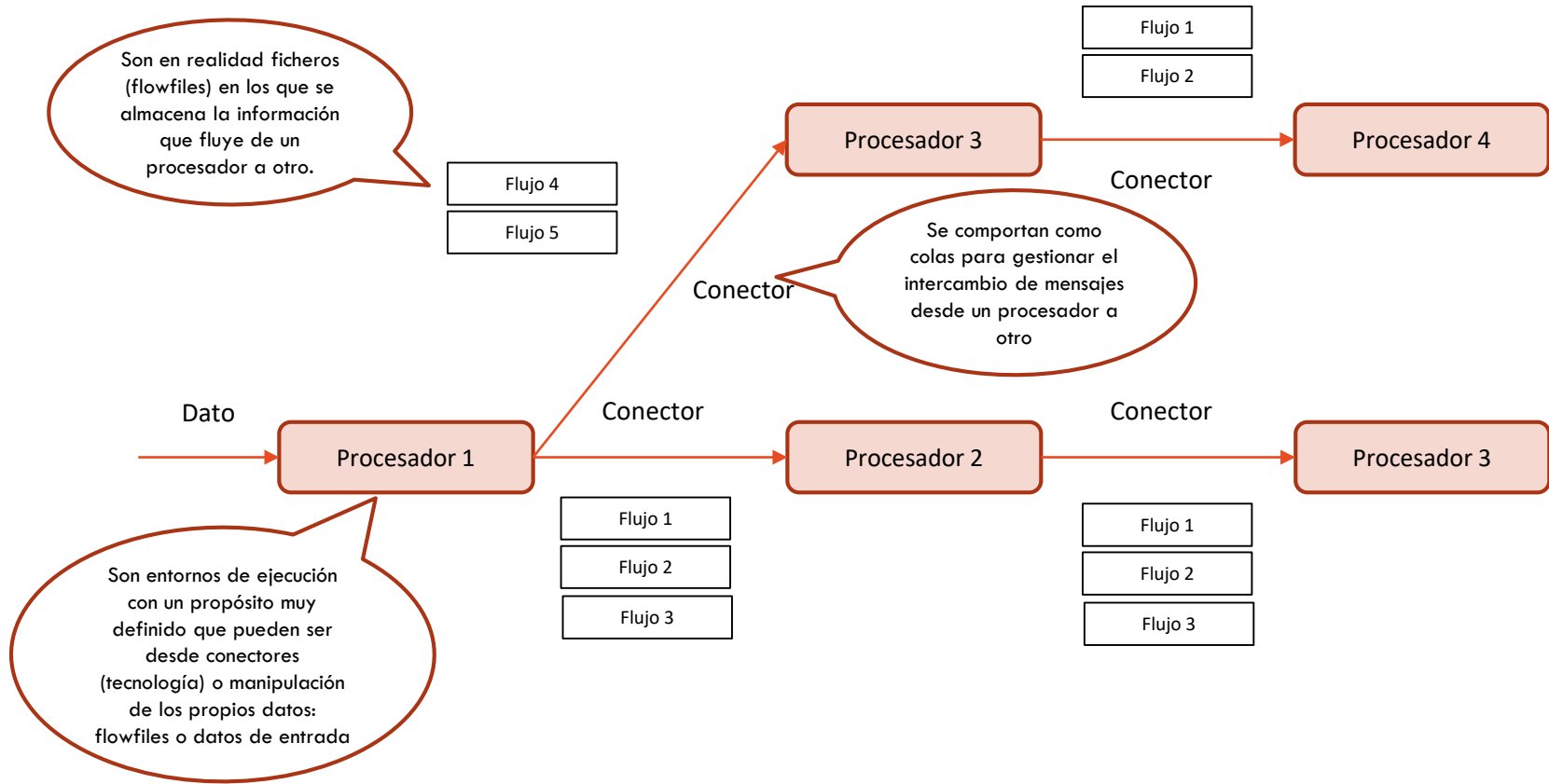


Apache NIFI

Apache Nifi. Sesion 2

- **Consideraciones adicionales:**
 - Manejo de Metadatos/Atributos
 - Nifi Expression Language
 - Expresiones regulares
- **Dimensionamiento**
- **MiNifi**

En sesiones anteriores...



Manejando Atributos

- Cada flowfile se crea con una serie de atributos que variarán a lo largo del tiempo
- Gracias a ellos podremos:
 - Tomar decisiones para enrutar un flowfile dentro de un flujo (por ejemplo, existe el procesador, RouteOnAttribute)
 - Configurar adecuadamente un procesador si lo necesitamos. Por ejemplo PutFile permite usar atributos del flowfile para seleccionar el directorio de salida
 - El atributo puede tener relación con el propio contenido del flowfile, con lo que se pueden tomar decisiones más fácilmente

Atributos básicos

En rojo si NO se pueden modificar

- filename
- path
- **uuid**: identificador único de cada flowfile
- **entryDate**: Fecha y hora en la que fue creado
- **lineageStartDate**: Cualquier fecha y hora que intervenga en el linaje de un flowfile: clonación, unión, división...
- **filesize**: numero de bytes que ocupa el flowfile

¿Cómo se pueden gestionar?

- Hay procesadores específicos para extraer (y conocer) atributos de los flowfile
- También hay procesadores específicos para crear atributos que necesitemos. (por ejemplo el procesador Update_Attribute)

UpdateAttribute

- Nos permitirá:
 - Crear (o modificar) atributos y asignarles un valor específico
 - Crear (o modificar) atributos con cierta lógica y/o utilizando variables del sistema a través del Nifi Expression Language
 - `filename = ${hostname()}-${now():format('yyyy-dd-MM')}-${filename}`
 - Definir unas reglas que establezcan cuándo y con qué valor se debe crear un atributo

UpdateAttribute. Por ejemplo

Configure Processor

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Name

Actualizando Nombre de fichero

☒ Enabled

Automatically Terminate
☐ success
All successful Flows

Id

c471841c-8465-34cb-2c72-e34f3352e1ca

Type

UpdateAttribute 1.9.2

Bundle

org.apache.nifi - nifi-update-attribute-nar

Penalty Duration ?

30 sec

Yield Duration ?

1 sec

Bulletin Level ?

WARN

▼

⚙️ ADVANCED

Configure Processor

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Required field

+

Property	Value	
Delete Attributes Expression	<input type="text" value="No value set"/>	
Store State	<input type="text" value="Do not store state"/>	
Stateful Variables Initial Value	<input type="text" value="No value set"/>	
Cache Value Lookup Cache Size	<input type="text" value="100"/>	
filename	<input type="text" value="\${fuente.0}-\${UUID()}-\${filename}"/>	<input type="button" value="🗑"/>

⚙️ ADVANCED

CANCEL

APPLY

RouteOnAttribute

- Es un procesador específico para enrutar flowfiles de acuerdo a sus atributos
- Nos permite definir las propiedades en su propia configuración
- Cada atributo del flowfile se comparará con las propiedades para comprobar si cumple o no cumple los criterios de enrutamiento. El valor de cada propiedad deberá ser una (Nifi EL) y devolver un booleano (TRUE/FALSE)
- El enrutamiento puede ser distinto tipo. Por ejemplo: “Route to property name”

Caso “Route to Property name”

- Imaginemos una propiedad:
 - nombre-empieza-por-r
 - Y le asignamos la expresión: `${filename:startsWith('r')}`
- En ese caso si el nombre del flowfile empieza con r será enrutada a esa relación. Todos los demás flowfiles seguirán en el camino “unmatched”

Nifi Expression Language (I)

- Hemos visto que NifiEL nos permite modificar atributos existentes o nuevos
- No todos los procesadores permiten ser configurados en sus propiedades para incluir NifiEL.
- Para saberlo lo mejor es consultar el help de la propiedad (?) que nos dirá si tiene un valor por defecto, es modificable y si podemos incluir expresiones regulares
- Una expresión regular empieza con `${` y termina con `}`
 - Por ejemplo `${uuid}` nos permitirá utilizar el atributo uuid
- Los nombres de los atributos deberán ser nombres sencillos, pero si contienen caracteres distintos a números, letras, o puntos deberemos ponerlo entre comillas simples: ``${Siempre quise poner este atributo}``

Nifi Expression Language (II)

- Adicionalmente podremos utilizar funciones y comparadores de atributos:

- `${filename:contains('r')}`

- O encadenar funciones:

- `${filename:toLowerCase():contains('r')}`

La ejecución será de izquierda a derecha y no modificará el valor del atributo hasta el último paso

- También se pueden anidar invocaciones:

- `${attr1:equals(${attr2})}`

Nifi Expression Language (III)


- Cuando estemos escribiendo una expresión utilizando NifiEL dentro de una propiedad contaremos con cierta ayuda:
 - Si estamos dentro de la expresión `${ }` pulsando Ctrl + Space nos saldrá una ventana con las funciones que podemos utilizar y contaremos con autocompletado
 - Si navegamos por la lista de las funciones que nos propone aparecerá una ayuda que nos dirá qué hace la función, los argumentos esperados y el tipo del resultado de la función

Nifi Expression Language (IV)

- O también podemos utilizarlas en algunas asignaciones de atributos de un procesador
- En un procesador ExtractText creamos una variable (la que coje, por ejemplo el primer registro)

Include Capture Group 0	?	true	
Enable repeating capture group	?	false	
firstWord	?	^(.+?),	

- Y lo utilizamos en un procesador UpdateAttribute

Store State	?	Do not store state	
Stateful Variables Initial Value	?	No value set	
Cache Value Lookup Cache Size	?	100	
filename	?	\${uuid}-\${firstWord}-\${filename}	

Clasificación de funciones

- Lógica booleana
- Manipulación de cadenas
- Búsqueda
- Operaciones matemáticas y manipulación numérica
- Manipulación de fechas
- Conversión de tipos
- Funciones sin parámetros
- Evaluación de múltiples atributos

<https://nifi.apache.org/docs/nifi-docs/html/expression-language-guide.html>

Lista no completa (I)

Funcionalidad	Funciones
Lógica booleana	isNull, notNull, isEmpty, equals, equalsIgnoreCase, gt, ge, lt, le, and, or, not
Manipulación de cadenas	toUpperCase, toLowerCase, trim, urlEncode, urlDecode, substring, substringBefore, substringBeforeLast, substringAfter, substringAfterLast, getDelimitedField, append, prepend, replace, replaceAll, replaceNull, replaceEmpty, length
Búsqueda	startsWith, endsWith, contains, find, matches, indexOf, lastIndexOf

Lista no completa (II)

Funcionalidad	Funciones
Operaciones matemáticas y manipulación numérica	Plus, minus, multiply, divide, mod, toRadix
Manipulación de fechas	Format, toDate, now
Conversión de tipos	toString, toNumber
Funciones sin parámetros	Ip, hostname, uuid, nextInt, literal
Evaluación de múltiples atributos	anyAttribute, allAttributes, anyMatchingAttribute, allMatchingAttributes, anyDelineatedValue, allDelineatedValues, join, count

Uso de expresiones regulares

- En algunas asignaciones a propiedades o atributos podemos utilizar algún valor que haga referencia al contenido del flowfile
- Se aceptarán expresiones regulares que sean válidas en Java

Regular Expression	Description
.	Matches any character
^regex	Finds regex that must match at the beginning of the line.
regex\$	Finds regex that must match at the end of the line.
[abc]	Set definition, can match the letter a or b or c.
[abc][vz]	Set definition, can match a or b or c followed by either v or z.
[^abc]	When a caret appears as the first character inside square brackets, it negates the pattern. This pattern matches any character except a or b or c.
[a-d1-7]	Ranges: matches a letter between a and d and figures from 1 to 7, but not d1.
X Z	Finds X or Z.
XZ	Finds X directly followed by Z.
\$	Checks if a line end follows.

Por ejemplo, el valor de un atributo

Configure Processor

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Name

Asignar Valores

Id

1d66f2f3-0b0a-3902-34d8-39b8e9b619f8

Type

ExtractText 1.9.2

Bundle

org.apache.nifi - nifi-standard-nar

Penalty Duration ?

30 sec

Bulletin Level ?

WARN

Yield Duration ?

1 sec

Automatically Terminate Relationships ?

☐ matched

FlowFiles are routed to this relationship when the Regular Expression is successfully evaluated and the FlowFile is modified as a result

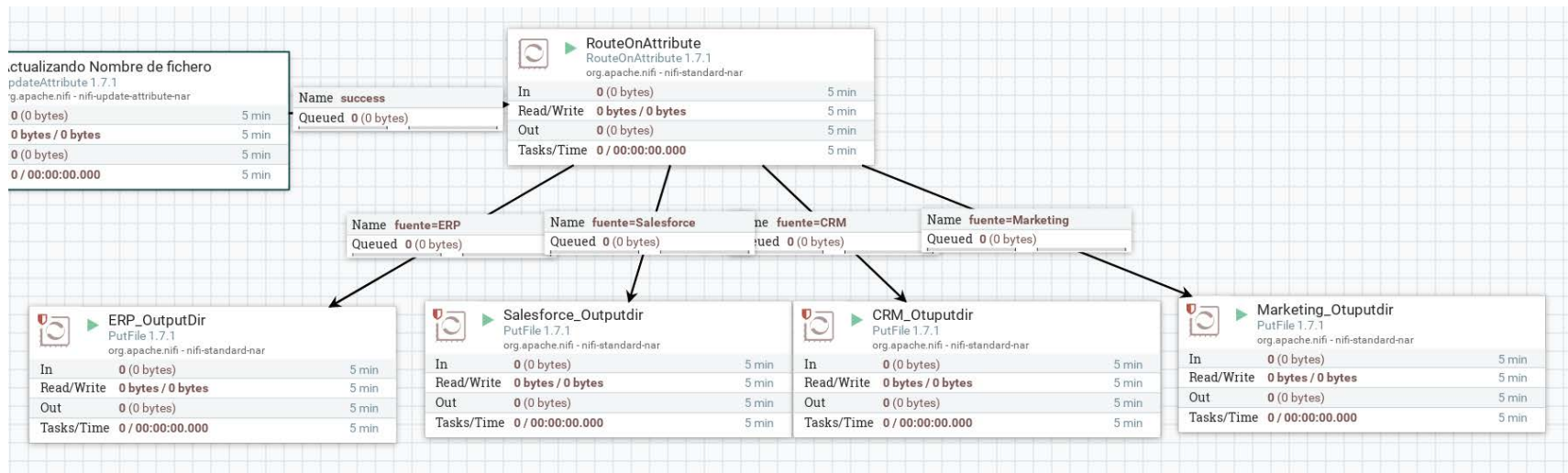
☒ unmatched

FlowFiles are routed to this relationship when no provided Regular Expression matches the content of the FlowFile

Enable DOTALL Mode	<input type="checkbox"/>	false
Enable Literal Parsing of the Pattern	<input type="checkbox"/>	false
Enable Multiline Mode	<input type="checkbox"/>	false
Enable Unicode-aware Case Folding	<input type="checkbox"/>	false
Enable Unicode Predefined Character Classes	<input type="checkbox"/>	false
Enable Unix Lines Mode	<input type="checkbox"/>	false
Include Capture Group 0	<input type="checkbox"/>	true
Enable repeating capture group	<input type="checkbox"/>	false
fuelle	<input type="checkbox"/>	(^ERP)(^(Salesforce))(^(CRM))(^(Marketing))

Y así, por ejemplo, podré usar su valor para enrutar

SETTINGS	SCHEDULING	PROPERTIES	COMMENTS
Required field			
Property	Value		
Routing Strategy	?	Route to Property name	
fuelle=CRM	?	\$(fuente.0.equals("CRM"))	
fuelle=ERP	?	\$(fuente.0.equals("ERP"))	
fuelle=Marketing	?	\$(fuente.0.equals("Marketing"))	
fuelle=Salesforce	?	\$(fuente.0.equals("Salesforce"))	

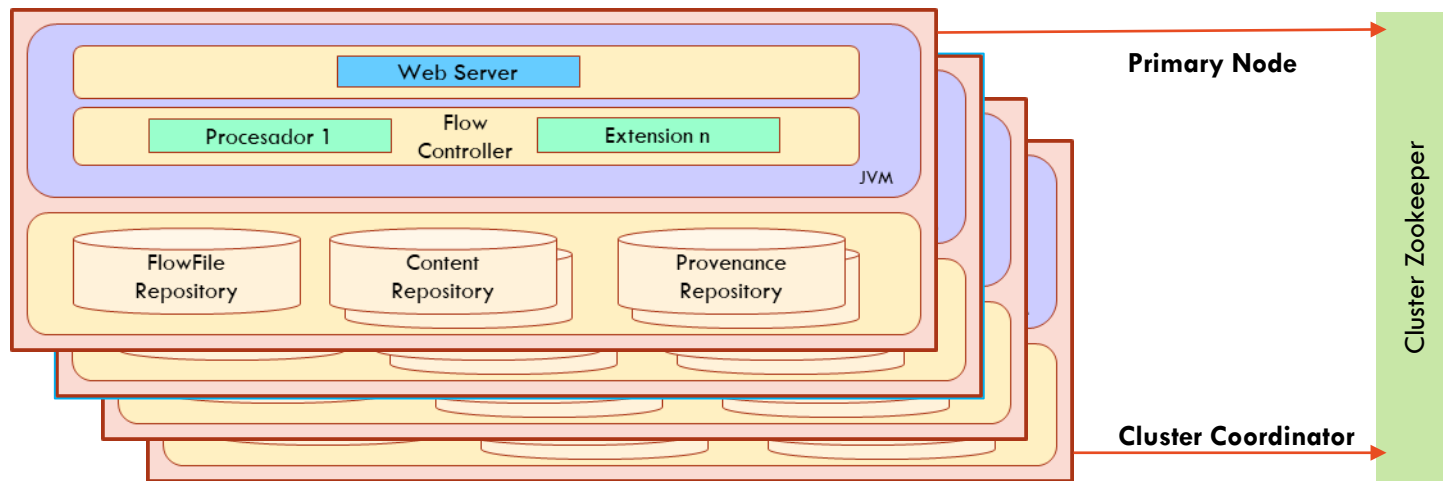


Cluster

Utiliza Apache **Zookeeper** para determinar qué componentes del cluster están disponibles
Nifi lleva **embebido** Zookeeper pero puede hacer uso de un cluster externo

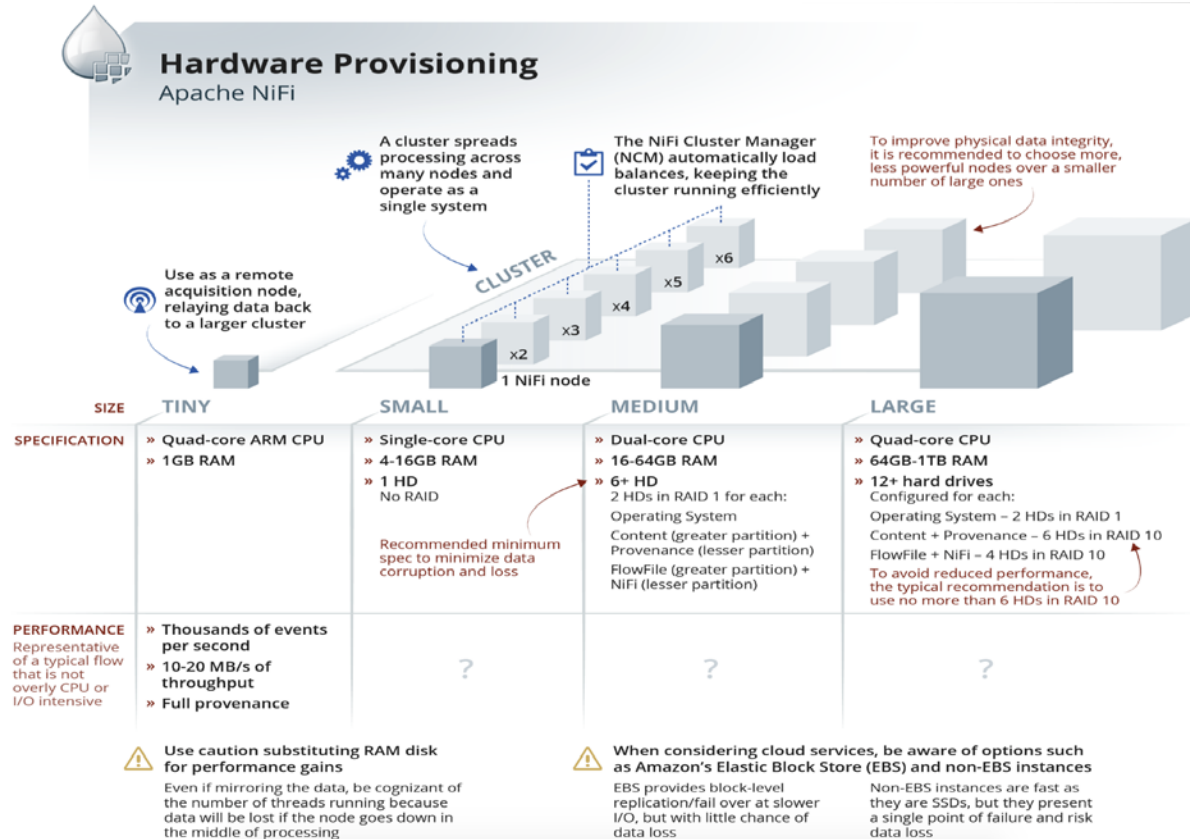
Zookeeper selecciona y asigna dos roles:

- **Cluster Coordinator**, recibe los heartbeats y estado de los miembros del cluster.
- **Primary Node** ejecuta aquellos procesadores que no son escalables o que no pueden ser ejecutados en paralelo en el cluster. (En un principio todos los procesadores se ejecutan en todos los miembros del cluster).



La **edición** de un DataFlow se puede hacer desde cualquiera de los nodos del cluster y es replicada a todos los miembros

Aprovisionamiento



<https://community.cloudera.com/t5/Community-Articles/NiFi-Sizing-Guide-Deployment-Best-Practices/ta-p/246781>

Throughput

Throughput Target	Number of NiFi nodes	CPU Cores/node	Number of disks/node, size of each disk (RAID 5/10)	RAM/node	Ideal Networking Setup
50 MB/s, 1000 events/s	3	16+	6+, 1TB	8+ GB	1 Gigabit bonded NICs
100 MB/s, 10,000 events/s	5	16+	6+, 2TB	8+ GB	1 Gigabit bonded NICs
200 MB/s, 100,000 events/s	7	24+	12+, 4TB	16+ GB	10 Gigabit bonded NICs
400 MB/s, 100,000+ events/s	9	24+	12+, 8TB	16+ GB	10 Gigabit bonded NICs

Apache Minifi

- Pensemos en Apache Minifi como en un *agente* Nifi que desplegaremos donde se *genere* el dato
- Está disponible en dos lenguajes Java y C++
 - Java: 49MB de código y 24MB de heapsize por defecto. Casi todos los procesadores de Minifi están disponibles
 - C++: 3.2MB de código y 5MB de memoria en reposo. Solo puede ejecutar un número limitado de procesadores
- Ejecuta flujos definidos a través de Nifi:
 - Se exportan en template (.xml)
 - Se convierten a flujo de minifi (.yml)

¿Cómo se hace? (I)

En el servidor:

1. Se configura Nifi para habilitar sockets por un puerto determinado (nifi.properties)
2. Creamos un flujo (que luego enviaremos a Minifi. **Es lo que se ejecuta en el agente**). Especificamos el servidor de Nifi a través de un “Grupo remoto de procesadores”



¿Cómo se hace? (II)

3. Exportamos el flujo (como template). El fichero xml que se genera nos lo llevamos al agente.

En el agente:

1. El fichero xml (*) se convierte a formato .yaml (utilizando el **toolkit** de minifi)
2. Copiamos el fichero .yaml al directorio /conf en el agente (llamándolo config.yaml)
3. Levantamos el *agente* minifi

(*) Debido a un bug en la versión actual es necesario borrar la entrada etiquetada como “target-id” para que la conversión no de problemas.

¿Cómo se hace? (III)

De vuelta en el servidor:

1. Creamos un flujo utilizando “Input Port” y lo asociamos al flujo que deseemos:

