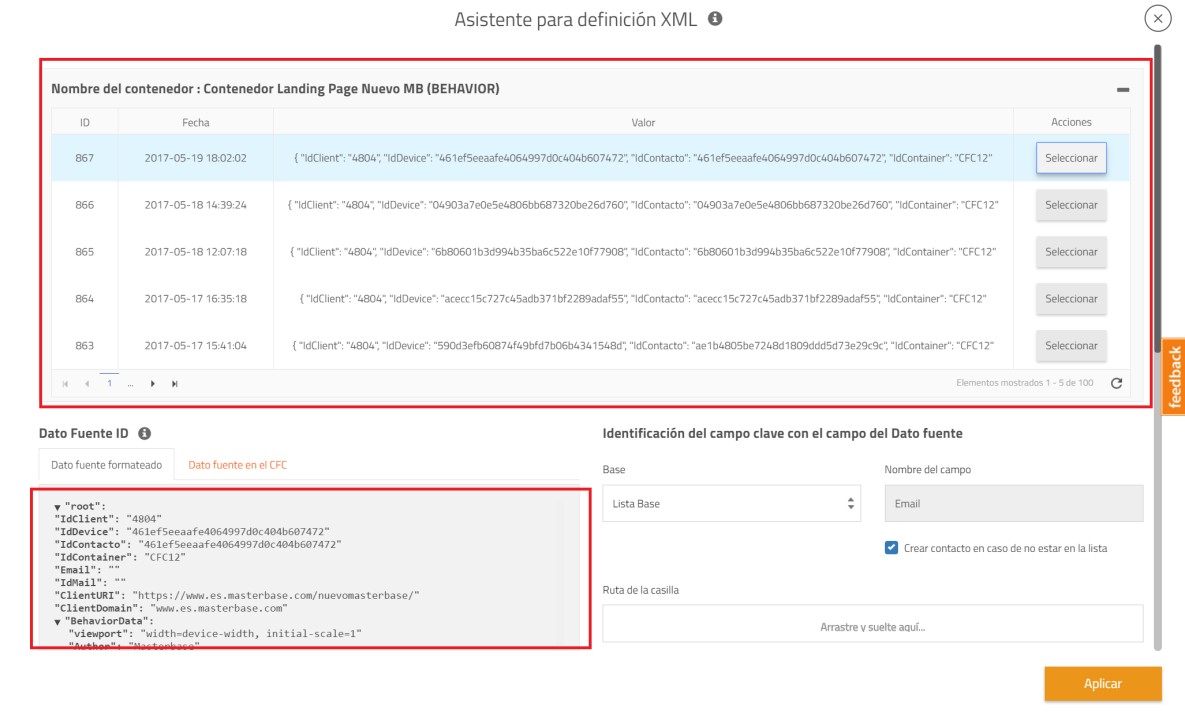
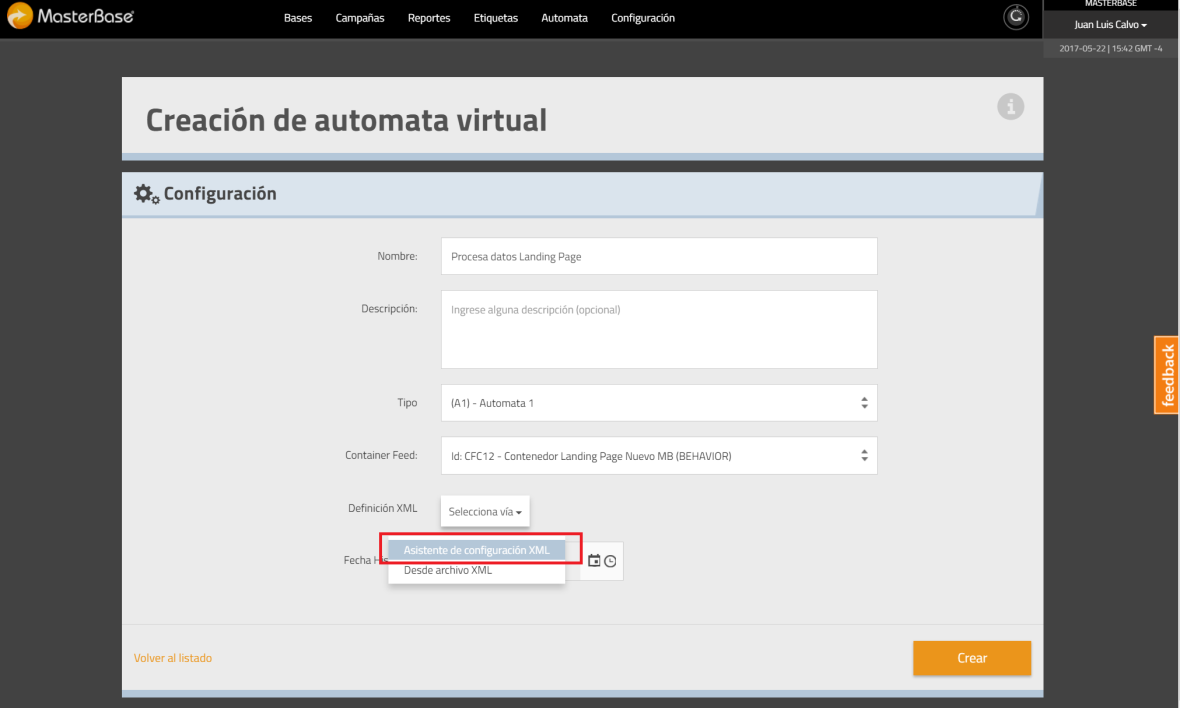
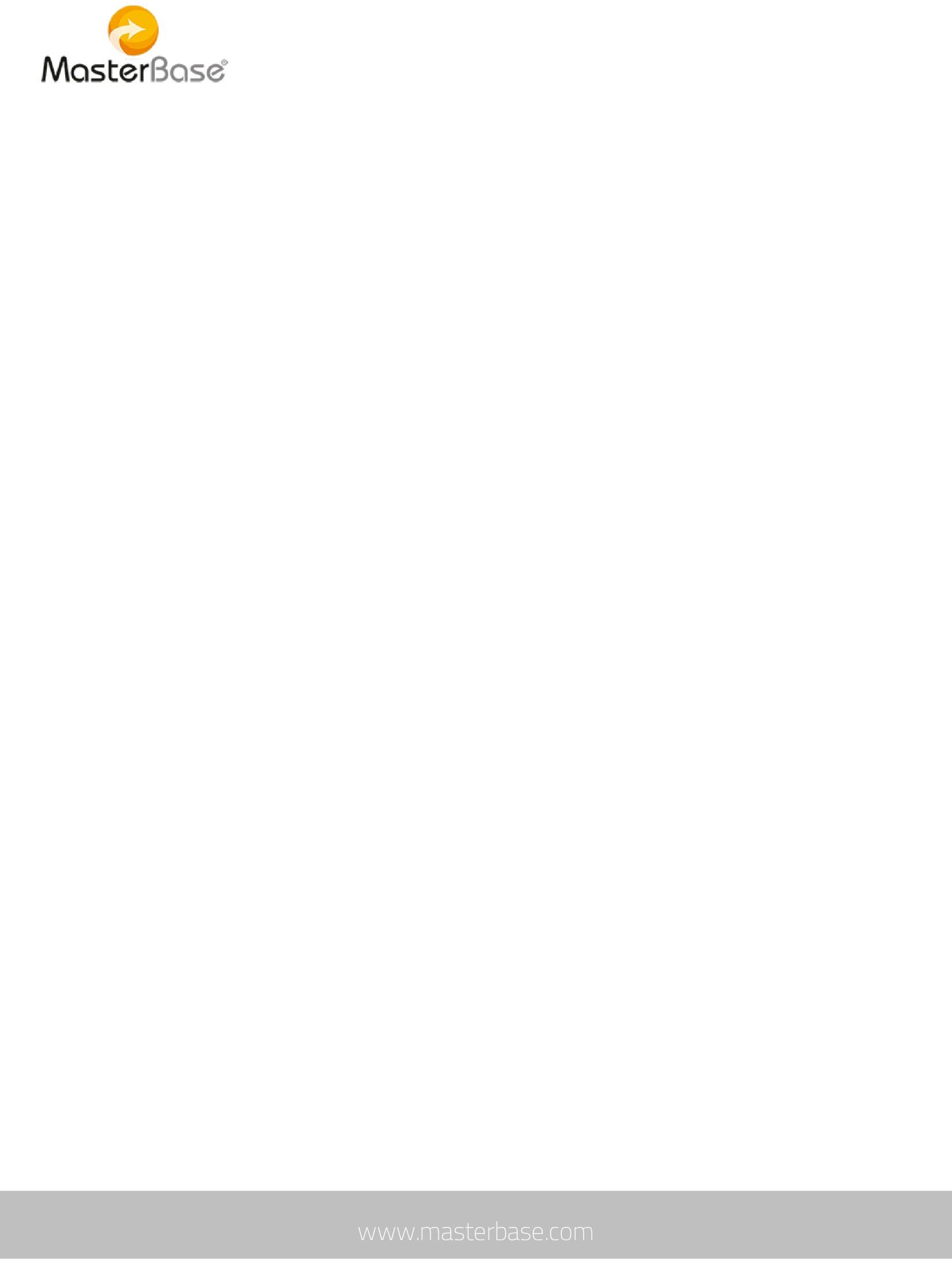
### Existen 2 formas de trabajar la configuración de un autómata virtual desde la interfaz web:

1. Creación asistida de XML de configuración de Autómata virtual:

Al ingresar al módulo de Automata/Ver, accedemos al listado de los Automatas

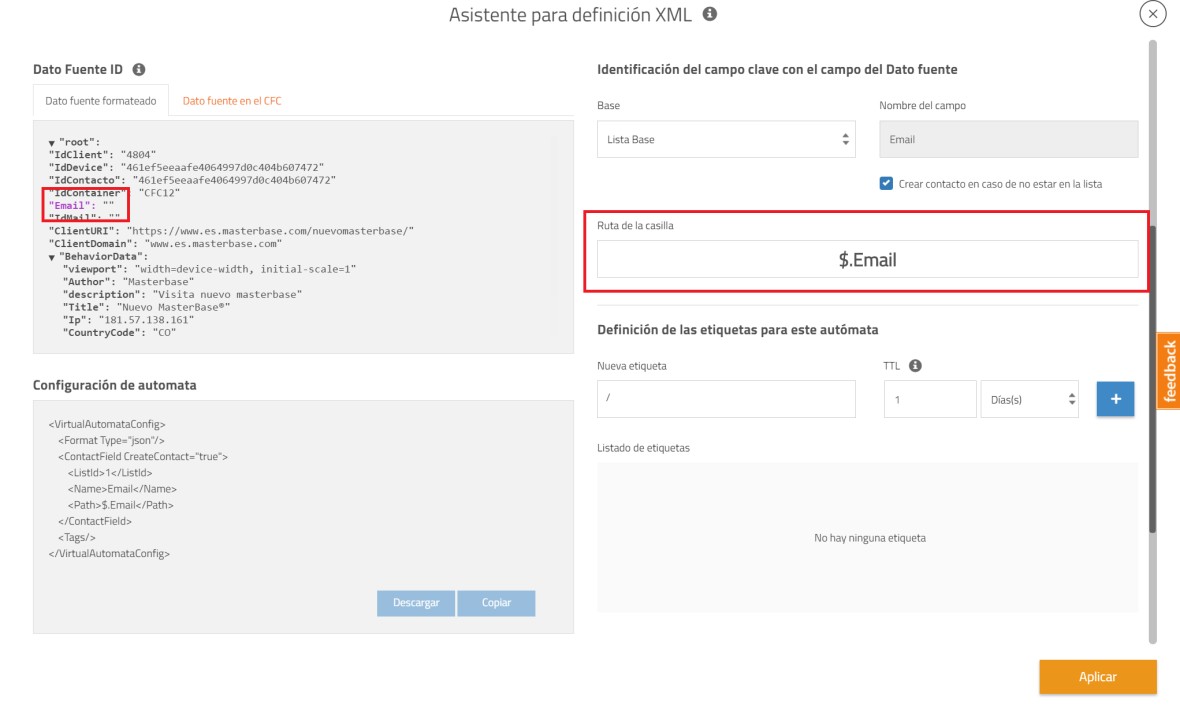
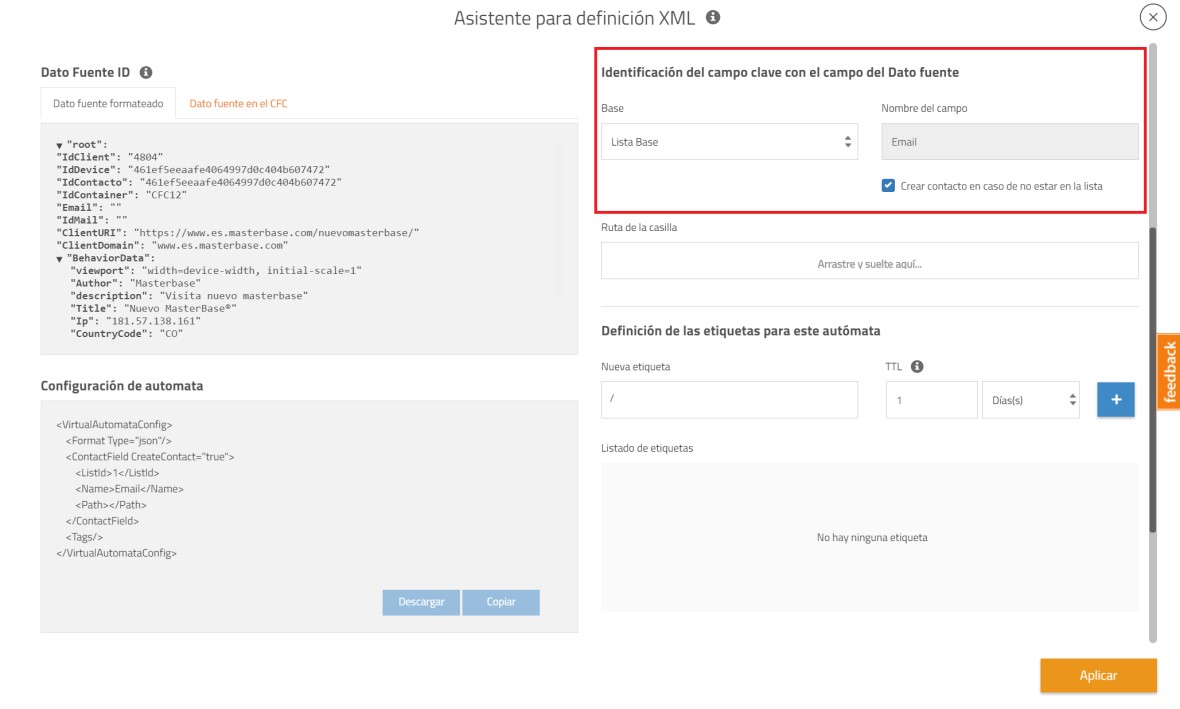
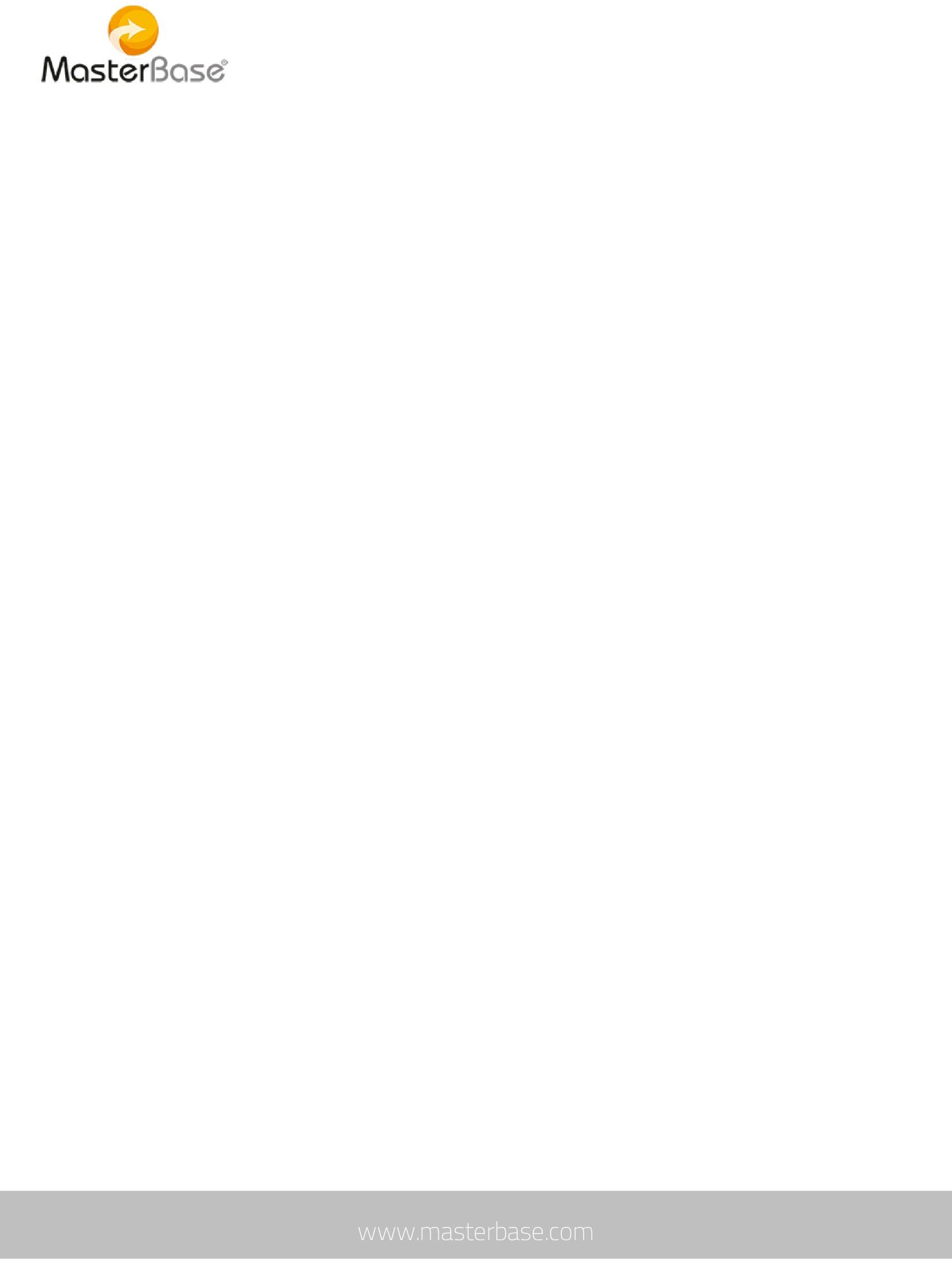
Virtuales que existen. En este módulo hacer clic en la opción “Crear nuevo autómata”:

Se abrirá el módulo de creación de un autómata virtual, en donde se deben completar los datos básicos indicados y en la opción “Definición XML” seleccionar “Asistente de configuración XML”:



Esta opción abrirá un cuadro de configuración asistido en el cual en un primer lugar encontrará una muestra de los datos fuentes acumulados en el contenedor definido, el cual puede ser seleccionado y se verá en un cuadro inferior con mayor detalle.

Basándonos en esta muestra de datos fuentes, podrá generar la configuración del autómata virtual en base a los conceptos entregados en el presente documento. En orden de configuración se debe definir:



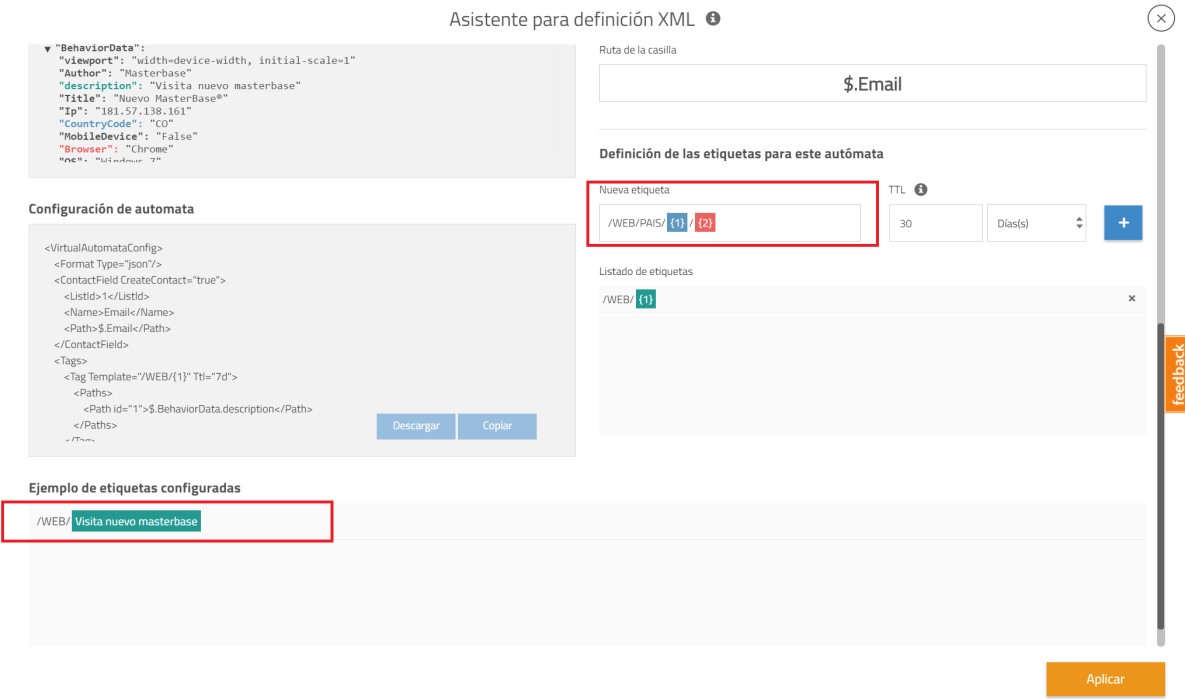
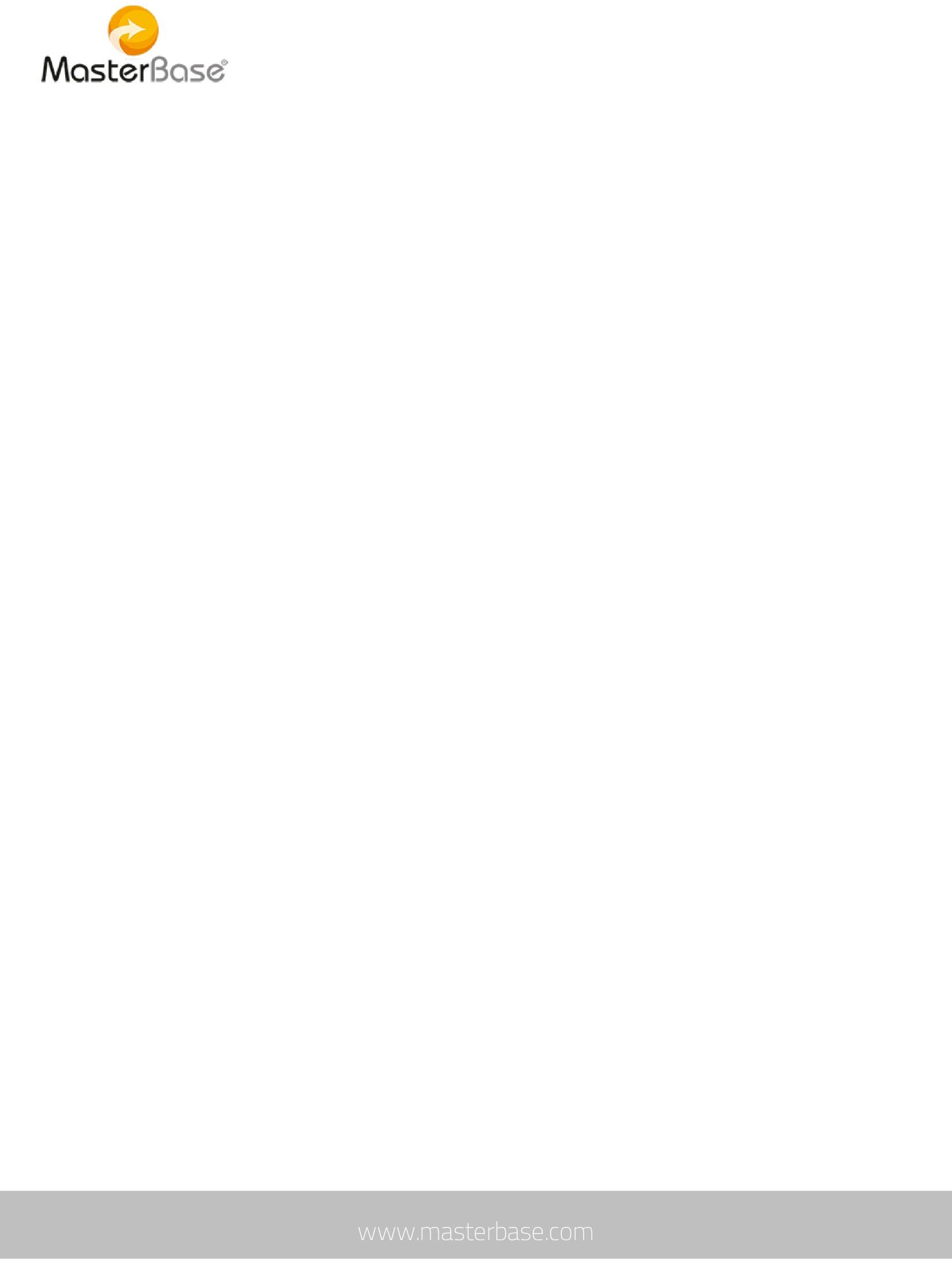
Por defecto se selecciona la lista base y el campo llave es el Email, se considera por defecto la opción de crear el contacto en caso de que no exista en la lista base.

Luego, a través del método “arrastrar y soltar” se debe seleccionar el elemento del dato

fuente que será validado con el campo llave definido, en este caso el campo “Email”:

Luego, se prosigue a construir las “etiquetas” que se busca generar a partir de los datos fuentes obtenidos, para lo cual, se seleccionan el o los elementos del dato fuente y se arrastran hacia el cuadro “Nueva etiqueta”, cabe recordar que una etiqueta puede tener

hasta 10 niveles en su composición. A continuación, se define el concepto de TTL, que representa la vigencia que tendrá esa etiqueta en particular a partir de que sea asociada a un contacto.



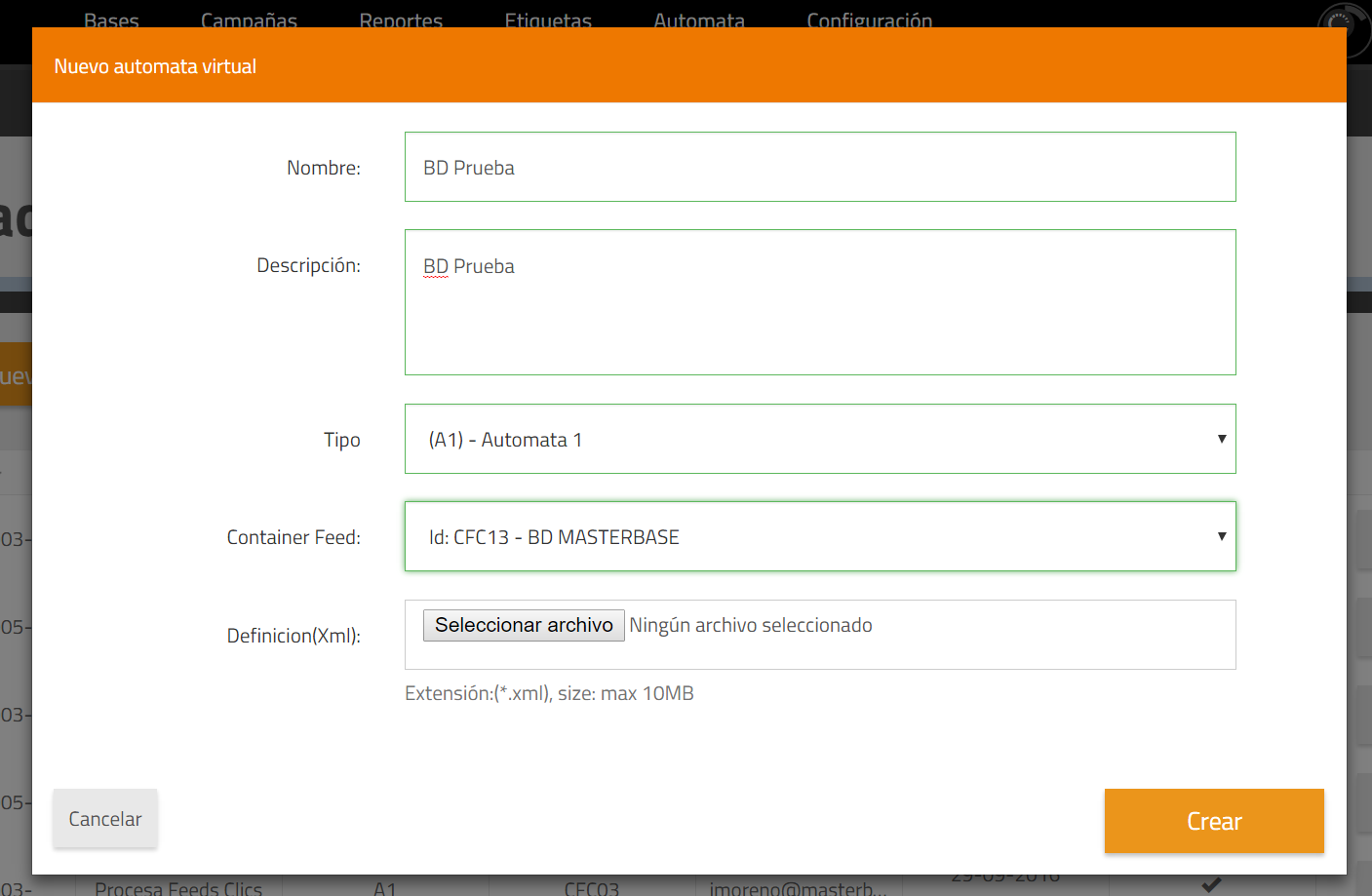
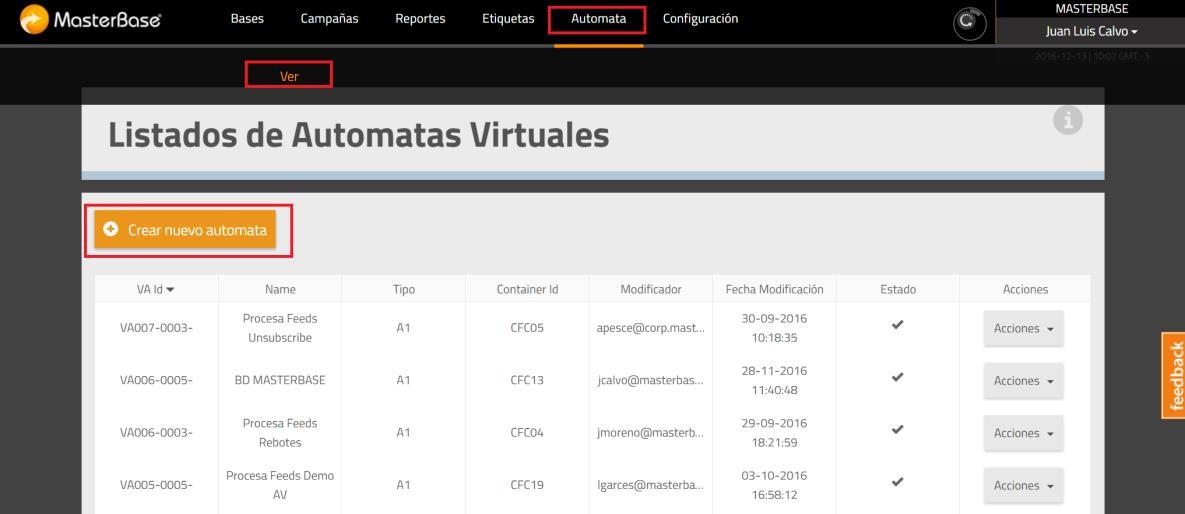
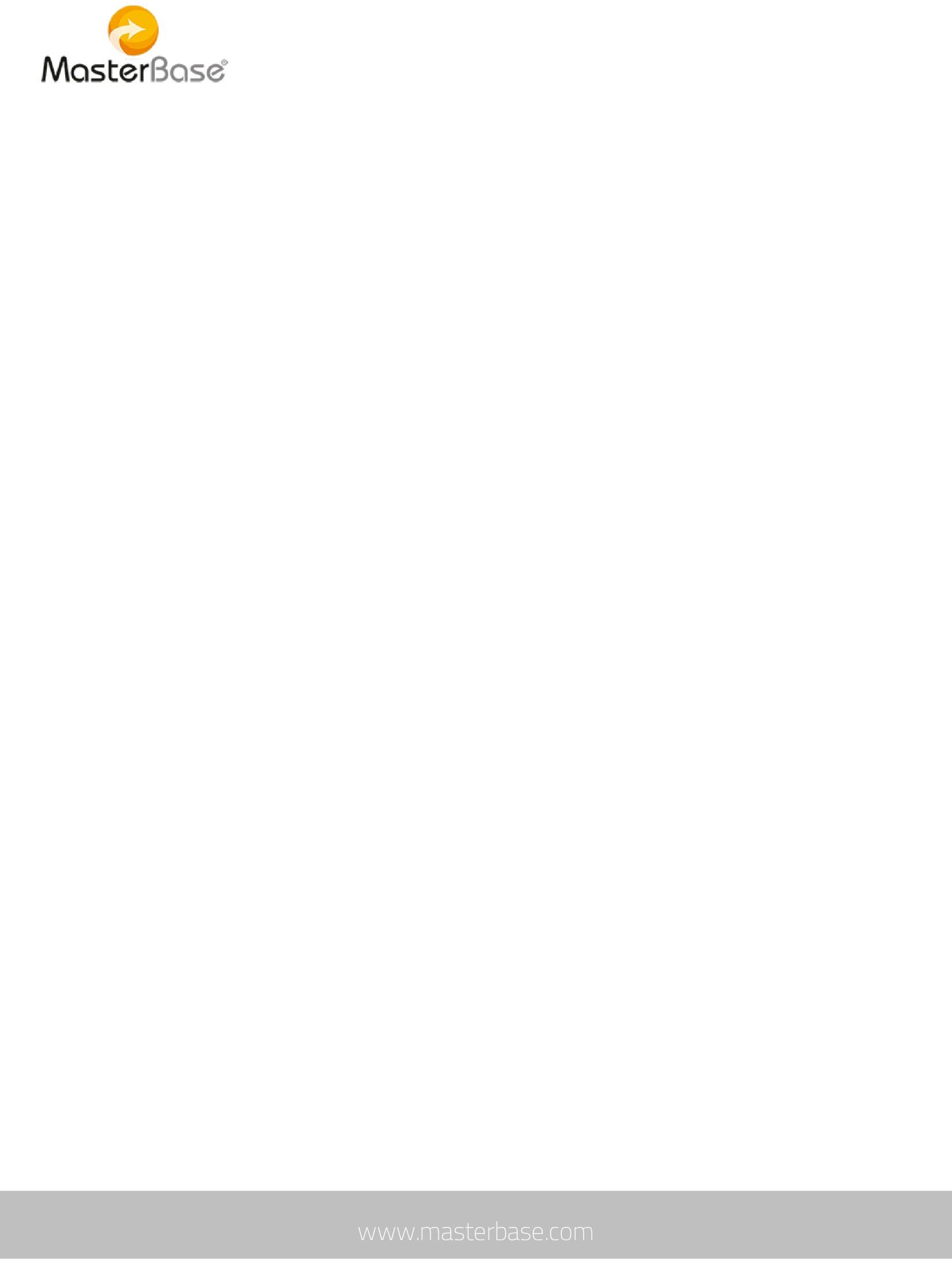
A medida que se vayan creando nuevas etiquetas, estas se irán mostrando en el cuadro “Listado de etiquetas”, además, se podrán revisar ejemplos reales a partir del dato fuente seleccionado en el cuadro “Ejemplo de etiquetas configuradas.

Una vez que se hayan creado todas las etiquetas deseadas se puede guardar la

configuración haciendo clic en “Aplicar”.

1. Configuración externa del XML y subida a través de la interfaz:

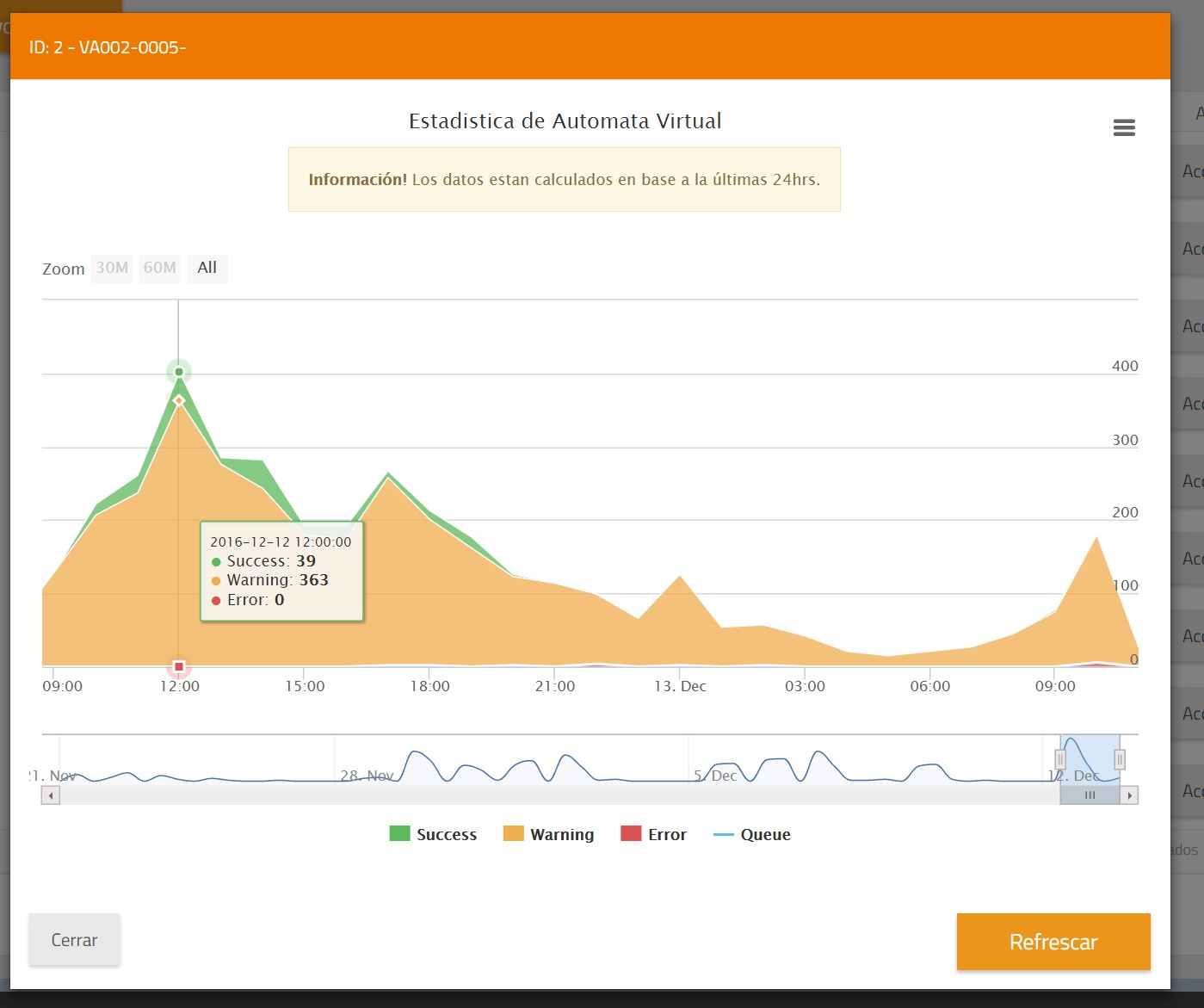
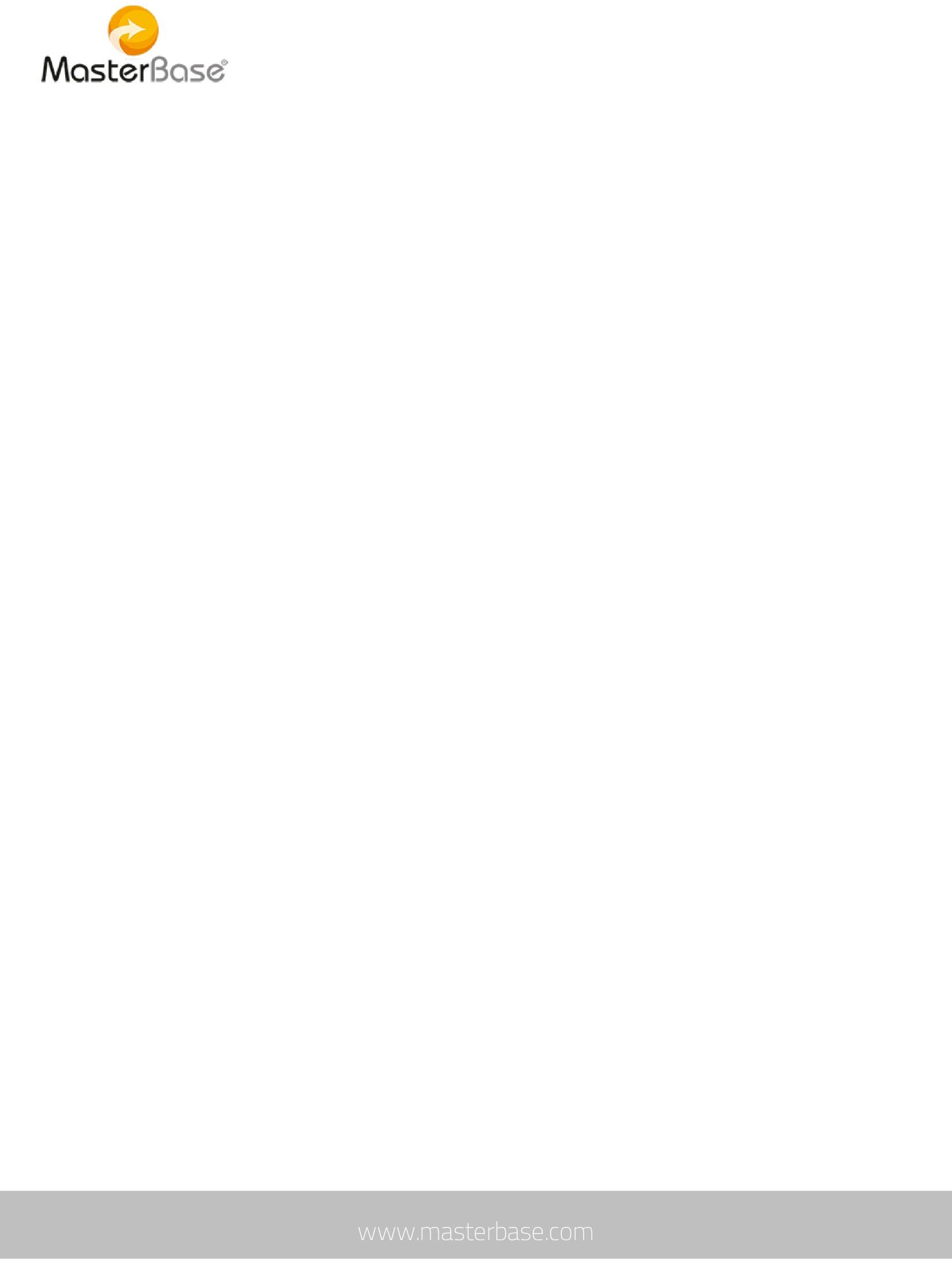
Quienes gusten crear el XML de configuración en alguna herramienta externa lo pueden hacer sin problemas y luego pueden cargarla en el sistema, para esto debe ir a la interfaz web para crear el nuevo Autómata Virtual. Para esto accedemos en el menú principal a la opción “Autómata”, luego “Ver” y seleccionar la opción “Crear nuevo autómata”:



Se debe completar los datos solicitados en el formulario de creación, seleccionando el tipo de autómata (A1) el contenedor de origen y seleccionando el archivo XML de configuración virtual.

Una vez creado el Autómata Virtual hay que “habilitarlo” para que comience a procesar los datos fuentes del contenedor seleccionado. Una vez que comienza a procesar los datos, se puede ver la definición del AV y las estadísticas, identificando la cantidad de datos fuente procesados y el resultado del procesamiento:

* *Success:* cuando se generan las etiquetas y se asignan a contactos en la base



* *Warning:* cuando se generan las etiquetas, pero no se logran asignar a ningún contacto
* *Error:* cuando no se logran generar las etiquetas a partir del dato fuente, posiblemente por error en el formato del dato fuente o conflictos con la definición del XML de configuración virtual.
* *Queue:* cuando el procesamiento de las etiquetas se encuentra encolada en el sistema.

Por último, para modificar o editar un Autómata virtual, primero debe ser “deshabilitado”, luego se puede reemplazar por otro XML de configuración virtual modificado.

