

---

# **Documentación de PySafet**

***Publicación***

**Cenditel**

26 de May de 2015



<b>1. 1.- Introducción</b>	<b>1</b>
1.1. ¿Qué es PySafet? . . . . .	1
1.2. Arquitectura . . . . .	1
1.3. Organización de trabajo . . . . .	2
1.4. Motivación y objetivos . . . . .	2
1.5. Ejemplos de algunos Diagrama de flujo de trabajo . . . . .	3
1.6. Sistema Automatizado de Firma Electrónica y Estampado Electrónico . . . . .	6
<b>2. 2.- Instalación</b>	<b>7</b>
2.1. Debian Wheezy 7.6.0 (64 bits/32bits) . . . . .	7
2.2. Ubuntu (32 bits/ 64 bits) . . . . .	9
2.3. Desde los Archivos fuentes . . . . .	9
2.4. Herramienta <b>inflow</b> . . . . .	13
<b>3. 3.- Configuración</b>	<b>17</b>
3.1. Cargar un proyecto(Productos) en el directorio <b>.safet/</b> . . . . .	17
3.2. Guardar el proyecto del directorio <b>.safet/</b> . . . . .	19
3.3. Sobre el directorio <b>.safet/</b> . . . . .	20
<b>4. 4.- Descripción de parámetros del directorio &lt;HOME&gt;.safet/</b>	<b>23</b>
4.1. <b>auth.conf</b> (Archivo de autenticación y autorización) . . . . .	23
4.2. <b>safet.conf</b> (Archivo de configuración inicial) . . . . .	27
<b>5. 5.- API de Safet para el lenguaje de Python</b>	<b>39</b>
5.1. Clase MainWindow(Safet.MainWindow) . . . . .	39
5.2. Clase SafetXObject (Safet.SafetXObject) . . . . .	45
5.3. Clase SafetDocument . . . . .	45
5.4. Clase SafetVariable . . . . .	49
5.5. Clase SafetYAWL . . . . .	49
5.6. Clase SafetWorkflow . . . . .	50
<b>6. 6.- Tutorial de un ejemplo de inventario</b>	<b>51</b>
6.1. Creación de la <b>base de datos</b> . . . . .	51
6.2. Operaciones <b>CRUD+(flujo)</b> utilizando PySafet . . . . .	63

---

6.3.	Crear el <b>flujo de trabajo</b> utilizando un archivo XML . . . . .	85
6.4.	Ver <b>datos</b> usando flujos de trabajo . . . . .	118
6.5.	Ver <b>fichas</b> usando flujos de trabajo . . . . .	131
6.6.	Ver <b>gráficos</b> coloreado usando flujos de trabajo . . . . .	139
6.7.	<b>Parámetros</b> usando flujos de trabajo . . . . .	150
<b>7.</b>	<b>7.- Tutorial del “ejemplo de inventario” usando la interfaz gráfica (Inflow)</b>	<b>163</b>
7.1.	Introducción a <b>inflow</b> . . . . .	163
7.2.	Ejecución de operaciones <b>CRUD+</b> . . . . .	176
7.3.	Ejecución de <b>listados</b> de reportes <b>normal</b> . . . . .	227
7.4.	Ejecución de <b>listados</b> de reportes con <b>parámetros</b> . . . . .	241
7.5.	Ejecución de <b>listados</b> de reportes con <b>Autofiltros</b> . . . . .	256
7.6.	Ejecución de <b>gráficos</b> usando flujos de trabajos <b>normal</b> . . . . .	278
7.7.	Ejecución de <b>gráficos</b> usando flujos de trabajos con <b>parámetros</b> . . . . .	301
7.8.	Ejecución de <b>gráficos</b> usando flujos de trabajos con <b>autofiltros</b> . . . . .	320
7.9.	Diseño de <b>creación</b> de un <b>nuevo gráfico</b> de flujo de trabajo ( <b>.xml</b> ) . . . . .	333
7.10.	Diseño de <b>agregar</b> nuevos nodos al gráfico ( <b>.xml</b> ) . . . . .	358
7.11.	Diseñar el <b>cambio</b> de <b>conexiones</b> de nodos enlazados . . . . .	378
7.12.	Diseño de <b>borrar</b> un nodos del gráficos de flujo de trabajo . . . . .	399
<b>8.</b>	<b>8.- Tutorial del “ejemplo de inventario” usando la interfaz gráfica (Framework Web-Safet)</b>	<b>413</b>
8.1.	Instalación y configuración . . . . .	413
8.2.	Ejecución de operaciones <b>CRUD+</b> . . . . .	424
8.3.	Ejecución de <b>listados de reportes</b> . . . . .	461
8.4.	Ejecución de <b>gráficos</b> de flujos de trabajo . . . . .	469
8.5.	Ejecución de <b>gráficos</b> de flujos de trabajo de <b>seguimiento</b> . . . . .	477
<b>9.</b>	<b>9.- Tutorial de firma electrónica</b>	<b>501</b>
9.1.	Conceptos básico . . . . .	501
9.2.	Instalación de los componentes . . . . .	501
9.3.	Uso de la firma electrónica de manera <b>Web</b> . . . . .	503
<b>10.</b>	<b>Documentación del sistema de Plan Operativos Anuales(POA)</b>	<b>509</b>
10.1.	<b>1.- Instroducción</b> . . . . .	509
10.2.	<b>2.- Instalación</b> . . . . .	558
10.3.	<b>3.- Configuración</b> . . . . .	562
10.4.	<b>4.- Secciones del Editorflow</b> . . . . .	563
10.5.	<b>Ejemplo paso a paso de la herramienta EditFlow</b> . . . . .	570

## 1.- Introducción

---

### 1.1 ¿Qué es PySafet?

Es una **librería** disponible para los lenguajes de programación **C++** y **Python**, que permite construir aplicaciones informáticas de manera rápida, para ello se definen dos conceptos:

1. **Flujo de trabajo(workflow):** Consiste en el estudio de aspectos operacionales de una actividad de trabajo, esto es, cómo se realizan y estructuran las tareas, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información y cómo se hace su seguimiento. [Más contenido AQUI](#).
2. **Firma electrónica:** Es un concepto jurídico equivalente a electrónico de la firma manuscrita, donde una persona acepta el contenido de un mensaje electrónico a través de cualquier medio electrónico válido.

Toda la aplicación se desarrolla con **PySafet** y no como **Framework** tradicional. Los conocimientos que deben manejar para esta aplicación **PySafet** son:

1. **XML** (lenguaje de marcador). [Xml](#).
2. **SQL** (lenguaje de consultas de base de datos). [Sql](#).
3. **Python** (lenguaje dinámico). [Python](#).

### 1.2 Arquitectura

Un motor de flujo de trabajo tiene como principal objetivo calcular (los) el siguiente estado(s) disponibles para la ficha.

- **Se agregan:**

1. Visualización.

2. Reporte y tiempos.
3. Validación de estados.

### 1.3 Organización de trabajo

- Unas de las características de las organizaciones de hoy en día es su funcionalidad dinámica, emergente y pro activo se plasman en sus procesos.
- Una organización es medible a través de sus procesos. Si los procesos funcionan bien la organización funciona bien.
- Es por ello que la gestión debe fundamentarse en los procesos, identificando de manera expedita posibles fallos.
- Para medir los procesos se recurre a herramientas informáticas
- **¿Qué pasa si estas herramientas no son flexibles (adaptarse en uno de los procesos)?**
  1. Muchos procesos o roles de profesionales no se automatizan.
  2. Los dueños de los procesos no son los adecuados.
  3. Se produce aumentos de errores.
  4. No es buena la ejecución de los procesos (muchas veces es mala).
  5. Es frecuente la aparición de “cuellos de botella”; Al funcionamiento es pésimo cuando se aumenta la carga.
  6. No hay información oportuna y fiable.
  7. Las auditorías aumentan grandes esfuerzos y no se obtiene buenos resultados.

**Ejemplo de una Organización:** *Figura 1: Organización de trabajo:*

### 1.4 Motivación y objetivos

1. Pysafet surge como idea de Fundacite y sucede para complementar la infraestructura de firma electrónica.
2. Se planteaba como un **“motor de flujos de trabajo”**.
3. La construcción de un motor de flujo de trabajo es compleja, hay dos modelos teóricos:
  - **BPMN:** Extensión UML para comunicar y como segundo objetivo implementar y calcular.
  - **Redes de petri:** Calcular y segundo objetivos comunican.

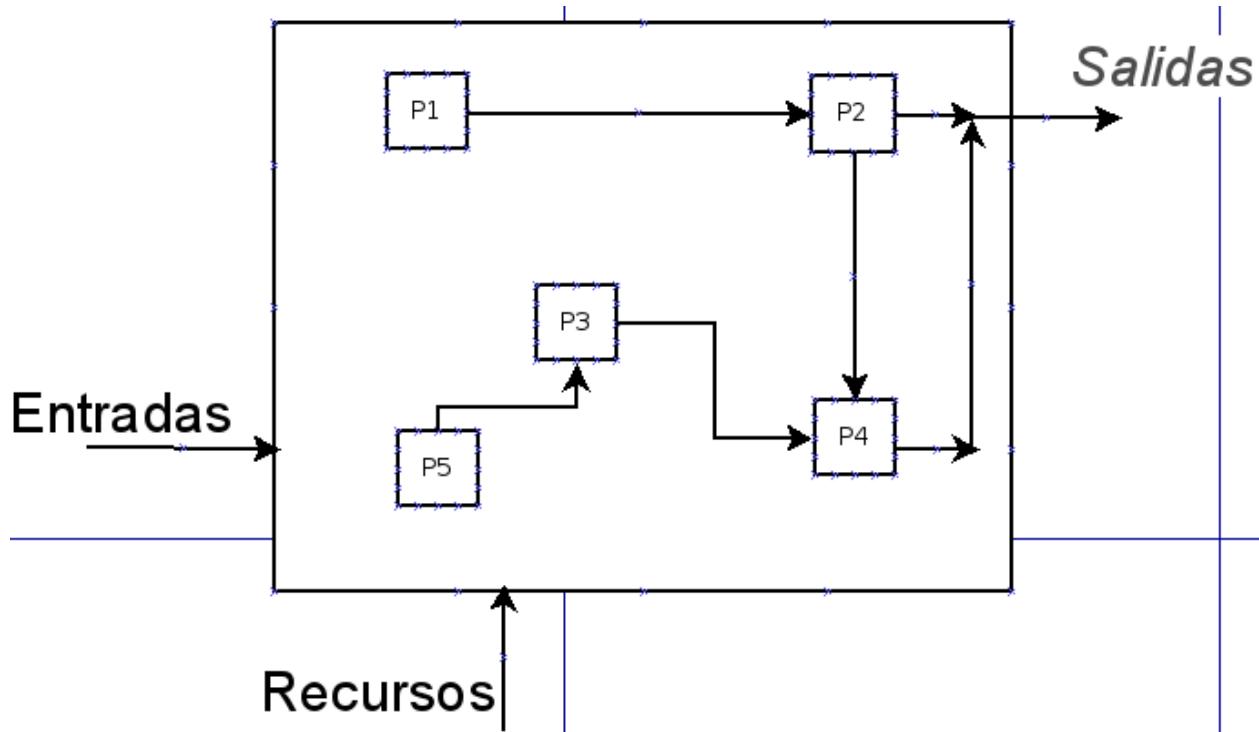


Figura 1.1: Figura 1: Organización de trabajo

## 1.5 Ejemplos de algunos Diagrama de flujo de trabajo

Todo proceso puede modelarse como un flujo de trabajo. Se hace necesario identificar “**Eventos**”, “**Estado**”, “**Fichas**”, “**Recursos**”, “**Dependencias**”, “**Roles**”, “**Patrones**”, “**Mensajes**”.

- **Eventos:** Posible hecho que cambia de estado es una ficha.
- **Estado:** Lugar donde reside una ficha.
- **Ficha:** Documento único dentro del proceso(puede cambiar de datos durante el cambio de estado, pero puede mantener un enlace o **id** único).
- **Recursos:** Lo que necesita el proceso para funcionar.
- **Roles:** Identidades (digitales) como posible o responsabilidades en los activos de información.
- **Patrones:** Compuestos de modelos generales o recurrentes en los procesos.
- **Mensajes:** Información sobre el acontecer de un evento dirigido a un rol externo (generalmente).

### 1.5.1 1º PRIMER EJEMPLO

Solicitar un permiso de un usuario. PySafet utiliza el gráfico de **safet** para observar el flujo de trabajo, como se muestra en la siguiente *Figura 2: Gráfico safet*:

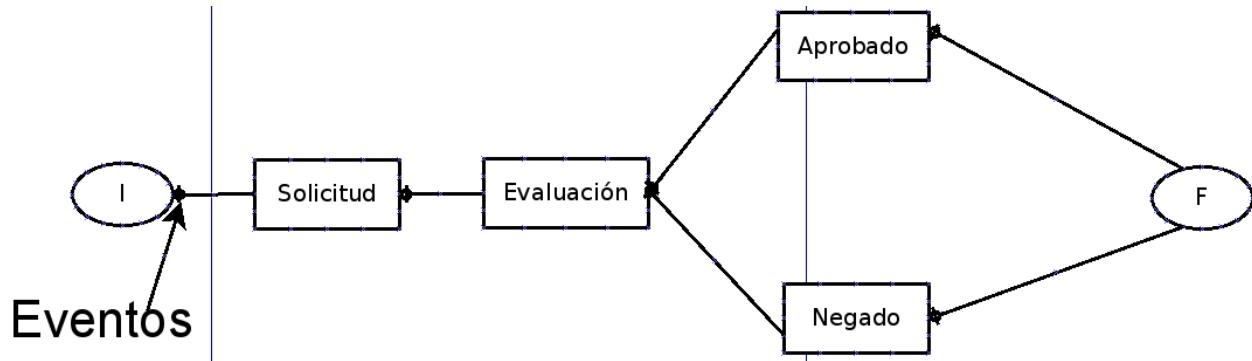


Figura 1.2: **Figura 2: Gráfico safet**

### 1.5.2 2° SEGUNDO EJEMPLO

Como determinar los (**Estado, eventos, fichas, dependencias, recursos y roles**), de la solicitud de permiso de un usuario con el **gráfico de Petrii**, como se muestra en la siguiente *Figura 3: Gráfico Petrii*:

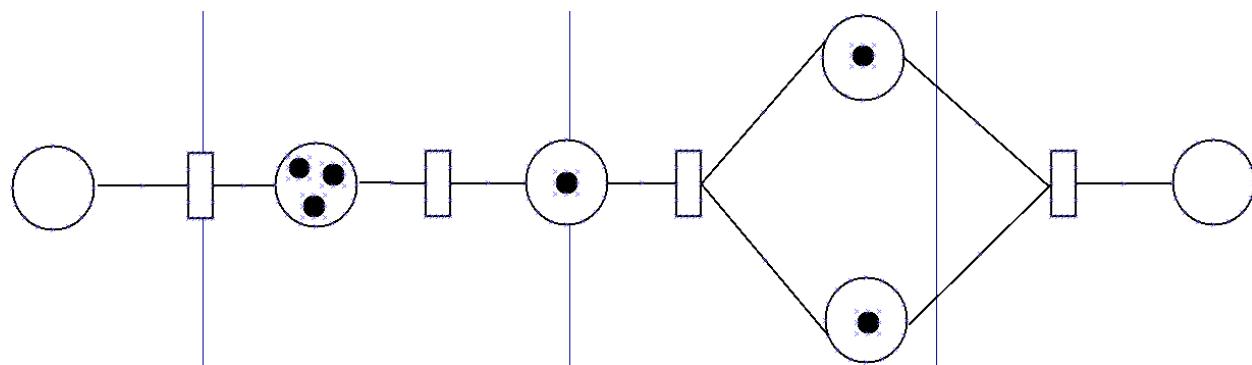


Figura 1.3: **Figura 3: Gráfico Petrii**

### 1.5.3 3° TERCER EJEMPLO

Como identificar los (**eventos, estados, dependencias, roles, patrones, mensajes**) de una caja de ahorro, con el gráfico de **safet**, como se muestra en la siguiente *Figura 4: Caja de ahorro*:

### 1.5.4 4° CUARTO EJEMPLO

El ejemplo basico de **PySafet** hola mundo con el gráfico de **safet**, como se muestra en la siguiente *Figura 5: Inicio de PySafet*:

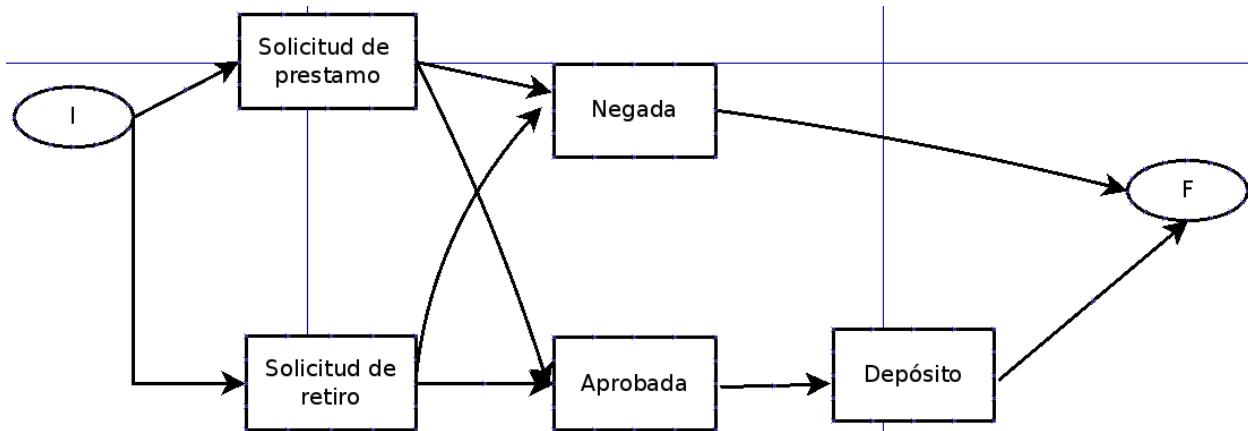


Figura 1.4: Figura 4: Caja de ahorro



Figura 1.5: Figura 5: Inicio de PySafet

## 1.6 Sistema Automatizado de Firma Electrónica y Estampado Electrónico

Es una herramienta que permite desarrollar nuevas aplicaciones de software con flujos de trabajo, es decir, automatización de forma expedita de procesos de una organización, agregando firma Electrónica y estampillado de tiempo.

La firma electrónica tiene un soporte legal en la Ley sobre Mensajes de Datos y Firma Electrónica (2001) de la República Bolivariana de Venezuela.

El sistema automatizado para la Firma Electrónica y Estampado de Tiempo (SAFET) surge como una herramienta que permite desarrollar nuevas aplicaciones de software que incluyan las funcionalidad de flujos de trabajo, firma Electrónica y estampillado de tiempo.

## 2.- Instalación

---

### 2.1 Debian Wheezy 7.6.0 (64 bits/32bits)

En este tutorial explicaremos la instalación se **Pysafet** en la distribución **linux(debian)**, para ello seguimos los siguientes pasos:

#### 2.1.1 1° PRIMER PASO

- Abrimos el archivo `sources.list` y agregamos el siguiente repositorio:

```
deb http://tibisay.cenditel.gob.ve/repositorio wheezy main
```

#### 2.1.2 2° SEGUNDO PASO

- Desde la consola de comando, como (**SuperUsuario**), obtenemos la clave (GPG) del repositorio Debian, para ello ejecutamos los siguientes linea:

```
# wget http://tibisay.cenditel.gob.ve/repositorio/apt-seguridad.gpg.asc  
# apt-key add apt-seguridad.gpg.asc
```

---

**Nota:** Se nos muesta lo siguiente:

---

#### 2.1.3 3° TERCER PASO

- Desde la consola de comando, como (**SuperUsuario**), ejecutamos un **update** como se muestra a continuación:

```
# aptitude update
```

## 2.1.4 4° CUARTO PASO

- Desde la consola de comando, como (**SuperUsuario**), ejecutamos un **update** como se muestra a continuación:

```
# aptitude install pysafet
```

**Nota:** En la siguiente *Figura 6: Instalación Pysafet* se muestra como esta instalando **Pysafet**:

The screenshot shows a terminal window titled 'Terminal' with the command 'aptitude install pysafet' being run. The output shows the package list, dependencies, and a warning about unsigned packages. It ends with a prompt asking if the user wants to proceed.

```
Actividades Terminal jue 2:45 PM cenditel@debian:/home/cenditel/ejemplo_sphinx/sphinx# aptitude install pysafet
root@debian:/home/cenditel/ejemplo_sphinx/sphinx# aptitude install pysafet
Se instalarán los siguientes paquetes NUEVOS:
  libqtassistantclient4{a} libsafet{a} pysafet python-qt4{a}
  python-qt4-sql{a}
0 paquetes actualizados, 5 nuevos instalados, 0 para eliminar y 98 sin actualizar.
Necesito descargar 0 B/7.713 kB de ficheros. Después de desempaquetar se usarán
34,0 MB.
¿Quiere continuar? [Y/n/?] y
AVISO: se instalarán versiones sin firmar de los siguientes paquetes!

Los paquetes sin firmar pueden comprometer la seguridad del sistema.
Sólo debe continuar con la instalación si está completamente seguro de que es lo
que quiere.

pysafet

¿Quiere ignorar este aviso y continuar de todos modos?
Para continuar, introduzca "Si"; para abortar, introduzca "No":si
```

Figura 2.1: **Figura 6: Instalación Pysafet**

---

## 2.1.5 5° QUINTO PASO

- Desde la consola de comando, como (**UsuarioNormal**), entramos a la consola de python, como se muestra a continuación:

```
$ python
```

**Nota:** Al ejecutar python entramos a la consola como se muestra a continuación:

```
Python 2.7.6 (default, Mar 22 2014, 22:59:56)
[GCC 4.8.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

## 2.1.6 6° SEXTO PASO

- En la consola de python importamos a **Safet**, como se muestra a continuación:

```
>>> import Safet  
>>>
```

## 2.1.7 7° SEPTIMO PASO

- En la consola de python visualizamos algunas funciones de **PySafet** como se muestra a continuación

```
>>> dir(Safet)  
['MainWindow', 'ParsedSqlToData', 'SafetDocument', 'SafetVariable',  
 'SafetWorkflow', 'SafetXmlObject', 'SafetYAWL',  
 '__doc__', '__file__', '__name__', '__package__']
```

Otra forma de mostrar las funciones de **PySafet**:

```
>>> for i in range(len(dir(Safet))):  
...     dir(Safet)[i]  
...  
'MainWindow'  
'ParsedSqlToData'  
'SafetDocument'  
'SafetVariable'  
'SafetWorkflow'  
'SafetXmlObject'  
'SafetYAWL'  
'__doc__'  
'__file__'  
'__name__'  
'__package__'
```

## 2.2 Ubuntu (32 bits/ 64 bits)

## 2.3 Desde los Archivos fuentes

El código fuente de la **librería Libsafet** se encuentra alojada en una plataforma de desarrollo colaborativo de software (**forja**) llamada **Github**, esta plataforma utiliza como sistema de control de versión el software **Git**.

### 2.3.1 A.- Dependencias

**PySafet** depende de las siguientes dependencias:

1. **libgraphviz-dev**
2. **libtar-dev**

3. **g++**
4. **libglib2.0-dev**
5. **x11proto-xext-dev**
6. **libqt4-dev**
7. **libssl-dev**
8. **make**
9. **python-qt4-sql**
10. **python-sip-dev**
11. **python-qt4-dev**
12. **libqt4-sql-sqlite**

### Para la misma se debe hacer los siguiente:

- Desde la línea de comando o terminal como usuario (**root**) instalamos la dependencias para pysafet, como se muestra a continuación:

```
# aptitude install libgraphviz-dev x11proto-xext-dev g++ libglib2.0-dev  
# aptitude install libqt4-dev libssl-dev make python-qt4-sql libtar-dev  
# aptitude install python-qt4-dev libqt4-sql-sqlite python-sip-dev
```

### 2.3.2 B.- Descargamos el repositorio Libsafet

Para descargar **PySafet** debemos clonar el repositorio de **Libsafet**, para ello seguimos los siguientes paso:

#### 1° PRIMER PASO

- Desde la consola de comando como usuario (**root**),instalamos el control de versiones **git**, como se muestra a continuación:

```
# aptitude install git
```

#### 2° SEGUNDO PASO

- Desde nuestro directorio de trabajo, descargamos las fuente pysafet con el comando (**git clone url**), como se muestra en el siguiente comando:

```
$ git clone https://github.com/victorrbravo/pysafet.git pysafet
```

---

**Nota:** Dentro del directorio de trabajo se crea el directorio **pysafet**, donde se tiene el clone de la librería Libsafet

---

### 2.3.3 C.- Pasos para la compilación de la librería Libsafet

#### 1° PRIMER PASO

- Desde la línea de comando o terminal como usuario (**normal**), accedemos al directorio (**pysafet/websafet**) desde nuestro directorio de trabajo, para la misma debe hacer los siguientes:

```
$ cd pysafet/websafet
```

#### 2° SEGUNDO PASO

- Ejecutamos los siguiente:

```
pysafet/websafet $ make clean  
pysafet/websafet $ qmake-qt4  
pysafet/websafet $ make
```

#### 3° TERCER PASO

- Nos cambiamos al directorio (**PySafet**), como se muestra a continuación:

```
pysafet/websafet $ cd ../PySafet/
```

#### 4° CURTO PASO

- Dentro de directorio (**PySafet**), ejecutamos lo siguiente:

```
pysafet/PySafet $ python configure.py  
pysafet/PySafet $ make
```

#### 5° QUINTO PASO

- Dentro de directorio (**PySafet/**) pero como usuario (**root**), ejecutamos los siguiente:

```
pysafet/PySafet # make install
```

#### 6° SEXTO PASO

- Nos cambias al directorio (**websafet**), como se muestra a continuación:

```
pysafet/PySafet $ cd ../websafet/
```

## 7° SEPTIMO PASO

- Dentro de directorio (**websafet/**) pero como usuario (**root**), ejecutamos los siguiente:

```
pysafet/websafet # make install
```

### 2.3.4 D.- Verificamos si se instaló PySafet

#### 1° PRIMER PASO

- Desde la línea de comando o terminal como usuario (**normal**) accedemos al interprete de **Python**, para la misma debe hacer los siguientes:

```
pysafet/websafet $ python
```

---

**Nota: Se mostrara de esta forma:**

```
Python 2.7.3 (default, Jan 2 2013, 13:56:14)
[GCC 4.7.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

---

#### 2° SEGUNDO PASO

- Dentro **prompt** de **python(>>>)**, importamos la librería de **Safet** ejecutando el siguiente comando

```
>>> import Safet
>>>
```

#### 3° TERCER PASO

- Dentro **prompt** de **python(>>>)**, observamos algunas funciones de **Safet**, para la misma debe hacer los siguientes:

```
>>> dir(Safet)
```

---

**Nota: Se nos muestra las funciones de la siguiente manera:**

```
['MainWindow', 'ParsedSqlToData', 'SafetDocument', 'SafetVariable',
 'SafetWorkflow', 'SafetXmlObject', 'SafetYAWL', '__doc__',
 '__file__', '__name__', '__package__']
```

---

## 2.4 Herramienta inflow

En este tutorial es necesario tener conocimiento básico de instalación de paquetes, y actividades mínimas de administración. Los **paquetes** necesarios para la instalación **inflow** son:

1. Librería Trolltech Qt >= 4.4 (**libqt4-dev**).
2. Librería de Conexión a Datos Qt PSQL (**libqt4-sql-psql**).
3. Librería de Conexión a Datos Qt SQLITE (**libqt4-sql-sqlite**).
4. Librería XML2 (**libxml2-dev**).
5. Librería de archivo tareados (**libtar-dev**).
6. Librería Graphviz (**libgraphviz-dev**, para plugins de visualización de gráfico).
7. Libreria **libtar-dev** (Utilidad “tar” para respaldo de archivos).
8. Compilador **g++ y make**.

A continuación seguimos los siguientes pasos para la instalación de la (interfaz inflow):

### 2.4.1 1º PRIMER PASO

- Desde la consola de comando, como usuario **root**, instalamos todo los paquetes para **inflow**, como se muestra a continuación:

```
# aptitude install libqt4-dev libqt4-sql-psql libxml2-dev libtar-dev  
# aptitude install g++ make libqt4-sql-sqlite libgraphviz-dev
```

---

**Nota:** A continuación se muestra la instalación en la siguiente *Figura 7: Instalación de paquetes para inflow*:

```
root@debian:/home/cenditel# aptitude install g++ libqt4-dev libqt4-sql-psql libxml2-dev libqt4-sql-sqlite libtar-dev libgraphviz-dev g++ make  
Se instalarán los siguiente paquetes NUEVOS:  
libxml2-dev  
Se actualizarán los siguientes paquetes:  
libxml2  
1 paquetes actualizados, 1 nuevos instalados, 0 para eliminar y 140 sin actualizar.  
Necesito descargar 1.804 kB de ficheros. Después de desempaquetar se usarán 2.757 kB.  
¿Quiere continuar? [Y/n/?] y  
Des: 1 http://security.debian.org/ wheezy/updates/main libxml2 amd64 2.8.0+dfsg-1.7+wheezy1 [904 kB]  
25% [1 libxml2 444 kB/904 kB 49%] 14,7 kB/s 1min. 32seg.
```

Figura 2.2: **Figura 7: Instalación de paquetes para inflow**

---

## 2.4.2 2° SEGUNDO PASO

- Desde la consola de comando, como usuario **normal**, creamos un directorio llamado **inflow** en nuestro **<HOME>**, como se muestra a continuación:

```
$ mkdir $HOME/inflow
```

## 2.4.3 3° TERCER PASO

- Desde la consola de comando, como usuario **normal**, accedemos al directorio que creamos **inflow**, como se muestra a continuación:

```
$ cd $HOME/inflow
```

## 2.4.4 4° CUARTO PASO

- Desde la consola de comando, como usuario **normal**, descargarmos la fuentes con el comando **git**, como se muestra a continuación:

```
inflow $ git clone https://bitbucket.org/victorrbravo/inflow.git
```

---

**Nota:** Observamos la siguiente *Figura 8: Descargamos inflow*:

```
cenditel@debian:~/Inflow$ git clone https://bitbucket.org/victorrbravo/inflow.git
Cloning into 'inflow'...
remote: Counting objects: 1426, done.
remote: Compressing objects: 100% (1172/1172), done.
Receiving objects: 9% (129/1426), 1.41 MiB | 40 KiB/s
```

Figura 2.3: **Figura 8: Descargamos inflow**

---

## 2.4.5 5° QUINTO PASO

- Desde la consola de comando, como usuario **normal**, accedemos al directorio descomprimido **inflow/**, la cual se descargo cuando ejecutamos el comando **git**, como se muestra a continuación:

```
inflow $ cd inflow/
```

## 2.4.6 6° SEXTO PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos el paquete **qmake** o **qmake-qt4** para generar el archivo de compilación (**Makefile**), como se muestra a continuación:

```
inflow/inflow $ qmake-qt4
```

**Nota:** Observe la siguiente *Figura 9: qmake-qt4*:

```
cenditel@debian:~/Inflow/inflow$ qmake-qt4
Project MESSAGE: Configuración y compilación de SAFET
Project MESSAGE: ... Verificación de librería Libxml2: OK
Project MESSAGE: ... Verificación de encabezados de Libgraphviz: OK
Project MESSAGE: ... Verificación de librería Libgraphviz: OK
Project MESSAGE: ... Verificación de encabezados de libtar-dev: OK
Project MESSAGE: ... Verificación de compilador g++ : OK
```

Figura 2.4: **Figura 9: qmake-qt4**

**Nota:** Si en la pantalla no se muestran errores pasamos al siguiente paso.

#### 2.4.7 7° SEPTIMO PASO

- Desde la consola de comando, como usuario **normal**, realizamos la compilación ejecutando **make**, como se muestra a continuación:

```
inflow/inflow $ make
```

**Nota:** Observe la siguiente *Figura 10: make*:

```
cenditel@debian:~/Inflow/inflow$ make
cd src/ && make -f Makefile
make[1]: se ingresa al directorio `/home/cenditel/Inflow/inflow/src'
/usr/bin/qmake-qt4 -o Makefile src.pro
Project MESSAGE: Configuring for xml2...
Project MESSAGE: Configuring for gsoap...
Project MESSAGE: Configuring for xml2...
Project MESSAGE: Configuring for gsoap...
Project MESSAGE: Configuring for xml2...
Project MESSAGE: Configuring for gsoap...
make[1]: se sale del directorio `/home/cenditel/Inflow/inflow/src'
make[1]: se ingresa al directorio `/home/cenditel/Inflow/inflow/src'
make -f Makefile.Release
make[2]: se ingresa al directorio `/home/cenditel/Inflow/inflow/src'
g++ -c -m64 -pipe -O2 -fPIC -w -D_REENTRANT -DQT_NO_DEBUG -DSAFET_N
O_DBXML -DSAFET_XML2 -DSAFET_TAR -DSAFET_GSOAP -DQT_NO_DEBUG -DQT_W
EBKIT_LIB -DQT_SVG_LIB -DQT_SQL_LIB -DQT_XML_LIB -DQT_GUI_LIB -DQT_N
ETWORK_LIB -DQT_CORE_LIB -DQT_SHARED -I/usr/share/qt4/mkspecs/linux-g++-64 -I. -I/usr/include/qt4/QtCore -I/usr/include/qt4/QtNetwork -I/usr/include/qt4/QtGui -I/usr/include/qt4/QtX
ml -I/usr/include/qt4/QtSql -I/usr/include/qt4/QtSvg -I/usr/include/qt4/QtWebKi
```

Figura 2.5: **Figura 10: make**

## 2.4.8 8° OCTAVO PASO

- Desde la consola de comando, como usuario **root**, ejecutaremos **make install** desde el directorio <HOME>**inflow/inflow**, como se muestra a continuación:

```
inflow/inflow $ make install
```

---

**Nota: Observe la siguiente Figura 11: make install:**

```
root@debian:/home/cenditel/Inflow/inflow# make install
cd src/ && make -f Makefile install
make[1]: se ingresa al directorio `/home/cenditel/Inflow/inflow/src'
make -f Makefile.Release install
make[2]: se ingresa al directorio `/home/cenditel/Inflow/inflow/src'
install -m 644 -p /home/cenditel/Inflow/inflow/src/libdotar.h /usr/include/libs
afet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetAutofilter.h /usr/include/
libsafet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetBinaryRepo.h /usr/include/
libsafet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetCipherFile.h /usr/include/
libsafet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetCondition.h /usr/include/
libsafet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetConfFile.h /usr/include/
libsafet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetConfiguration.h /usr/include/
libsafet/
install -m 644 -p /home/cenditel/Inflow/inflow/src/SafetConnection.h /usr/include/
```

Figura 2.6: **Figura 11: make install**

---

---

**Nota: ¡Si no se presenta ningún mensaje de error SAFET, entonces ya está instalado correctamente en su sistema!**

---

### 3.- Configuración

---

#### 3.1 Cargar un proyecto(Productos) en el directorio .safet/

En este tutorial montaremos nuestro primer proyecto inicial **Pysafet** en el directorio <HOME>.safet/, la cual comenzaremos a trabajar con **pysafet** para ello seguimos los siguientes paso:

##### 3.1.1 1° PRIMER PASO

- Desde la consola de comando, creamos un directorio en nuestro <HOME> por ejemplo **Pysafet**, como se muestra a continuación:

```
$ mkdir $HOME/Pysafet
```

##### 3.1.2 2° SEGUNDO PASO

- Descargamos nuestro **ProyectoInicial.tar** dando un clik al link que se muestra en la siguiente imagen:

##### 3.1.3 3° TERCER PASO

- Desde la consola de comando, copiamos el archivo que descargamos anteriormente **ProyectoInicial.tar** a nuestra carpeta <HOME>**Pysafet/**, como se muestra a continuación:

```
$ cp $HOME/Descargas/ProductosInicial.tar $HOME/Pysafet/
```

##### 3.1.4 4° CUARTO PASO

- Desde la consola de comando, accedemos al directorio <HOME>**Pysafet**, como se muestra a continuación:

```
$ cd $HOME/Pysafet
```



Figura 3.1: Click Aquí (ProyectoInicial.tar)

### 3.1.5 5° QUINTO PASO

- Desde la consola de comando, creamos un archivo (.py) por ejemplo **MontarProyectoInicial.py** dentro del directorio <HOME>**Pysafet/**, como se muestra a continuación:

```
Pysafet $ touch MontarProyectoInicial.py
```

### 3.1.6 6° SEXTO PASO

- Abrimos el archivo .py(**MontarProyectoInicial.py**), insertamos el siguiente Script de python:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

# Colocamos el nombre del proyecto (.tar)
myinflow.doLoadConfiguration("ProyectoInicial.tar")
```

### 3.1.7 7° SEPTIMO PASO

- Desde la consola de comando, ejecutamos el archivo .py(**MontarProyectoInicial.py**), como se muestra a continuación:

```
Pysafet $ python MontarProyectoInicial.py
```

---

**Nota: Al ejecutar el archivo (MontarProyectoInicial.py) se mostrara el siguiente mensaje:**

---

### 3.1.8 8° OCTAVO PASO

- Desde la consola de comando, listaremos el directorio creado <HOME>.safet/, como se muestra a continuación:

```
Pysafet $ ls $HOME/.safet/
```

---

**Nota: Se nos mostrara el siguiente contenido:**

```
auth.conf digidoc.conf flowfiles images log reports xmlrepository  
certs dtd graphs input mydb.db safet.conf
```

---

## 3.2 Guardar el proyecto del directorio .safet/

En este tutorial explicaremos con un **script python**, como guardar el proyecto que estamos trabajando del directorio **.safet/** con la extención **.tar**, para ello seguimos los siguientes paso:

### 3.2.1 1° PRIMER PASO

- Desde la consola de comando, accedemos al directorio que creamos en el paso anterior <HOME>**Pysafet**, como se muestra a continuación:

```
$ cd $HOME/Pysafet
```

### 3.2.2 2° SEGUNDO PASO

- Desde la consola de comando, creamos un archivo (**.py**) por ejemplo **GuardarProyecto.py** dentro del directorio <HOME>**Pysafet/**, como se muestra a continuación:

```
Pysafet $ touch GuardarProyecto.py
```

### 3.2.3 3° TERCER PASO

- Abrimos el archivo **.py(GuardarProyecto.py)**, insertamos el siguiente Script de python:

```
import Safet
import os

myhome = os.getenv("HOME")
```

```
mymedia = myhome + "/tmp"  
myurl = "http://localhost"  
  
myinflow = Safet.MainWindow(myhome)  
  
myinflow.setMediaPath(mymedia)  
myinflow.setHostURL(myurl)  
  
# Nombre del proyecto a guardar (.tar)  
myinflow.doSaveConfiguration("Proyecto.tar")
```

### 3.2.4 4° CUARTO PASO

- Desde la consola de comando, ejecutamos el archivo **GuardarProyecto.py**, como se muestra a continuación:

```
Pysafet $ python MontarProyectoInicial.py
```

---

**Nota:** Al ejecutar el archivo (**GuardarProyecto.py**) se mostrara el siguiente mensaje:

---

### 3.2.5 5° QUINTO PASO

- Desde la consola de comando, listamos la carpeta que estamos trabajando **Pysafet** para ver el proyecto que guardamos del directorio **.safet/**, como se muestra a continuación:

```
Pysafet $ ls
```

---

**Nota:** Archivos:

```
GuardarProyecto.py, Proyecto.tar
```

---

## 3.3 Sobre el directorio **.safet/**

Es importante destacar las características básicas del directorio **.safet/**, Para ello el comando (**tree**) ya instalado, nos permite mostrar el contenido en forma de árbol. A continuación seguimos los siguientes pasos:

### 3.3.1 1° PRIMER PASO

- Desde la consola de comando, ejecutaremos comando **tree -d \*\*** para obtener la estructura de directorio **\*\*<HOME>.safet/** como se muestra a continuación:

```
$ tree -d $HOME/.safet/
```

---

**Nota:** Nos mostrara que el contenido es de nueve (9) directorios:

---

```
.safet/ # Carpeta principal
|-- certs # Certificados de ejemploss y la CRL,
    # Certificate Revocation y lista para verificar
    # firmas elctrónicas
|-- dtd # Archivos de validación de los .xml
|-- flowfiles # Diversos ejemplos para el flujo de
    # tickets 'workflows'
|-- images # Imagenes en formato png - Portable Network Graphics
|-- input # Archivos de configuración de entrada de datos
|-- log # Archivo de registro de errores
|-- reports # Archivos de reportes de sistemas
|-- sql # Archivos de las base de datos de ejemplo PostgreSQL
|-- xmlrepository #
```

---

### 3.3.2 2° SEGUNDO PASO

- Desde la consola de comando, ejecutaremos comando **tree** para obtener los directorios con su contenido, como se muestra a continuación:

```
$ tree $HOME/.safet/
```

---

**Nota:** Nos mostrara el contenido de los nueve (9) directorios de ".safet/":

---



---

## 4.- Descripción de parámetros del directorio <HOME>.safet/

---

### 4.1 auth.conf (Archivo de autenticación y autorización)

En este archivo (**auth.conf**) se especifican los valores de (**autorización, autenticación y conexión**) y su tipo de gestor de base de datos admitidos.

Los tipos de base de base de dato que admite son:

- Psql : Postgresql.
- Sqlite : sqlite3.
- Mysql:

A continuación se explicara las secciones de configuración:

#### 4.1.1 1º PRIMERA SECCION [Database]

En esta sección se configura las fuentes de datos. Una fuente de datos esta asociada a una conexión con una base de datos relacional. Es posible tener varias fuentes de datos, cada una representada por un índice.

A continuación el siguiente block:

[Database]

```
# Información de configuración de fuentes de datos.
# Una fuente de datos está asociada a una conexión con una base
# De datos relacional. Es posible tener varias fuentes de datos, cada
# una representada por un índice.

# Especifica el número de fuentes de datos diferentes
database.datasources = 1
```

---

**Nota:** Esté elemento (**database.datasources**) me indica el número de base de datos, de aquí en adelante todos los elementos que se requiera estará identificado por el índice asociado a una base de datos.

---

```
# Para realizar la conexión de base de datos asociada a una fuente de datos
# Se requieren varios elementos:
```

```
# Nombre de la fuente de datos 1
database.sourcename.1 = nombre_fuente_datos_1

# Especifica el controlador asociado a la fuente de datos
# Tipo de gestor de base de datos admitidos
database.driver.1 = controlador_fuente_datos_1
```

---

**Nota: Ejemplo:** database.driver.1 = psql

---

```
# Especifica si la fuente de datos esta activa o no
database.actived.1 = on

# Especifica el nombre de host para la fuente de datos
database.host.1 = host_base_de_datos
```

---

**Nota: Ejemplo:** 127.0.0.0

---

```
# Especifica el nombre de la base de datos para la fuente de datos
database.db.1 = nombre_database .1

# Especifica el puerto de conexión para la base de datos
database.port.1 = puerto_database.1

# Especifica el nombre de usuario para la fuente de datos
database.user.1 = usuario_database.1

# Especifica la contraseña de acceso para la fuente de datos
database.password.1 = contrasena_database.1
```

### 4.1.2 2° SEGUNDA SECCION [Roles]

En esta sección se definen los roles para un sistema como tal. Un rol comprende dos : command:(2) elementos:

- **roles.name.n** = nombre del rol n.
- **roles.description.n** = descripción del rol n.

**A continuación el siguiente block:**

```
[Roles]
```

```
roles.name.1 = Administrador
roles.description.1 = usuario(s) que administra el sistema

roles.name.2 = rol_1
roles.description.2 = descripción del rol

roles.name.3 = rol_2
roles.description.3 = descripción del rol
```

```
•  
•  
•  
  
roles.name.n = rol_n  
roles.description.3 = descripción del rol
```

---

**Nota: Ejemplo:** Acciones asociadas al grupo o al usuario con el rol\_1, rol\_2,rol\_n.

---

#### 4.1.3 3° TERCERA SECCION [Auth]

En esta sección se agrega los usuarios del sistemas, la definición de un usuario comprende cuatro (**4**) elementos:

- **auth.account.n** = Nombre del usuario n.
- **auth.pass.n** = Contraseña del usuario n cifrada en md5.
- **auth.realname** = Nombre apellido y correo del usuario n separada por espacio.
- **auth.role.n** = Rol del usuario n, este rol debe estar definido en el sección [Roles].

A continuación el siguiente block:

```
[Auth]
```

```
auth.account.1 = usuario 1  
auth.pass.1 = 0f885ebbc5a6c77dac0c319a7411f6039496f06f  
auth.realname.1 = Nombre_apellido_usuario1 correo_usuario1  
auth.role.1 = rol del usuario  
  
auth.account.2 = usuario1  
auth.pass.2 = 22cd2d1c596f4091e248aff0e4aa0d47c84b2b36  
auth.realname.2 = Nombre_apellido_usuario1 usuario1@mail.com  
auth.role.2 = rol del usuario  
  
•  
•  
•  
  
auth.account.n = usuarion  
auth.pass.n = ddc6fdd484d5a71b9619beb8b19a7ea06980d8ff  
auth.realname.n = Nombre_apellido_usuarion usuarion@mail.com  
auth.role.n = Desarrollador
```

---

**Nota: Ejemplo:**

- **auth.account.1:** admin
- **auth.pass.1:** contraseña del admin en md5
- **auth.realname.1:** admin@mail.com
- **auth.role.1:** Administrador

### 4.1.4 4° CUARTA SECCION [Permises]

En esta sección se van a definir los permisos de los usuarios en función de los roles y acciones que puedan ejecutar en el sistema.

La definición de cada permisos comprende cuatro: command:(4) elementos:

- **permises.operation.n** = nombre de la operación n a ejecutar (esta acción debe estar definida en el archivo deftrac.xml que más adelante se explicara).
- **permises.accounts.n** = nombre del o los usuario(s) que ejecutara(n) la operación (esta usuario debe estar definido en la sección [Auth]). Si son varios usuarios deben estar separados por punto y como (:). **Ejemplo:** permises.accounts.n = usuario1;usuario2;usuaron.
- **permises.types.3** = tipo de acción a ejecutar, entre las acciones tenemos: read,execute y modify deben ir separadas por punto y como (:). **Ejemplo:** permises.types.3 = read;execute;modify.
- **permises.roles.4** = rol que puede ejecutar esta acción, de esta manera se puede indicar a un grupo de usuarios que tenga un rol en específico ejecutar la acción sin la necesidad de especificarlo en el elemento **permises.accounts.n**, los roles debe estar definidos en la sección **[Roles]**.

A continuación el siguiente block:

```
[Permises]

permises.operation.1 = operacion_1
permises.accounts.1 = usuario1;admin
permises.types.1 = read;execute;modify
permises.roles.1 = Administrador;rol_1;rol_2

permises.operation.2 = operación_2
permises.accounts.2 = admin
permises.types.2 = read;execute;modify
permises.roles.2 = Administrador

.
.
.

permises.operation.n = operacion_n
permises.accounts.n = admin
permises.types.n = read;execute;modify
permises.roles.3 = Administrador;rol_1;rol_2

# Operaciones de Consulta, Lista de datos y gráficos

permises.operation.24 = Listar_datos
permises.accounts.24 = admin
permises.types.24 = read;execute;modify
permises.roles.24 = Desarrollador;Administrador

permises.operation.25 = Listar_datos_con_autofiltro
permises.accounts.25 = admin
```

```
permises.types.25 = read;execute;modify
permises.roles.25 = Administrador

# Viñetas

permises.operation.31 = Formulario
permises.accounts.31 = admin
permises.types.31 = read;execute;modify
permises.roles.31 = Administrador;Desarrollador

permises.operation.32 = Consulta
permises.accounts.32 = admin
permises.types.32 = read;execute;modify
permises.roles.32 = Administrador;Desarrollador
```

## 4.2 safet.conf (Archivo de configuración inicial)

En este archivo se especifican los parámetros generales de flujo de trabajo para el funcionamiento de la librería PySafet.

**A continuación los siguientes parámetros generales:**

### 4.2.1 1° PRIMER PARAMETRO [Workflow]

En esta sección se define el color del flujo y Sección para la interfaz Web.

**A continuación el siguiente block:**

```
[Workflow]

# Establece el color por defecto
defaultcolor = white

# Establece el color activo
activecolor = white

# Establece el color pasivo
tracicolor = white
```

---

**Nota: Admite:** white,black,blue y **formato** RGB #000000

---

### 4.2.2 2° SEGUNDO PARAMETRO [Operators]

En esta sección se establecen las variables de la imágenes de los operadores que son definido en un flujo de trabajo, entre las variables se encuentra:

- **split.xor** = Se activa un enlace saliente una vez terminada la tarea.
- **split.and** = Activa todos los vínculos salientes una vez terminada esta tarea
- **split.or** = Se utiliza para desencadenar algunos, pero no necesariamente todos los flujos salientes a otras tareas.
- **join.xor** = Esta tarea se activa cada vez que un nuevo enlace se ha activado.
- **join.and** = Esta tarea se activa cuando todos los vínculos se han activado.
- **join.or** = Esta tarea se activa cuando almeno un vinculo se ha activado.

### A continuación el siguiente block:

```
[Operators]

# Operador split.xor
split.xor = imgs/xor.png

# Operador split.and
split.and = imgs/and.png

# Operador split.or
split.or = imgs/or.png

# Operador join.xor
join.xor = imgs/join-xor.png

# Operador join.and
join.and = imgs/join-and.png

# Operador join.or
join.or = imgs/join-or.png

# Ningun operador
none = imgs/none.png
```

---

**Nota:** Archivo de imagen se encuentra en el directorio <HOME>.safet/imgs

---

### 4.2.3 3º TERCER PARAMETRO [Log]

En esta sección se configura las diferentes opciones de la información sobre el registro de los eventos. Entre las opciones tenemos:

- **log.filepath** = Establece la ruta absoluta donde reside el archivo de registro ejemplo: <HOME>.safet/.log
- **log.debug** = Habilita la opción de registro de mensajes de depuración (on/off)
- **log.action** = Habilita la opción de registro de de mensajes de acciones (on/off)
- **log.warning** = Habilita la opción de registro de de mensajes de advertencia (on/off)

- **log.error** = Habilita la opción de registro de mensajes de error (on/off)
- **log.output** = Salida del registro de ejecución, las opciones son: default (file), file y screen para pantalla.

#### A continuación el siguiente block:

[Log]

```
# Establece la ruta absoluta donde reside el archivo de registro
log.filepath = /home/pbuitrago/.safet/log

# Habilita la opción de registro de mensajes de depuración
log.debug = on

# Habilita la opción de registro de mensajes de acciones
log.action = on

# Habilita la opción de registro de mensajes de advertencia
log.warning = on

# Habilita la opción de registro de mensajes de error
log.error = on

# Salida del registro de ejecución,
# las opciones son: default (file), file y screen
log.output = file
```

#### 4.2.4 4° CUARTO PARAMETRO [XmlRepository]

Información sobre repositorio de documentos firmados electrónicamente bajo el formato “**bdoc**” (firma electrónica xml) asociados a flujos de trabajos.

#### A continuación el siguiente block:

[XmlRepository]

```
# En caso de utilizar un repositorio de documentos XML remoto se debe
# especificar la información de acceso al mismo a través de un servicio web
xmlrepository.remote.ip = 150.187.36.6
xmlrepository.remote.urlwebservice = http://150.187.36.6/cgi-bin/safet

# Establece el tipo de repositorio de documentet_auth_conf XML.
# Valores posibles:
# 1.- (dir): para repositorio basado en un directorio local
# 2.- (dbxml): para repositorio DBXML
xmlrepository.type = dir

# Establece la ruta absoluta al repositorio XML
xmlrepository.path = <HOME>.safet/xmlrepository

# Establece el nombre del repositorio XML
xmlrepository.name = container1
```

#### 4.2.5 5° QUINTO PARAMETRO [Input]

Opciones para la entrada/modificación de datos

**A continuación el siguiente block:**

```
[Input]
# Establece la ruta absoluta para el directorio
# en la entrada/modificación de datos
input.path = <HOME>.safet/input

#input.file = deflista.xml
input.file = deftrac.xml
```

#### 4.2.6 6° SEXTO PARAMETRO [System]

Opciones generales referentes al funcionamiento (**inflow**) del tipo de aplicación : Consola/Web/Gráfica.

**A continuación el siguiente block:**

```
[System]
# Información referente a la base de datos o repositorio
# para el acceso de la librería
system.evalexit = on
```

---

**Nota:** Preguntar al salir de la aplicación “inflow”.

---

#### 4.2.7 7° SEPTIMO PARAMETRO [Widgets]

Objetos para entrada de datos.

**A continuación el siguiente block:**

```
[Widgets]
# Lector de archivo en el sistema
# lista de archivos de acceso rápido
widgets.getfilewidget.* = <HOME>.safet/flowfiles/mensajes.xml

# Introducción para el widget "wiki"
widgets.texteditwidget.wiki.leftbold = ''
widgets.texteditwidget.wiki.rightbold = ''
widgets.texteditwidget.wiki.leftitalic = ''
widgets.texteditwidget.wiki.rightitalic = ''
widgets.texteditwidget.wiki.leftunderline = __
widgets.texteditwidget.wiki.rightunderline = __
```

## 4.2.8 8° OCTAVO PARAMETRO [GeneralOptions]

Esta sección incluye parámetro de uso de opciones generales.

**A continuación el siguiente block:**

[GeneralOptions]

```
# Consultas del archivo productos.xml
generaloptions.consoleactions.* = Gráfico Mensajes;
                                operacion:Generar_gráfico_coloreado
Cargar_archivo_flujo: <HOME>.safet/flowfiles/productos.xml

# Consultas del archivo productos.xml con clave
generaloptions.consoleactions.* = Grafico por clave;
                                operacion:Generar_gráfico_para_clave
Cargar_archivo_flujo: <HOME>.safet/flowfiles/productos.xml Clave: 1

# Titulo para el gráfico de flujo de trabajo
generaloptions.currentflowtitle = Tareas Proyecto SAFET

# Opciones On (Incluir en el gráfico, No incluir),
# Off si no se quiere incluir
generaloptions.currentflowtitle.font = Dejavu Sans

# Tamaño de la fuente para el texto de informacion
# en cada flujo de trabajo
generaloptions.currentflowtitle.size = 18

# Separación de la fuente para el texto de informacion
# en cada flujo de trabajo
generaloptions.currentflowtitle.separation = 10

# Directorio donde se almacenan los archivos
# de salida grafos(.png,svg)
generaloptions.dir.media = <HOME>PySafet
generaloptions.currentflowtitle.include = off

# Ver http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# Zona horaria
generaloptions.timezone = America/Caracas

# mostrar información estadística en el grafo
generaloptions.extrainfo.showonly = off
generaloptions.extrainfo.showhumanize = on
```

**Nota:** opciones:

- Mostrar el dialogo para los parámetros para cada documento de flujo de trabajo
- Se colocan las opciones generales del sistema SAFET (Lista de acciones guardadas, entre otros)

### 4.2.9 9° NOVENO PARAMETRO [Stats]

En este parámetro se usa para la estadística de grafos y sus valores para el cálculo de estadísticas.

**A continuación el siguiente block:**

```
[Stats]

#Activar la recolección de estadísticas
stats.actived = off

#Fecha de inicio de cálculo de estadística,
# (*) significa que no hay fecha colocada
stats.fromdate = *

#Fecha de fin de cálculo de estadística,
#(*) Significa que no hay fecha colocada
stats.todate = *
```

### 4.2.10 10° DECIMO PARAMETRO [Libdigidoc]

Este parámetro es para la firma electrónica y se establece variables de configuración para la librería Libdigidoc utilizada por Libsafet.

**A continuación el siguiente block:**

```
[Libdigidoc]

# Ruta del archivo de configuración digidoc para
# usarlo con protocolo OCSP
libdigidoc.configfi:q
lepath = <HOME>.safet/digidoc.conf

# directorio donde se almacenan los certificados de validación
libdigidoc.x509storepath = <HOME>.safet/certs

# Tipo de validación de firma de Digidoc
# "ocsp" : via protocolo OCSP, "keystore": Repositorio
# de claves del navegador Mozilla / Firefox
#libdigidoc.validationtype = keystore

# Tipo de archivos "MIME" permitidos
libdigidoc.mimestypes.* = application/vnd.sun.xml.writer sxw
libdigidoc.mimestypes.* = application/vnd.sun.xml.writer.template stw
libdigidoc.mimestypes.* = application/vnd.sun.xml.writer.global sxg
libdigidoc.mimestypes.* = application/vnd.stardivision.writer sdw vor
libdigidoc.mimestypes.* = application/vnd.stardivision.writer-global sgl
libdigidoc.mimestypes.* = application/vnd.sun.xml.calc sxc
libdigidoc.mimestypes.* = application/vnd.sun.xml.calc.template stc
libdigidoc.mimestypes.* = application/vnd.stardivision.calc sdc
libdigidoc.mimestypes.* = application/vnd.sun.xml.impress sxi
libdigidoc.mimestypes.* = application/vnd.sun.xml.impress.template sti
libdigidoc.mimestypes.* = application/vnd.stardivision.impress sdd sdp
```

```
libdigidoc.mimestypes.* = application/vnd.sun.xml.draw sxd
libdigidoc.mimestypes.* = application/vnd.sun.xml.draw.template std
libdigidoc.mimestypes.* = application/vnd.stardivision.draw sda
libdigidoc.mimestypes.* = application/vnd.sun.xml.math sxm
libdigidoc.mimestypes.* = application/vnd.stardivision.math smf
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.text odt
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.text
                           -template ott
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.text-web oth
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.text-master odm
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.graphics odg
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.graphics
                           -template otg
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.presentation
                           odp
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.presentation
                           -template otp
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.spreadsheet ods
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.spreadsheet
                           -template ots
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.chart odc
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.formula odf
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.database odb
libdigidoc.mimestypes.* = application/vnd.oasis.opendocument.image odi
libdigidoc.mimestypes.* = application/pdf pdf
libdigidoc.mimestypes.* = application/xml xml
libdigidoc.mimestypes.* = text/plain txt
libdigidoc.mimestypes.* = text/css css
libdigidoc.mimestypes.* = text/xml xml
libdigidoc.mimestypes.* = text/html html
libdigidoc.mimestypes.* = application/x-gzip tgz
libdigidoc.mimestypes.* = application/zip zip
```

#### 4.2.11 11° DECIMO PRIMERO PARAMETRO [Autofilter]

Este parámetro sirve para opciones de los Auto filtros y para realizar clasificaciones automáticas.

**A continuación el siguiente block:**

```
[Autofilter]
```

```
autofilter.datetime.period = 672
```

#### 4.2.12 12° DECIMO SEGUNDO PARAMETRO [DefaultValues]

Opciones que se toman por defecto.

**A continuación el siguiente block:**

```
[DefaultValues]
```

```
defaultvalues.report = yes
defaultvalues.digidoc.manifest = Investigador
defaultvalues.digidoc.city = Mérida
defaultvalues.digidoc.state = Mérida
defaultvalues.digidoc.country = VE
defaultvalues.digidoc.zip = 5101
defaultvalues.panel.info.secondsstohide = 4000
```

### 4.2.13 13° DECIMO TERCERO PARAMETRO [Reports]

Opción para la generación de reportes y configuraciones para mostrar información en la aplicación **Inflow**.

**A continuación el siguiente block:**

```
[Reports]
# tipo de protocolo (file, http, ftp, entre otros)
reports.protocol = file

# ruta para obtener la plantilla (HTML)
reports.path = <HOME>.safet/reports

# nombre de la plantilla HTML + AJAX
reports.general.template = <HOME>.safet/reports/sf_plantillaLista01.html

# nombre de la plantilla para documento a firmar
reports.documenttosign.template = <HOME>.safet/reports/
sf_plantillaFirma01.html

# transformar fecha numerica a formato de fecha
reports.options.datetransform = on
reports.options.dateformat = dd/MM/yyyy
```

### 4.2.14 14° DECIMO CUARTO PARAMETRO [Graphs]

Opciones Para los gráficos de flujo de trabajo

**A continuación el siguiente block:**

```
[Graphs]
#Opciones para el texto de información en cada flujo de trabajo
# opciones on/off
graphs.infotext.include = on

#Formato para el texto de información en cada flujo de trabajo
# (el %date indica Fecha y %time Hora, %datetime Hora y Fecha)
graphs.infotext.format = Generado en %datetime

#Nombre de la fuente para el texto de información en
# cada flujo de trabajo
graphs.infotext.font = Book Antiqua
```

```
#Tamaño de la fuente para el texto de información  
# en cada flujo de trabajo  
graphs.infotext.size = 14  
  
#Separación de la fuente para el texto de información  
# en cada flujo de trabajo  
graphs.infotext.separation = 30
```

#### 4.2.15 15° DECIMO QUINTO PARAMETRO [Ftp]

Opción para la trasferencia de archivos utilizando el protocolo FTP.

**A continuación el siguiente block:**

```
[Ftp]
```

```
ftp.default.server = victorbravo.info  
ftp.default.account = victorrbravo
```

#### 4.2.16 16° DECIMO SEXTO PARAMETRO [ExtraInfo]

Parámetros de la información extra de cada nodo de grafos.

**A continuación el siguiente block:**

```
[ExtraInfo]
```

```
extrainfo.infotext.fields = owner,porcentage,completed  
extrainfo.infotext.separator =  
  
extrainfo.infotext.completed = [*]  
  
# Mostrar campos coloreados (lo ya pasados)  
extrainfo.coloured = yes  
  
# Mostrar estadísticas al final del grafo (resumen)  
extrainfo.summarize = yes  
  
extrainfo.title = Porcentaje:  
  
extrainfo.formula = sumall  
  
extrainfo.pattern = #PAR0#CLA00-9#CLA1+#PAR1#SLA%
```

#### 4.2.17 17° DECIMO SEPTIMO PARAMETRO [Plugins]

Opciones para las librerías adicionales Plugins (**Complementos o componentes**).

**A continuación el siguiente block:**

[Plugins]

```
#Ruta donde se encuentra las librerías
#           adicionales (plugins)
plugins.path = /usr/lib/libsafet
```

### 4.2.18 18° DECIMO OCTAVO PARAMETRO [Plugins.Graphviz]

Componentes para la generación de grafos utilizando la biblioteca Graphviz.

A continuación el siguiente block:

[Plugins.Graphviz]

```
#Formato de archivo de salida de grafo,
# valores posibles: svg/png/gif
plugins.graphviz.graphtype = svg

# Información a mostrar en cuadro de información
# extra Porc,Tokens,Total,InfoText,InfoDate
plugins.graphviz.extrainfo.show = Porc,Tokens,Total,InfoText,InfoDate

#Color activo de para ser utilizado en los gráficos
plugins.graphviz.task.fillcolor = #f1f1f1

#Color activo de la linea para ser utilizado en los gráficos
plugins.graphviz.task.color = black

#Atributo utilizado en la estadística,
# opciones posibles (Color/Size/Line/Custom)
plugins.graphviz.stats.attribute = Color

# Tamaño máximo para la estadística de (Tamaño Máximo)
plugins.graphviz.stats.sizemax = 2

# Tamaño mínimo para la estadística de (Tamaño Mínimo)
plugins.graphviz.stats.sizemin = 1

# Color para dibujar estadística
plugins.graphviz.stats.colorgradient = yellow

# Color del texto para cuadro de estadística
plugins.graphviz.stats.fontcolor = black

# Color de fondo para cuadro de estadística
plugins.graphviz.stats.fillcolor = antiquewhite

# Estilo de la línea del cuadro de estadística
plugins.graphviz.stats.style = dashed

# Mostrar cuadro de estadística
plugins.graphviz.showstats = yes
```

```
#Dirección del grafo TB (Arriba-Abajo) LR (Izquierda-Derecha)
plugins.graphviz.graph.rankdir = TB

# Tamaño del fuente para todos los nodos
plugins.graphviz.graph.fontsize = 12

# Separador del rank
plugins.graphviz.graph.ranksep = 0.5 equally

# Figura para la tarea (Task)
plugins.graphviz.task.shape = box

# Estilo de la Figura para la tarea
#      (Task) filled, bold, dotted, empty
plugins.graphviz.task.style = filled

#Color activo de para ser utilizado en los gráficos
plugins.graphviz.condition.fillcolor = #FFFFFF

#Color activo de la línea para ser utilizado en los gráficos
plugins.graphviz.condition.color = black

# Figura para la (Condition) (box, ellipse, circle, etc.)
plugins.graphviz.condition.shape = ellipse

# Estilo de la Figura para la tarea (Condition)
plugins.graphviz.condition.style = filled

# Mostrar la información extra solo donde existan tokens (fichas)
plugins.graphviz.extrainfo.showwheretokens = on

#Mostrar la estadística en
#el nodo "FINAL", valores admitidos (on/off)
plugins.graphviz.extrainfo.showfinal = on
```



## 5.- API de Safet para el lenguaje de Python

---

### 5.1 Clase MainWindow(Safet.MainWindow)

#### 5.1.1 Constructor (MainWindow (home-usurio))

**home-usuario:** Directorio del usuario donde se encuentra la carpeta **.safet/**, como se muestra en el siguiente ejemplo de script python:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

# MainWindow
myinflow = Safet.MainWindow(myhome)
```

#### 5.1.2 Listado de myinflow

##### 1. setHostURL (u) [Procedimiento]:

Coloca el url del servidor web actual, como se muestra en el siguiente ejemplo de script python:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

# setHostURL
myinflow.setHostURL(myurl)
```

##### 2. setMediaPath (m) [Procedimiento]:

Coloca la ruta del directorio de los archivos gráficos,, como se muestra en el siguiente ejemplo de **script python:**

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

# setMediaPath
myinflow.setMediaPath(mymedia)
```

**3. addInfoGraphDateText () [Retorna cadenas (String)]:**

Agregar la información de fechas al grafo.

**4. addNodeToXMLWorkflow (fname,beforenode = “”,nodename = “newnode”,isparallel = false,options””,query = “”,nodetitle = “”,documentsource = “”) [Retorna cadenas (String)]:**

Agregar un nodo al grafo.

**5. autoComplete (path) [Retorna cadenas (String)]:**

Retorna la lista de autocompletación.

**6. changeConnXMLWorkflow (fname,currnode,nextnode,newnode,newoptions = “”,newquery =””) [Retorna cadenas (String)]:**

Cambia una conexión entre nodo.

**7. checkUserRegister (fullname,account,email,passone,passtwo) [Retorna cadenas (String)]:**

Realiza el registro de un nuevo usuario.

**8. createBdoc (content) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Crea un documento “bdoc” vacío.

**9. currentDATA () [Retorna cadenas (String)]:**

Devuelve los datos (DATA) actual.

**10. currentError () [Retorna cadenas (String)]:**

Devuelve el error actual, como se muestra en el siguiente ejemplo de **script python:**

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)
myinflow.setMediaPath(mymedia)
```

```
myinflow.setHostURL(myurl)

myconsult = u"operacion>Listar_datos \
Cargar_archivo_flujo:/home/cenditel/.safet/flowfiles/productos.xml
Variable: vIniciado"

result = myinflow.login("usuario", "usuario")

if not result:
    #currentError()
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()
```

#### 11. **currentGraphTitle () [Retorna cadenas (String)]:**

Devuelve el título del último gráfo generado.

#### 12. **currentJSON () [Retorna cadenas (String)]:**

Devuelve la información en formato JSON de la última acción generada, como se muestra en el siguiente ejemplo de **script python**:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)
myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

myconsult = u"operacion>Listar_datos \
Cargar_archivo_flujo:/home/cenditel/.safet/flowfiles/productos.xml
Variable: vIniciado"

result = myinflow.login("usuario", "usuario")

if not result:
    #currentError()
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

# currentJSON()
print u"%s" % (myinflow.currentJSON())
```

#### 13. **currentTable () [Retorna cadenas (String)]:**

Obtiene los datos en forma de tabla del último gráfo generado.

#### 14. **delNodeToXMLWorkflow (fname,nodename) [Retorna cadenas (String)]:**

Elimina un nodo de un gráfo.

#### 15. **doLoadConfiguration (fileName)[Procedimiento]:**

Carga un archivo de proyecto, como se muestra en el siguiente ejemplo de **script python**:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

# doLoadConfiguration
myinflow.doLoadConfiguration("Proyecto.tar")
```

**16. doSaveConfiguration (fileName)[Procedimiento]:**

Guarda un archivo de proyecto (en el directorio **.safet/** del usuario actual), como se muestra en el siguiente ejemplo de **script python**:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

# doSaveConfiguration
myinflow.doSaveConfiguration("usuario.tar")
```

**17. doSaveGraph (mypars) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Guarda los parámetros de último gráfo generado.

**18. generateFormFooter (o) [Retorna cadenas (String)]:**

Imprime el HTML del pie de pagina general para **websafet**.

**19. generateFormHead (o) [Retorna cadenas (String)]:**

Imprime el HTML de la cabecera general para **websafet**.

**20. generateModifyHTML (operation,fieldname,key,secondkey,form) [Retorna cadenas (String)]:**

Genera una consulta “AJAX” para los campos de combos de selección.

**21. getFlowParameters (flowfilename) [Retorna cadenas (String)]:**

Obtiene una lista de parámetros dado un nombre de gráfo.

**22. getInfoOfUser (user) [Retorna cadenas (String)]:**

Obtiene la información de un usuario.

**23. hostMediaPath () [Retorna cadenas (String)]:**

Obtiene la ruta del directorio de archivos multimedia.

24. **hostURL () [Retorna cadenas (String)]:**

Obtiene el url del host (**websafet**).

25. **inputPath () [Retorna cadenas (String)]:**

Obtiene el directorio (ruta) de las aplicaciones actual (**.safet**).

26. **lastInfoGraph () [Retorna cadenas (String)]:**

Obtiene información sobre el último gráfo generado.

27. **loadReportTemplate (json,filename = “”,nameoperation = “Listar\_datos”) [Retorna cadenas (String)]:**

Carga una plantilla para generar una salida HTML (**websafet**).

28. **log (message)[Procedimiento]:**

Escribe en el “log” de safet <HOME>.safet/log/safet.log.

29. **login (name,pass) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Inicia una sesión **PySafet**, como se muestra en el siguiente ejemplo de **script python**:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)
myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

myconsult = u"operacion>Listar_datos \
    Cargar_archivo_flujo:/home/cenditel/.safet/flowfiles/productos.xml
    Variable: vIniciado"

#login
result = myinflow.login("usuario", "usuario")
```

30. **logout () [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Sale una de las sesión **PySafet**.

31. **mediaPath () [Retorna cadenas (String)]:**

Obtiene las rutas del directorio “media” utilizando para generar los gráficos (.svg/.png).

32. **menuCommands () [Retorna cadenas (String)]:**

Obtiene la lista de acciones en formato HTML.

33. **menuForm (o) [Retorna cadenas (String)]:**

Obtiene un formulario escrito en HTML correspondiente a una acción en el archivo <HOME>.safet/input/deftrac.xml.

34. **postAction () [Retorna cadenas (String)]:**

Obtiene la acción posterior a ejecutar en una operación.

35. **registerLogin (user) [Procedimiento]:**

Escribe en el registro de **PySafet** el inicio de una sección.

36. **registerLogout (user) [Procedimiento]:**

Escribe en el registro de **PySafet** el inicio de la salida de una sesión.

37. **sendCheckEmail (user,plink) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Envía un correo de chequeo para el registro de un usuario.

38. **setConffileValues (values) [Procedimiento]:**

Coloca los valores de configuración.

39. **setHostMediaPath (m) [Procedimiento]:**

Coloca la ruta para los archivo de medias.

40. **setTemplatePath (t) [Procedimiento]:**

Coloca el directorio de los archivos de plantilla (templates).

41. **templatePath () [Retorna cadenas (String)]:**

Obtiene el directorio para los archivos de plantilla (templates).

42. **toInputConsole (action, withpermises = true) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Ejecuta una operación de consulta X en el archivo **defconsole.xml**, como se muestra en el siguiente ejemplo de **script python**:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)
myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

myconsult = u"operacion>Listar_datos \
Cargar_archivo_flujo:/home/cenditel/.safet/flowfiles/productos.xml
Variable: vIniciado"

# toInputConsole
result = myinflow.toInputConsole(myconsult)
```

#### 43. **toInputForm (action, withpermises = true) [Retorna cadenas (String)]:**

Se ejecuta una operación de entrada de datos (**Formulario**) descrita en el archivo de **deftrac.xml**, como se muestra en el siguiente ejemplo de script **python**:

```
import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)
myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

myconsult = u"operacion:agregar_mensaje Titulo: hola mundo"

# toInputForm
result = myinflow.toInputForm(myconsult)
```

#### 44. **toInputUsers (action) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]:**

Ejecuta una acción de gestión de usuarios descrita en el archivo **defusers.xml**.

## 5.2 Clase SafetXmlObject (Safet.SafetXmlObject)

### 5.2.1 Clase SafetXmlObject

```
{
%TypeHeaderCode
#include "../websafet/src/SafetXmlObject.h"
%End

public:
    SafetXmlObject();
    bool syncAttributes(const QDomElement&);

};
```

## 5.3 Clase SafetDocument

### 5.3.1 Clase SafetDocument

**SafetXmlObject:** Esta clase modela un documento de una variable en **PySafet** (el documento puede ser firmado electrónicamente).

### 5.3.2 Listado de myinflow

1. **getCertFileList (ext = ”.pem”)** [Retorna cadenas (String)]:

Obtiene la ruta del certificado usado para firmar el documento.

2. **numberOfDataFileOnOpenXAdESContainer ()** [Retorna un entero (int)]:

Número de archivos que contienen el documento firmado.

3. **numberOfSignaturesOnOpenXAdESContainer ()** [Retorna un entero (int)]:

Número de firmas que contienen el documento firmado.

4. **getDataFileName (index = 0)** [Retorna cadenas (String)]:

Nombre del archivo contenido de índice “index”.

5. **getDataFileLength (index = 0)** [Retorna un (long)]:

Tamaño del archivo contenido de índice “index”.

6. **getDataFileMimeType (index = 0)** [Retorna cadenas (String)]:

Tipo de archivo contenido de índice “indice”.

7. **numberOfDataFileAttributes (index = 0)** [Retorna un entero (int)]:

Atributo del archivo contenido de índice “indice”.

8. **getXmlQuery (query,dcount,info = ””)** [Retorna cadenas (String)]:

Obtiene consulta XML.

9. **addSignatureToExistingDigidocFile (keyFile,passwd,certFile,inFile,outFile)** [Retorna un entero (int)]:

Agrega una firma al archivo.

10. **signWithPrivateKeyOnFile (keyFile, passwd, certFile, inFile, outFile, manifest, city, state, zip,country, notaryUrl, proxyHost, proxyPort)**[Retorna un entero (int)]:

Firma electrónicamente un archivo con un certificado PCKS#LZ.

11. **initPKCS11Lib (libName)** [Procedimiento]:

Inicia el manejador de la tarjeta inteligente.

12. **initDigidocConfigStore (configFile)** [Retorna un entero (int)]:

Inicializa la configuración.

13. **verifySignMadeWithSmartCard (fileName)** [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]

14. **verifySignMadeWithSmartCard (fileName,signatureIndex)** [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]

15. **verifySignMadeWithSmartCard (fileName,listOfSigners,isneg,op = SafetDocument::AND )** [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]

```
#      (13-15) Verifica una firma electrónica que se realiza utilizando
# la tarjeta inteligente.
```

16. **testSmartCardDriver (slot) [Procedimiento]:**

Realiza una prueba a la conexión con la tarjeta inteligente.

17. **getCN (signatureIndex) [Retorna cadenas (String)]:**

Retorna el nombre común de la firma “signature Index” (commun name) .

18. **getSignerIndex (cn) [Retorna un entero (int)]:**

Retorna el índice de firma del correspondiente nombre camún (ch).

19. **getCommonNameOfSigners () [Retorna cadenas (String)]:**

Obtiene una lista con el nombre camún de los firmantes del archivo.

20. **getNumberOfPrivateKeys (slot,passwd,libName) [Retorna un entero (int)]:**

Obtiene el número de claves privadas utilizados para el archivo.

21. **writeFileToDisk (string,name) [Retorna cadenas (String)]:**

Escribe en otro archivo el documento.

22. **returnFileToString (pathFileName) [Retorna cadenas (String)]:**

Convierte el archivo a una cadena.

23. **decryptDocument (inFile,outputFile,pin) [Retorna un entero (int)]:**

Descifra (decrypta) el archivo.

24. **decryptString (inFile,pin)[Retorna cadenas (String)]:**

Descifra (decrypta) una cadena.

25. **getTempNameFiles (n) [Retorna cadenas (String)]:**

Crea “n” archivos temporales.

26. **getCommonNameFromCertificate (certPath) [Retorna cadenas (String)]:**

Obtiene el nombre camún (CN) de un archivo de certificación.

27. **getCountryOfSignature (index = 0) [Retorna cadenas (String)]:**

Obtiene el valor de país “country” especificado para la firma electrónica de índice “Index”.

28. **getStateOfSignature (index = 0) [Retorna cadenas (String)]:**

Obtiene el valor del Estado (STATE) departamento o región especificando para la firma electrónica de índice “Indice”.

29. **getCityOfSignature (index = 0) [Retorna cadenas (String)]:**

Obtiene el valor de la Ciudad.

30. **getPostalCodeOfSignature (index = 0) [Retorna cadenas (String)]:**

Obtiene el valor de código Postal.

31. **getCountOfRoles (index = 0) [Retorna un entero (int)]:**

Obtiene el número de roles (nominación de cargo del firmante).

32. **getRole (index = 0,roleIndex = 0) [Retorna cadenas (String)]:**

33. **getSingingTime (index = 0) [Retorna cadenas (String)]**

34. **getSingingTimeOnlyDate (index = 0) [Retorna cadenas (String)]**

35. **getSingingTimeOnlyHour (index = 0) [Retorna cadenas (String)]**

36. **getSignatureType (index = 0) [Retorna cadenas (String)]**

37. **getSignatureFormat (index = 0) [Retorna cadenas (String)]**

38. **getSignerCertificateIssuerName (index = 0) [Retorna cadenas (String)]**

39. **getSignerCertificateSerial (index = 0) [Retorna cadenas (String)]**

40. **getValidAt (index = 0) [Retorna cadenas (String)]**

41. **getValidUntil (index = 0) [Retorna cadenas (String)]**

# (32-42) Obtiene datos del certificado electrónico usado para la firma  
# de índice "Índice".

42. **getPathOfSafetDocument () [Retorna cadenas (String)]:**

Obtiene la ruta donde se guarda los documentos firmados.

43. **setPathOfSafetDocument (path) [Procedimiento]**

44. **getSubjectDN (index = 0) [Retorna cadenas (String)]**

45. **getIssuerDN (index = 0) [Retorna cadenas (String)]**

46. **policies (index = 0) [Retorna cadenas (String)]**

47. **getPublicKeyAlgorithm (index = 0) [Retorna cadenas (String)]**

48. **getPublicKeyLength (index = 0) [Retorna cadenas (String)]**

49. **writeX509ToFile (path,PEM,index = 0) [Procedimiento]**

QByteArray versionNumber(int index = 0)

QByteArray serialNumber(int index = 0);

QByteArray toHex( const QByteArray &in, QChar separator );

QByteArray authorityKeyIdentifier(int index = 0);

QByteArray subjectKeyIdentifier(int index = 0);

## 5.4 Clase SafetVariable

Clase **SafetVariable** : SafetXObject

### 5.4.1 Listado de myinflow

1. **addChild** (SafetXObject) [Procedimiento]
  2. **id** () [Retorna cadenas (String)]
  3. **setId** () [Procedimiento]
  4. **tokenlink** (s) [Retorna cadenas (String)]
  5. **setTokenlink** (b) [Procedimiento]
  6. **scope** (a) [Retorna cadenas (String)]
  7. **setScope** (z) [Procedimiento]
  8. **type** () [Retorna cadenas (String)]
  9. **setType** (a) [Procedimiento]
  10. **config** () [Retorna cadenas (String)]
  11. **setConfig** (q) [Procedimiento]
  12. **source** () [Retorna cadenas (String)]
  13. **setSource** (f) [Procedimiento]
  14. **documentsource** () [Retorna cadenas (String)]
  15. **setDocumentsource** (w) [Procedimiento]
  16. **description** () [Retorna cadenas (String)]
  17. **setDescription** (f) [Procedimiento]
  18. **foo** () [Retorna cadenas (String)]
  19. **getDocumentCount** () [Retorna un entero (int)]
  20. **getXMLDocument** (value,fieldno,documentid) [Retorna cadenas (String)]
  21. **createXMLFileFromQuery** (query,outputFileName) [Retorna cadenas (String)]
- QList<SafetDocument\*>& getDocuments();

## 5.5 Clase SafetYAWL

Clase **SafetYAWL** : SafetXObject

### 5.5.1 Listado de myinflow

1. **openXML** (a) [Procedimiento]
2. **convertXMLtoObjects** () [Procedimiento]
3. **openDataSources** () [Procedimiento]
4. **setCurrentPin** (p) [Procedimiento]
5. **currentPin** () [Retorna cadenas (String)]
6. **signDocumentsFromData** (c,nametowrite,list,doc,withsmartcard = true) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]
7. **verifyDocumentsFromData** (data,doc) [Retorna verdadero si se ejecuto la acción, en caso contrario retorna falso]
8. **loadAllConfFiles** (option = 3) [Procedimiento]
9. **loadPlugins** (pluginname) [Procedimiento]

## 5.6 Clase SafetWorkflow

### Clase SafetWorkflow: SafetXmlObject

#### 5.6.1 Listado de myinflow

```
enum OutputFormat { XML, JSON, SVG };
```

1. **SafetWorkflow** ()
2. **listTasks** (inc = false,c = “n”) [Retorna cadenas (String)]

```
# SafetDocument getDocuments(const QString& idvariable,
#                             OutputFormat of = XML const QString& info = "");
```

```
# SafetDocument getDocuments(const QString& idvariable,
#                            QList<QSqlField>& fields, int &howmanydocuments,
#                            OutputFormat of = XML, const QString& info = "");
```

## 6.- Tutorial de un ejemplo de inventario

---

### 6.1 Creación de la base de datos

**SQLITE** es un gestor de bases de datos muy ligero y potente. Por sus características se utiliza en una gran variedad de aplicaciones, como Skype, Mozilla Firefox, Adobe Photoshop Elements, el navegador web Opera, ...; y por supuesto en KEME-Contabilidad, donde es una alternativa para el almacenamiento de las contabilidades, junto con MySQL y PostgreSQL.

Las bases de datos bajo SQLITE se almacenan en un archivo que puede ser accedido por un programa monitor interactivo en modo texto denominado **sqlite3**. Mediante esta aplicación se pueden efectuar consultas y ediciones utilizando sentencias SQL.

Si estamos acostumbrados a utilizar **MySQL o Postgres**, sabemos que disponemos de aplicaciones para la administración de estas bases de datos bajo interfaz gráfica. Los máximos exponentes son las aplicaciones PHPmyADMIN y PgAdmin. Pues bien, para el caso de SQLITE estamos de enhorabuena porque disponemos de un plugin para el navegador Firefox que se denomina **SQLITE Manager** y que nos va a permitir la administración y consulta de nuestras bases de datos locales. [Más información en el siguiente enlace Administración gráfica de SQLITE con SQLite Manager](#)

En este tutorial crearemos la **base de datos** y sus 2 entidades (**productos** y **productos\_registro**) con el plugins (**sqlite-manager**), para trabajar con el modelo de (**inventario**). Permitiendo así el aprendizaje de **PySafet**.

Las dos tablas (**productos**) y (**productos\_registro**) significa los siguiente:

- La tabla (**productos**): Representa la lista de fichas (**token**) en los flujos de trabajo (**Workflow**).
- La tabla (**productos\_registro**): Representan la lista de eventos de cambio de estado (**status**) en los flujos de trabajo.

A continuación seguimos los siguiente pasos:

#### 6.1.1 A.- Instalación del plugins (**sqlite-manager**) en el navegador web

##### 1° PRIMER PASO

- Nos vamos al siguiente enlace [Sqlite-Manager](#).

### 2° SEGUNDO PASO

- Damos click al botón **+ Add to Firefox**, como se muestra en la *Figura 12: Agregar Sqlite-manager.*:

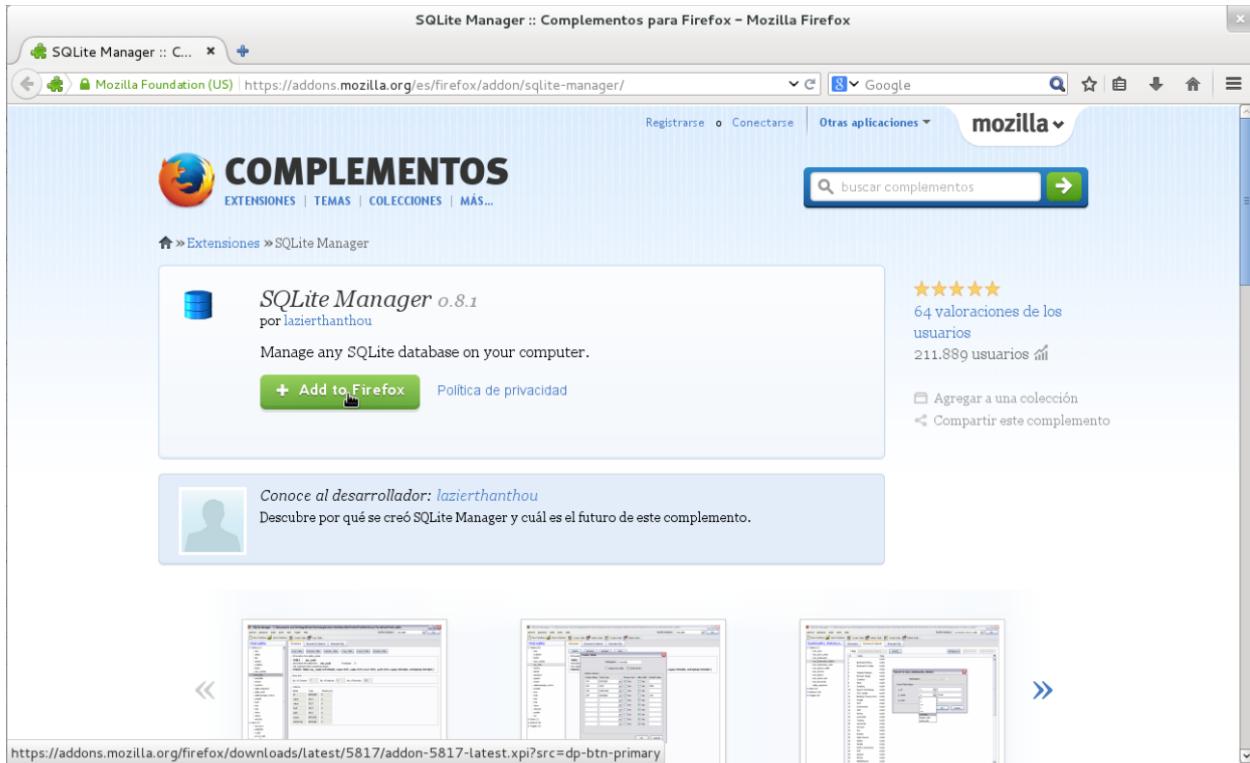


Figura 6.1: **Figura 12: Agregar Sqlite-manager.**

### 3° TERCER PASO

- Damos click en el botón **Permitir** para permitir la instalación en el navegador **Web**, como se muestra en la *Figura 13: Permitir la instalación.*:

### 4° CUARTO PASO

- Damos click en el botón **Instalar** para la instalación, como se muestra en la *Figura 14: Instalar el plugin.*:

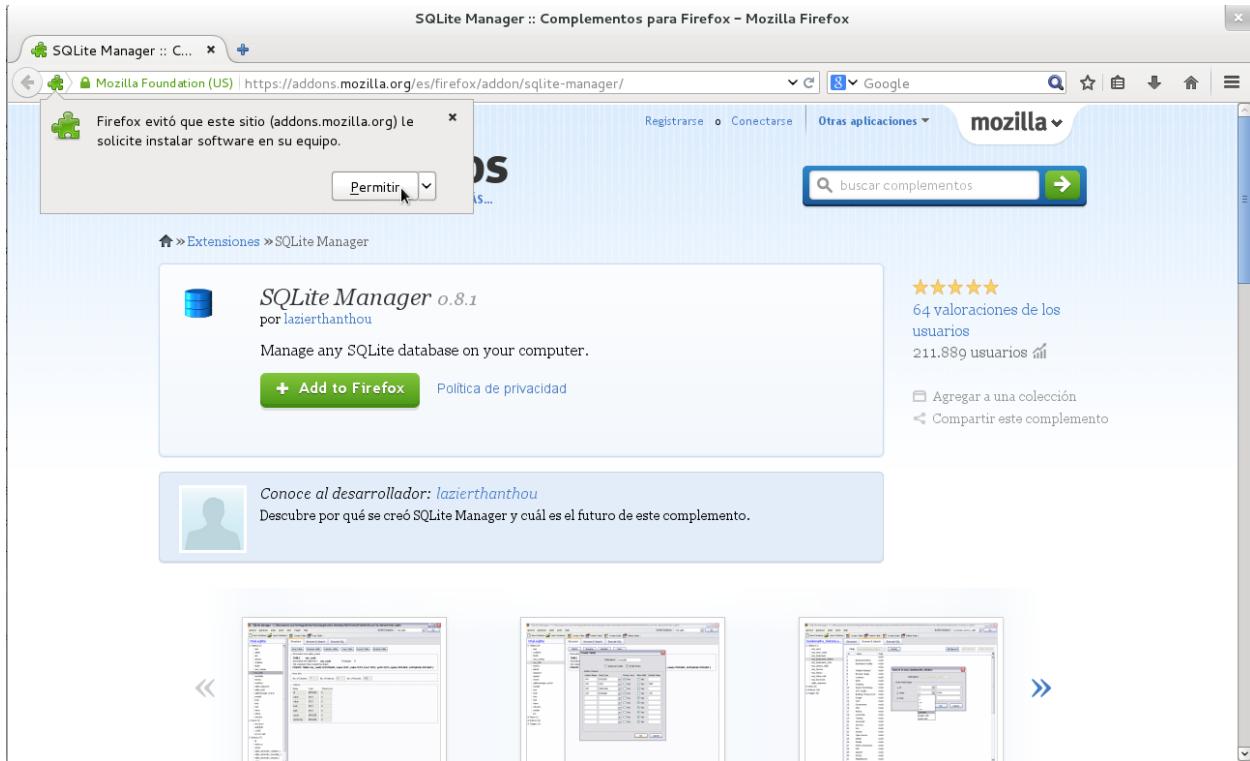


Figura 6.2: Figura 13: Permitir la instalación.

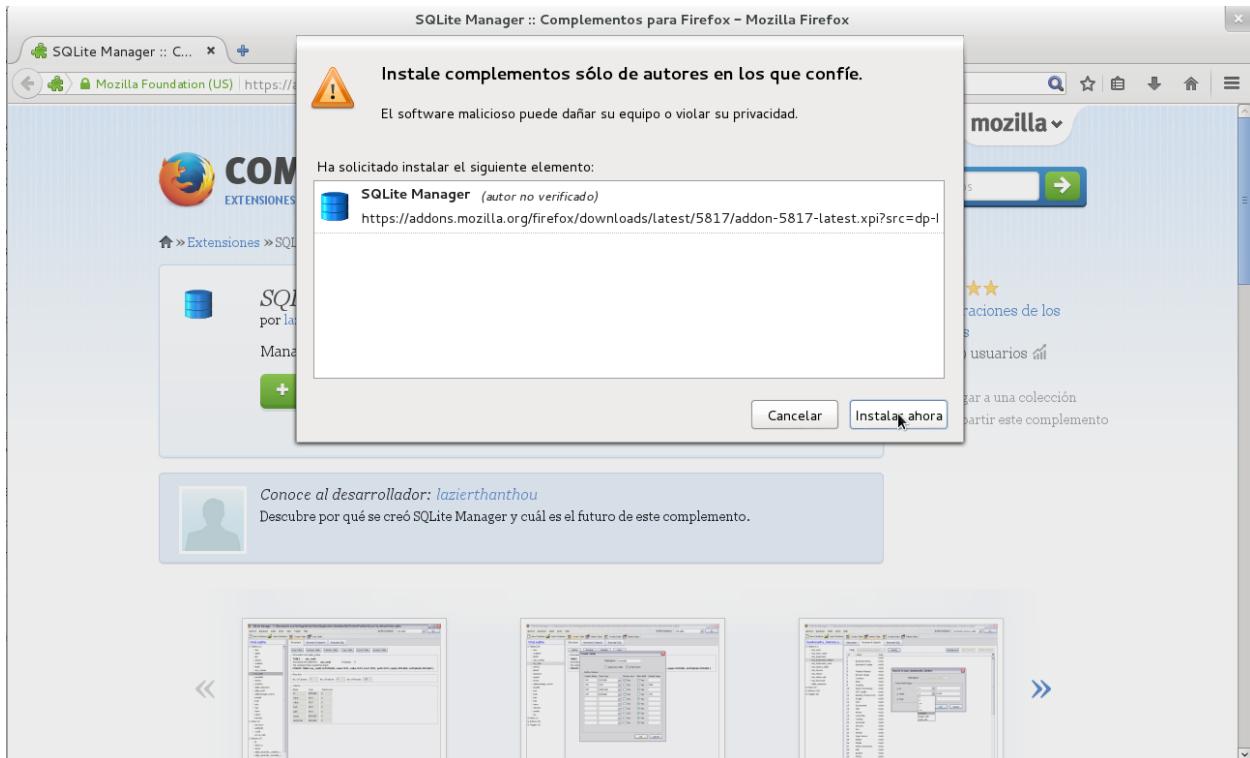


Figura 6.3: Figura 14: Instalar el plugins.

### 5° QUINTO PASO

- Damos click en el botón **Reiniciar** para reiniciar el navegador web, como se muestra en la *Figura 15*:  
*Reiniciar navegador:*

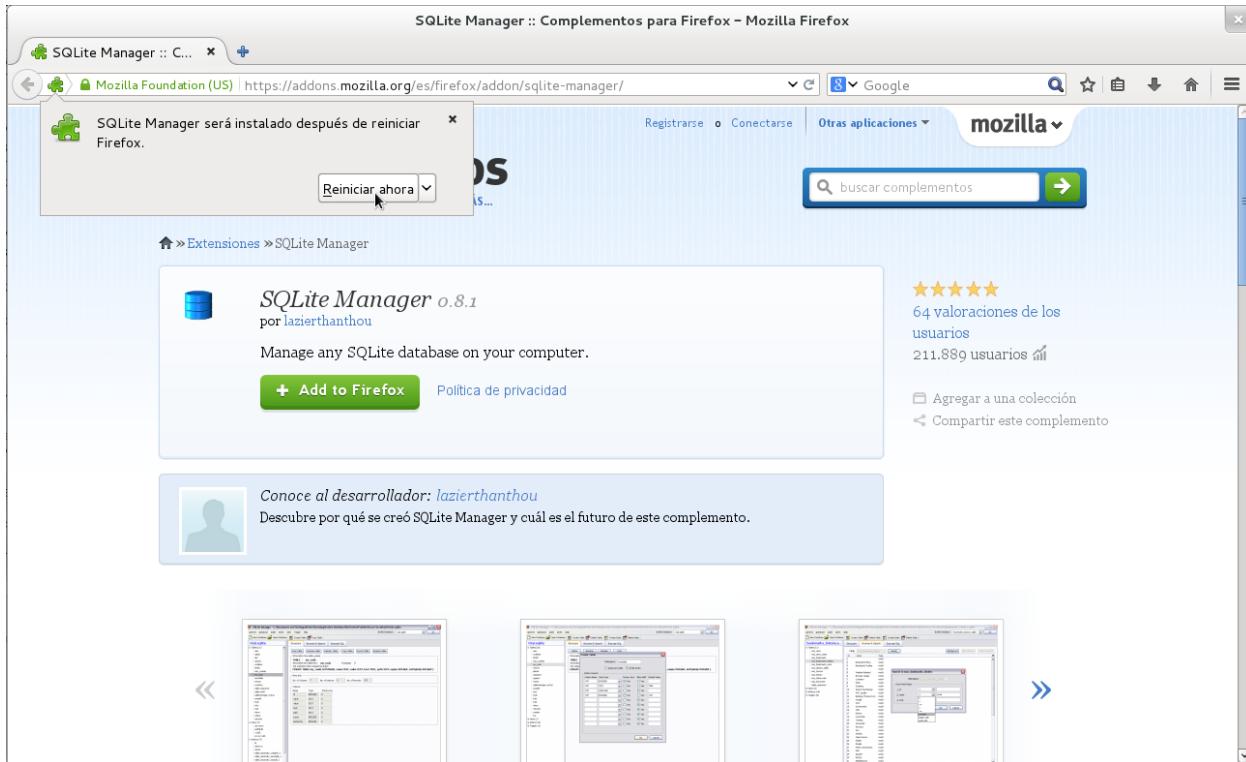


Figura 6.4: *Figura 15: Reiniciar navegador.*

### 6.1.2 B.- Creación de la base de datos con el plugins Sqlite-Manager

#### 1° PRIMER PASO

1. Abrimos el navegador web.
2. Buscamos en la barra de herramientas del navegador web, el plugins **Sqlite-Manager**.
3. Damos clik a la opción **Sqlite-Manager**.

Observen la siguiente *Figura 16: Plugins (Sqlite-Manager)*:

#### 2° SEGUNDO PASO

- Configuramos la extensión de la base de datos, para ello nos vamos a la opción de herramientas (**Opciones**), como se muestra en la *Figura 17: Configuración*.

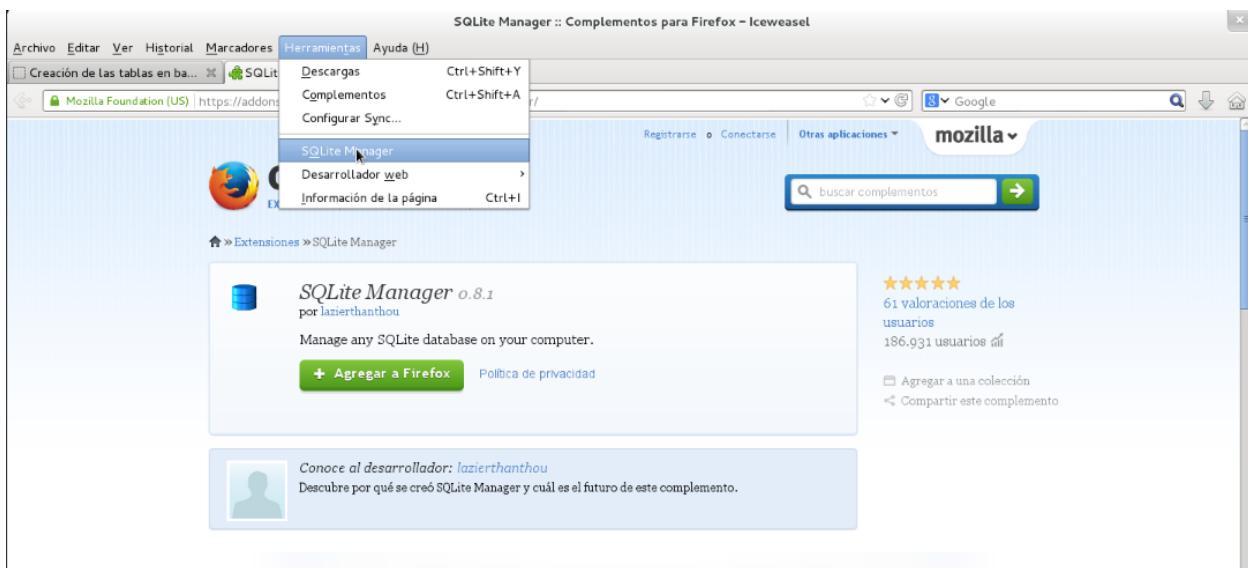


Figura 6.5: Figura 16: Plugins (Sqlite-Manager)

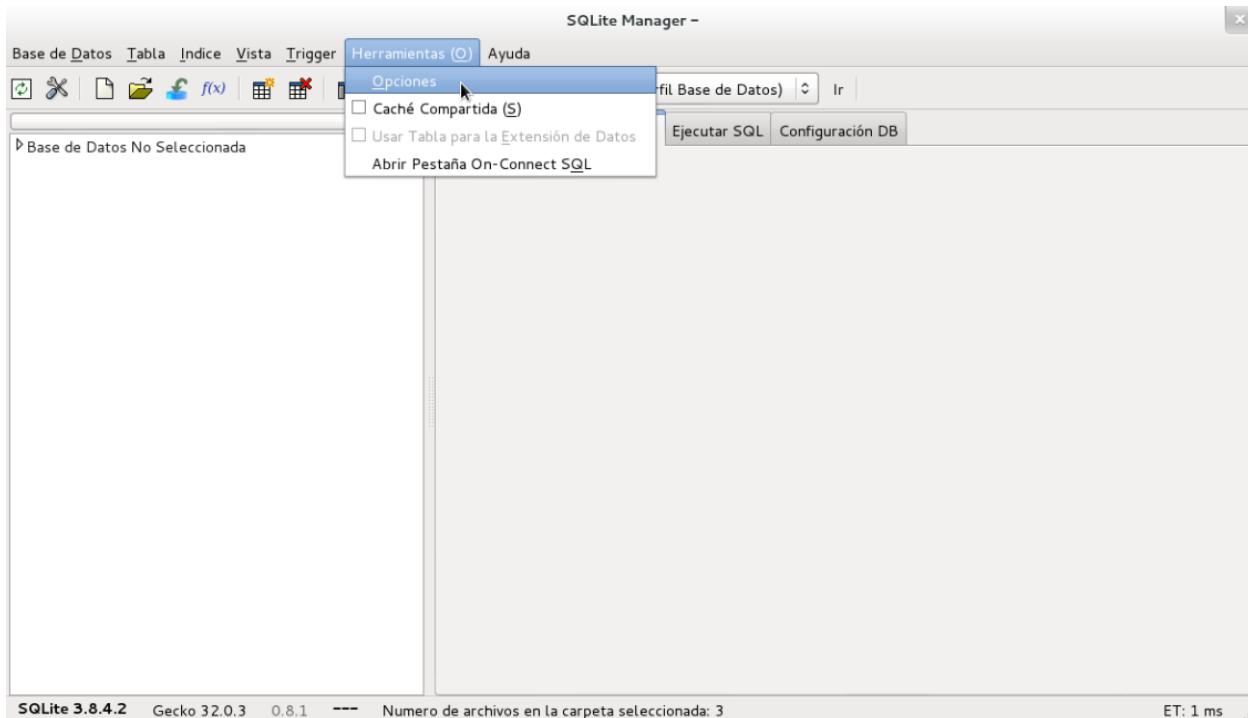


Figura 6.6: Figura 17: Configuración.

### 3° TERCER PASO

- Cambiamos la extensión por defecto (**sqlite**) por la extensión (**db**) y pulsamos en el botón (**Cerrar**), como se muestra en la *Figura 18: Extensión*:

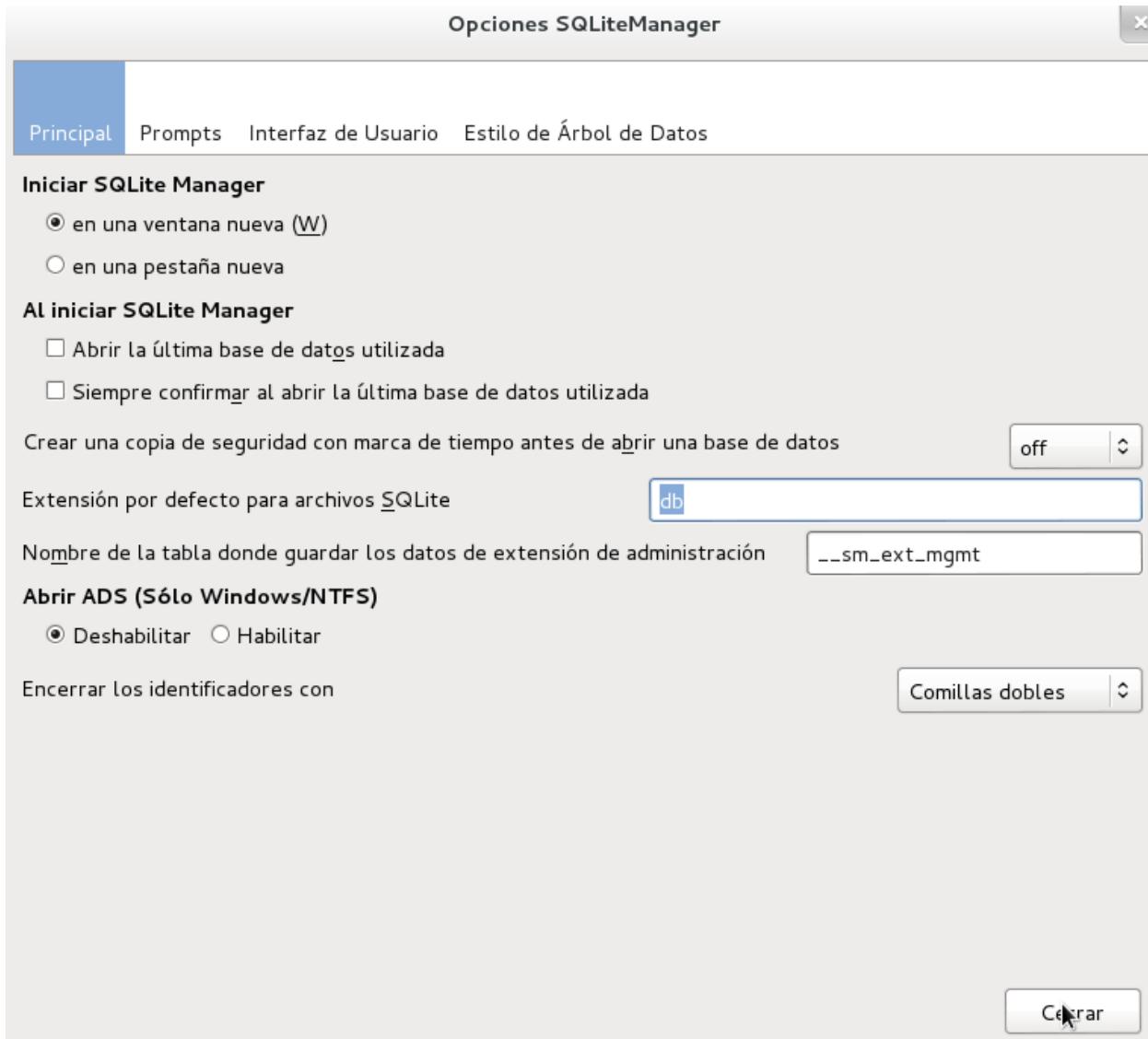


Figura 6.7: **Figura 18: Extensión.**

### 4° CUARTO PASO

- Damos click a la opción (**Nueva base de datos**), como se muestra en la siguiente *Figura 19: Opción para crear la base de datos*:

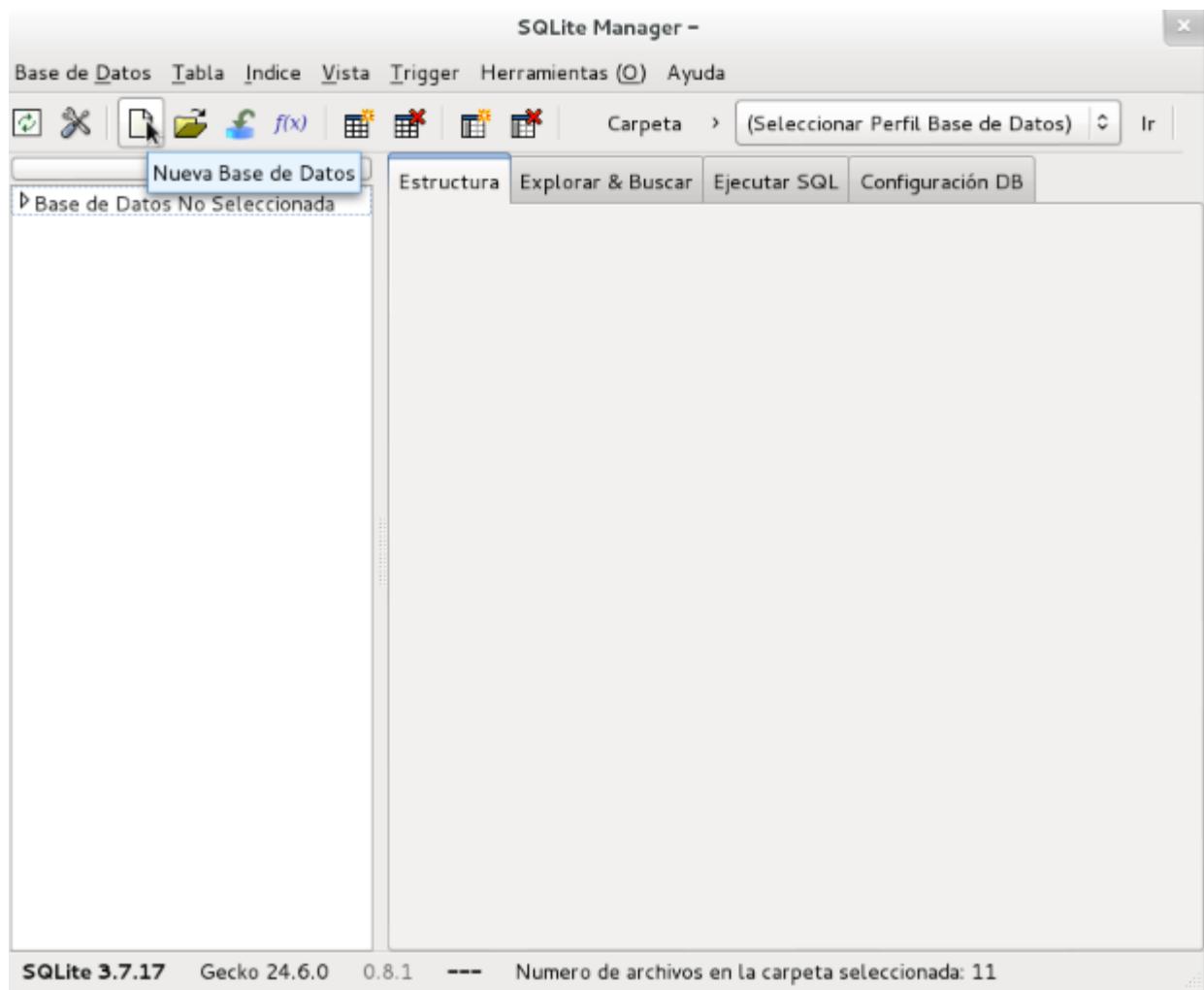


Figura 6.8: Figura 19: Opción para crear la base de datos

### 5° QUINTO PASO

- Agregamos el **nombre** de la base de datos, por ejemplo (**mydb**) y pulsamos en el botón (**Aceptar**), como se muestra en la siguiente *Figura 20: Nombre de la (base de datos)*:

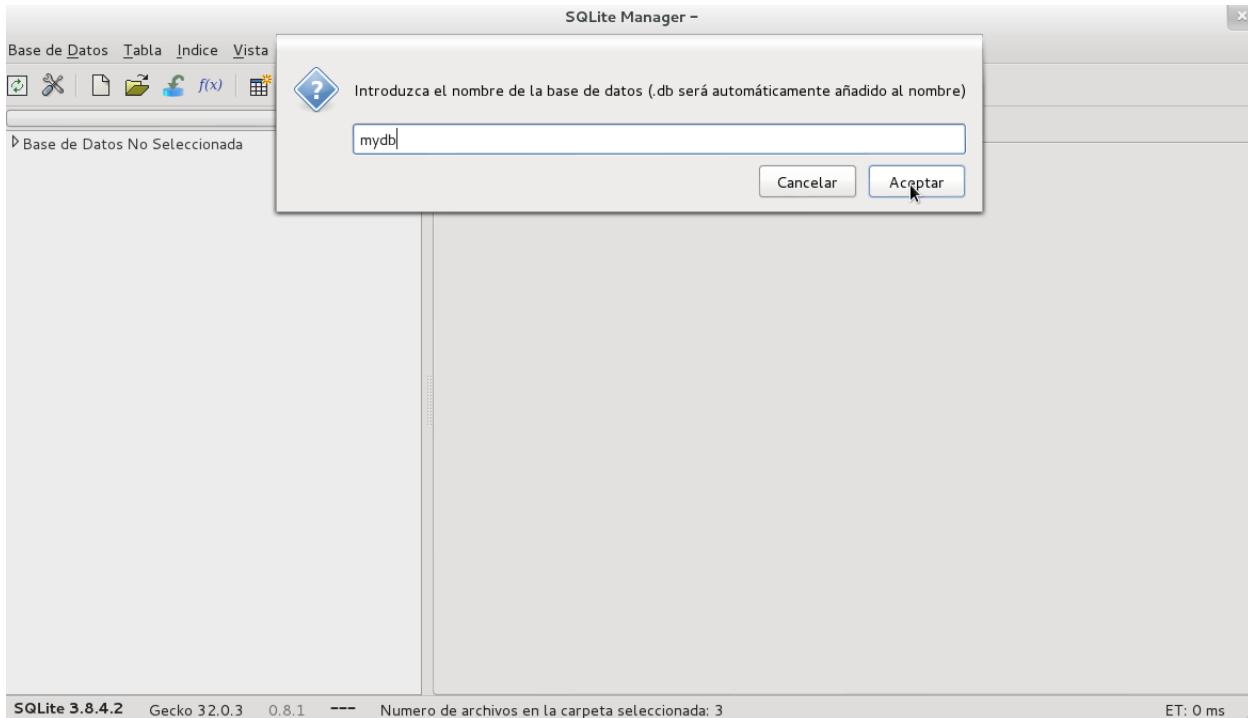


Figura 6.9: **Figura 20: Nombre de la (base de datos)**

### 6° SEXTO PASO

- Guardamos la base de datos en el directorio <HOME>.safet/, para ello buscamos el directorio y damos click al botón (**Abrir**), como aparece en la siguiente *Figura 21: Guardar la (Base de datos)*:

#### 6.1.3 C.- Creación de la tabla (productos)

La tabla (**productos**) contiene los atributos (**id, mensaje, status, cantidad, nombre, pedir**), para ello seguimos los siguientes pasos:

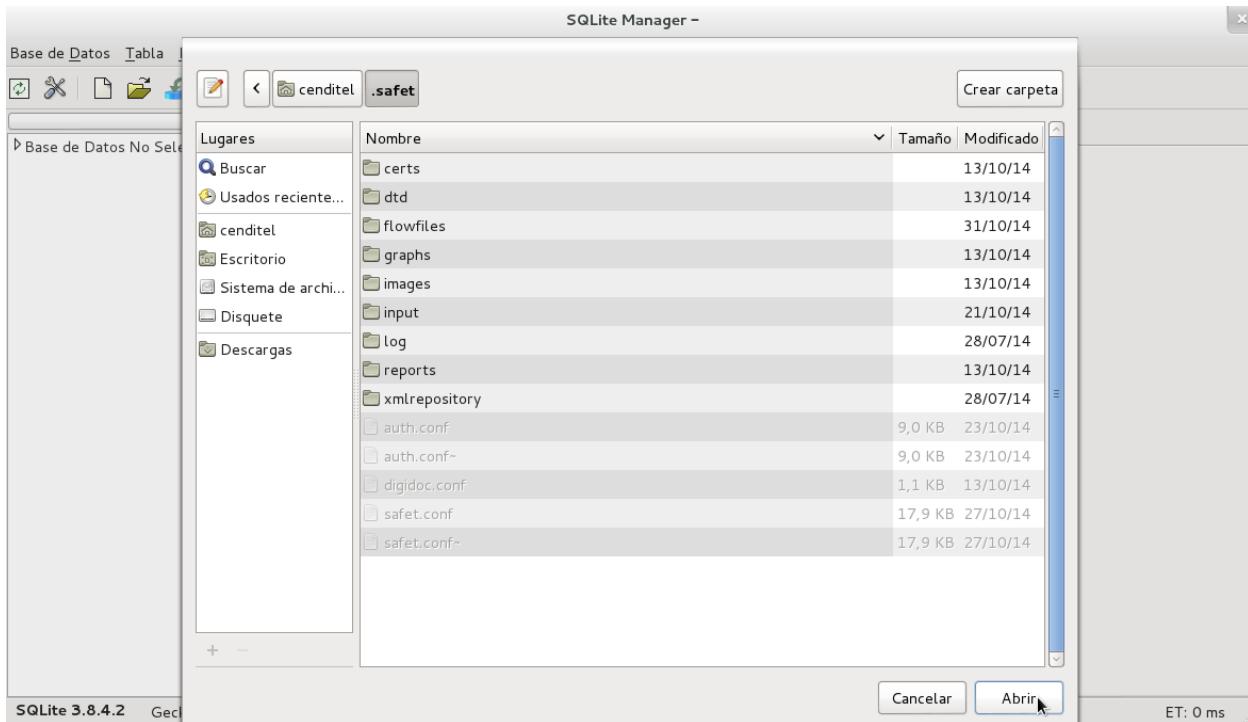


Figura 6.10: Figura 21: Guardar la (Base de datos)

## 1° PRIMER PASO

- Buscamos en la barra de herramientas la opción (**Tabla**) y damos click en (**Crear Tabla**), como se muestra en la siguiente *Figura 22: Opción (Crear tabla)*:

## 2° SEGUNDO PASO

- Llenaremos el formulario, colocando el nombre de la tabla (**productos**) y los atributos (**id, mensaje, status, cantidad, nombre, pedir**), como se muestra en la siguiente *Figura 23: tabla (productos)*:

**Nota:** El atributo (**id**) es una clave primaria y va a tener un contador.

## 3° TERCER PASO

- Se nos muestra un mensaje si deseamos realizar la operación, le decimos que (**si**), como se muestra en la siguiente *Figura 24: Mensaje de la tabla*:

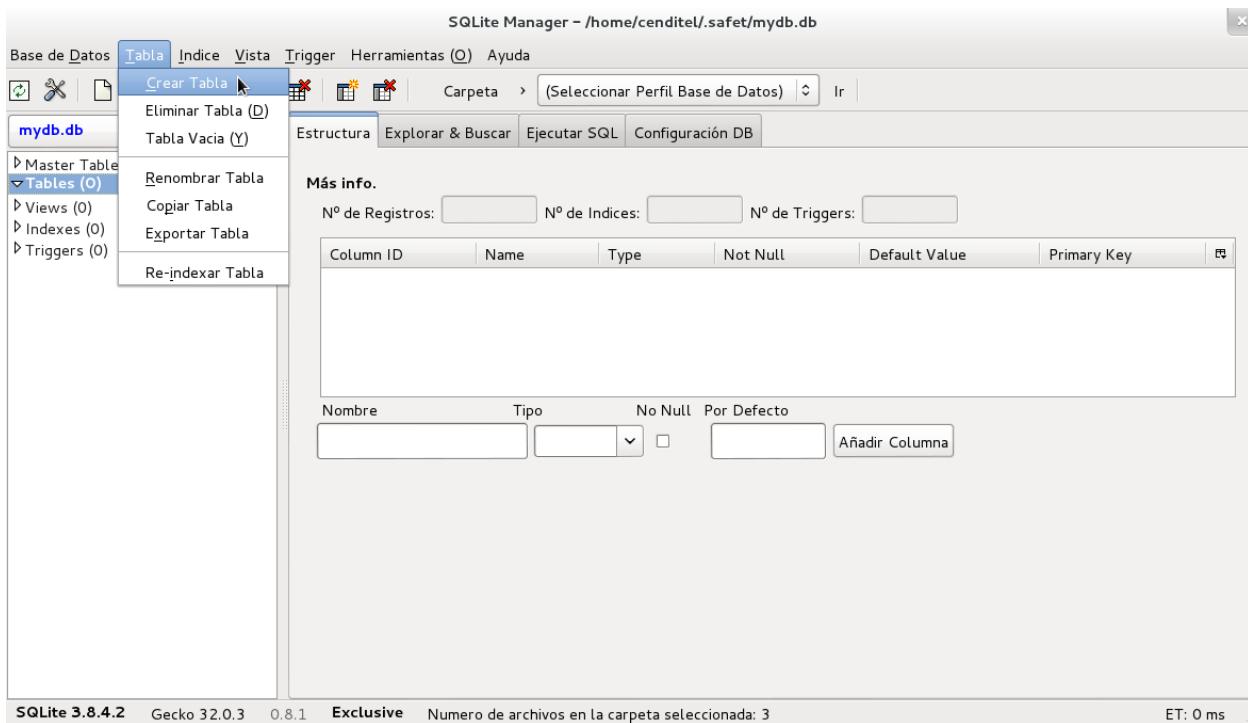


Figura 6.11: Figura 22: Opción (Crear tabla)

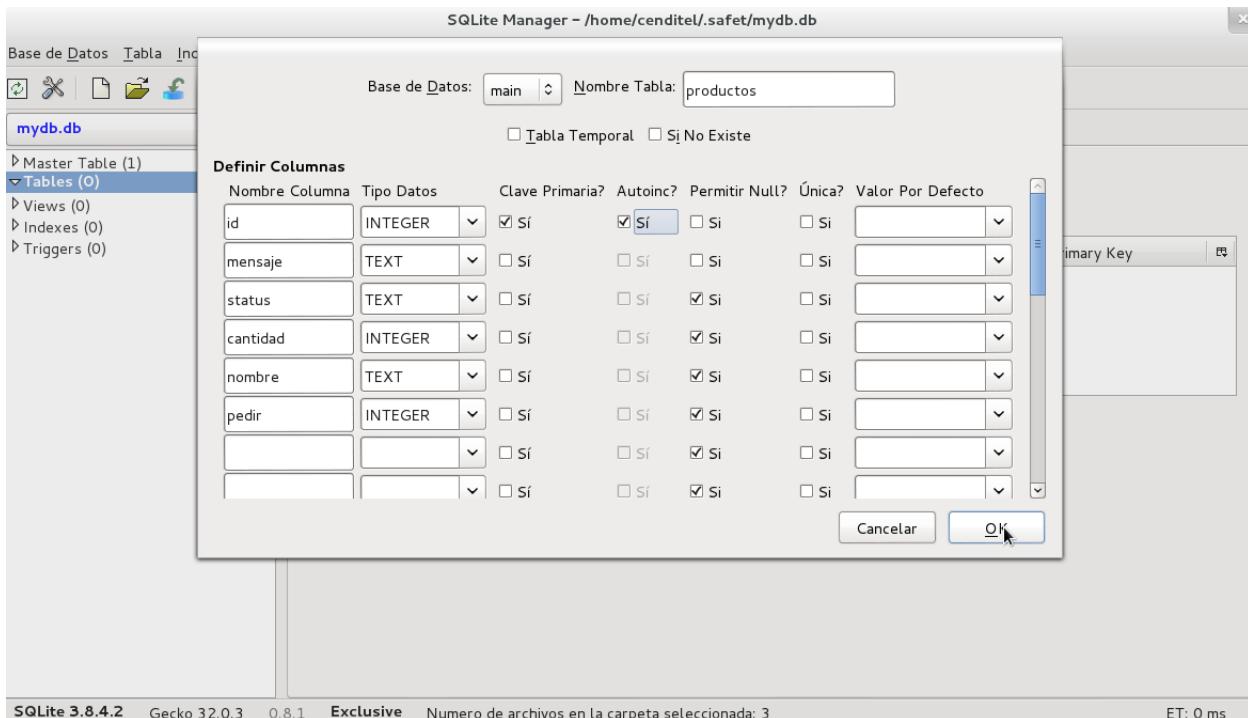


Figura 6.12: Figura 23: tabla (productos)

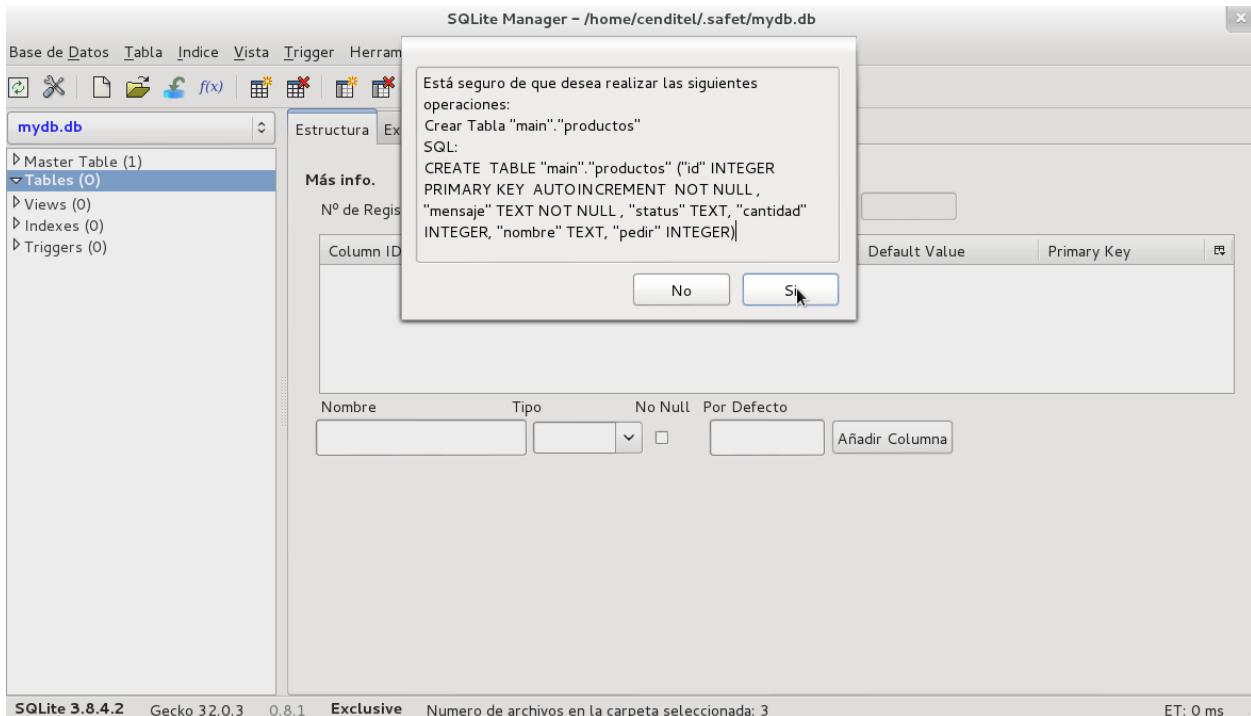


Figura 6.13: **Figura 24: Mensaje de la tabla**

## 4° CUARTO PASO

- Observamos la tabla (**productos**) con los campos creados en la base de datos, como se muestra en la siguiente *Figura 25: Tabla creada (productos)*:

### 6.1.4 D.- Creación de la tabla (**productos\_registro**)

La tabla (**productos\_registro**) contiene los atributos (**id, productoid, fecha, reqstatus**), para ello seguimos los siguientes pasos:

## 1° PRIMER PASO

- Buscamos en la barra de herramientas la opción (**Tabla**) y damos click en (**Crear Tabla**), como se muestra en la siguiente *Figura 26: Opción (Crear tabla)*:

## 2° SEGUNDO PASO

- Llenaremos el formulario, colocando el nombre de la tabla (**productos\_registro**) y los atributos (**id, productoid, fecha, reqstatus**), como se muestra en la siguiente *Figura 27: tabla productos\_registro*:

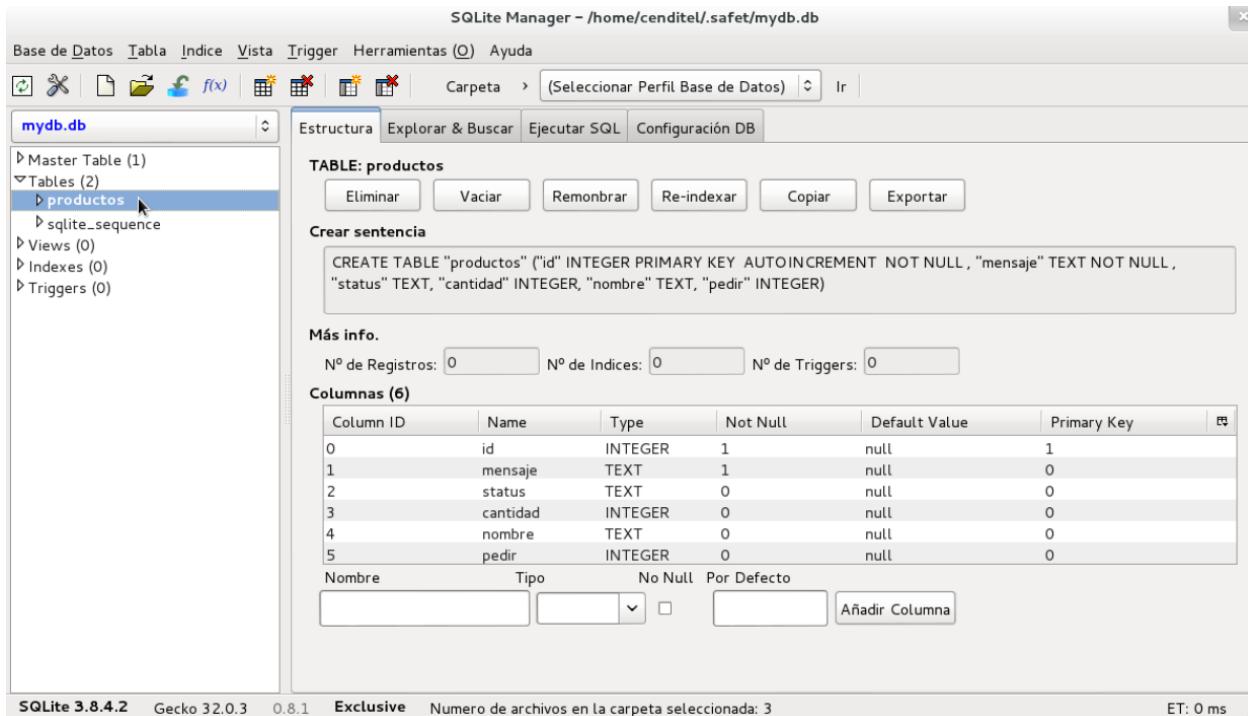


Figura 6.14: Figura 25: Tabla creada (productos)

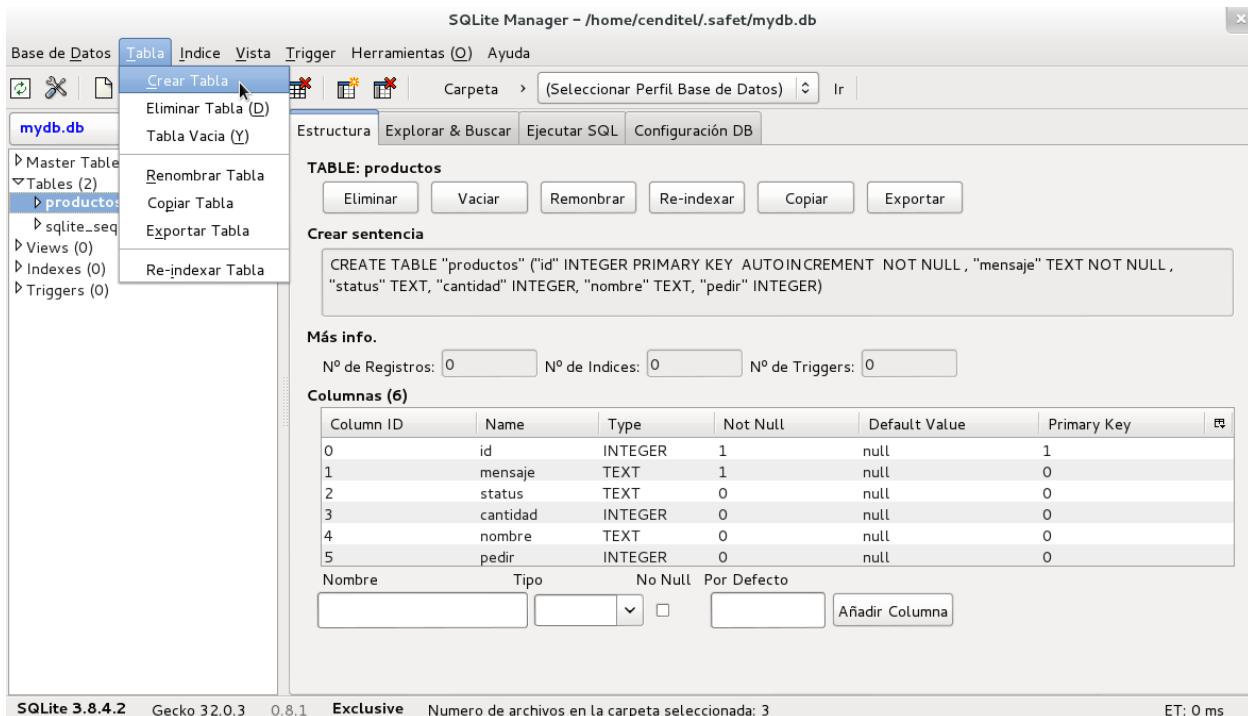


Figura 6.15: Figura 26: Opción (Crear tabla)

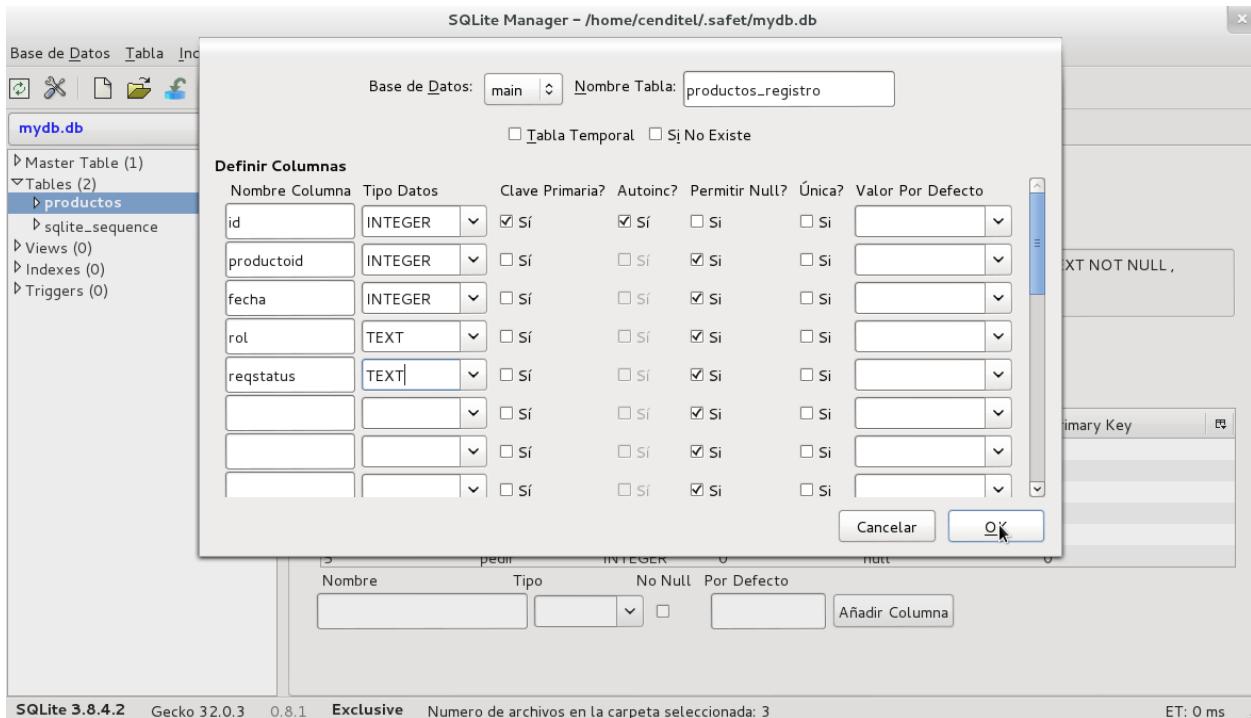


Figura 6.16: Figura 27: tabla productos\_registro

---

**Nota:** El atributo (**id**) es una clave primaria y va a tener un contador.

---

### 3º TERCER PASO

- Se nos muestra un mensaje si deseamos realizar la operación, le decimos que (**si**), como se muestra en la siguiente *Figura 28: Mensaje de la tabla*:

---

**Nota:** Si durante el tutorial les ocurrió algún error en la creación de base de datos o tablas, podemos descargar la base de datos (**mydb.db**) en el siguiente enlace. [Click aquí \(mydb.db\)](#):

---

## 6.2 Operaciones CRUD+(flujo) utilizando PySafet

Las operaciones **CRUD**, son acciones generales que representan un modelo básico para la creación de sistema de información. Puede ser una descripción más detallada en el siguiente enlace [SOBRE CRUD](#).

Los operaciones que vamos a utilizar son las siguientes:

1. C (Insertar)

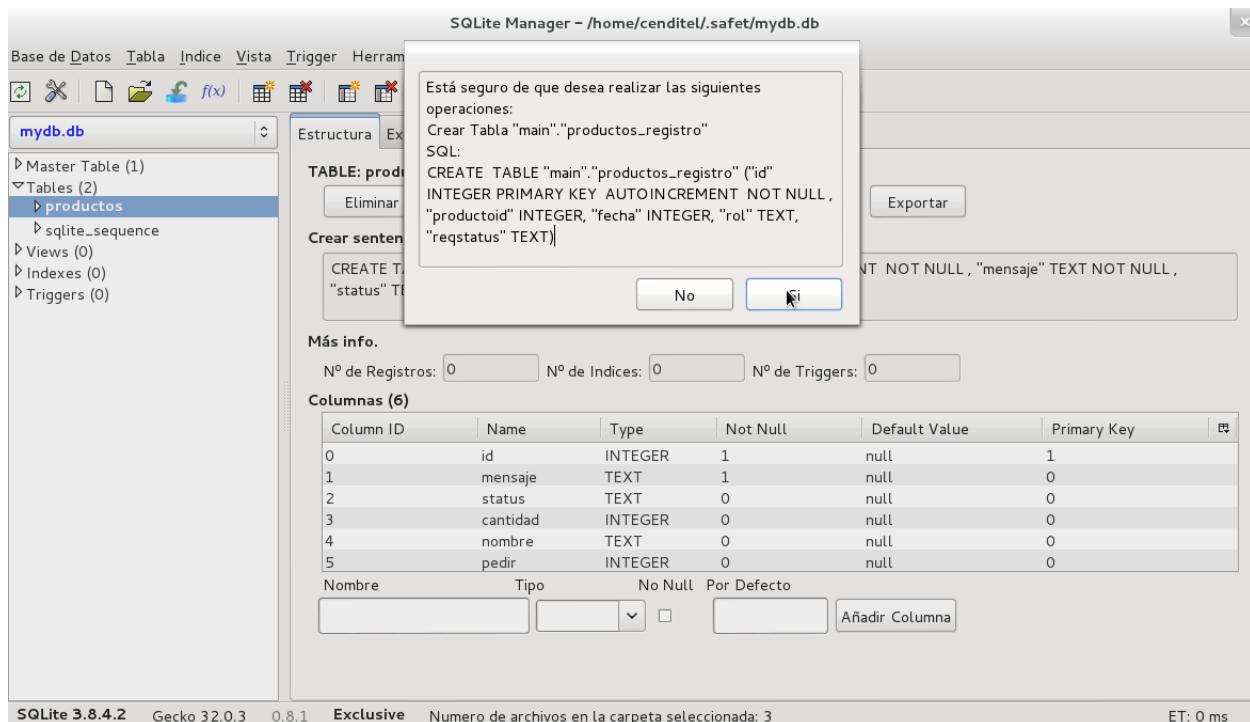


Figura 6.17: Figura 28: Mensaje de la tabla



Figura 6.18: Click aquí (mydb.db)

2. **R** (Listar\*)
3. **U** (Actualizar)
4. **D** (Borrar o Eliminar)
5. **+** (Flujo de trabajo)

El simbolo “+” indica las adición de operaciones asociada a flujo de trabajo. Las operaciones se alojan en una archivo llamado **deftrac.xml**, que se ubica en el directorio <HOME>.safe/input/, la cual se podara ejecutar estas operaciones mensionadas.

Este archivo debe crearse y contiene una estructura a seguir para ello comenzaremos con los siguientes pasos:

### 6.2.1 A.- Encabezado (Información de Autor).

Este archivo **deftrac.xml** contiene un encabezado como el siguiente:

#### 1° PRIMER PASO

- Creamos el archivo **deftrac.xml** en el directorio <HOME>.safe/input/.

```
.safet/input$ touch deftrac.xml
```

#### 2° SEGUNDO PASO

- Abrimos el archivo **deftrac.xml**, insertamos el siguiente encabezado:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--
Documento : deftrac.xml
Creado    : Fulano de tal
Autor     : Fulano de tal
Descripcion: Archivo de Entrada para SAFET - Inflow
-->
```

#### 3° TERCER PASO

- Debajo del encabezado insertamos la ruta del validador XML(formato DTD):

```
<!DOCTYPE operations SYSTEM
 "file:///home/cenditel/.safet/dtd/yawlinput.dtd">
```

### 6.2.2 B.- Operaciones CRUD+(flujo)

Las operaciones **CRUD** comienzan con la etiqueta principal **<operations>** donde se le asignan las operaciones principales **Agregar,Modificar,Eliminar,Listar**”.

- En el archivo **deftrac.xml**, insertamos el siguiente código:

```
<operations suffix=":" commandname="operacion">

<operation name="Productos"
           desc="Aregar, Modificar, Eliminar, Listar"
           icon="firmadoc.png">
</operation>
```

### 6.2.3 1° Primera operación CRUD+ “C(Insertar)”

Las operaciones en PySafet consiste en una lista de comandos que se ejecutan secuencialmente.

Los comandos se componen de campos “**Fields**” que corresponden a los diferentes tipos de datos básicos y complejos. Por ejemplo analicemos el código XML del archivo **deftrac.xml** (operación agregar producto). Se define los siguiente:

1. El tipo de comando es “**agregar**”.(type), los tipos posibles son “**agregar**”, “**actualizar**” y “**eliminar**”.
2. La tabla de la base de datos donde se realizará el comando “**command**” es “**productos**”.
3. Luego se especificar la lista de campos “**fields**”. Para este campo se especificaran dos campos, el nombre del producto “**Nombre**”, y el estado del producto que tomará el valor literal “**literal**”, “**Registrado**”.
4. El segundo comando de la operación “**Agregar\_producto**” también es del tipo “**Agregar**” y ojo los campos que son necesarios para registrar el evento “**Agregar\_producto**” en la tabla “**productos\_registro**”.
5. La palabra de PySafet “**\_USERNAME**” se utiliza para obtener el usuario actual.

A continuación se mostrara la operación para el archivo “**deftrac.xml**”:

### 1° PRIMER PASO

- Abrimos el archivo **deftrac.xml**, insertamos la primera operación:

```
<operation name="agregar_producto" desc="" icon="plus.png">
  <command id ="1" type="agregar" table="productos" >
    <fields>

      <field type="string" icon="resumen.png" mandatory="yes"
             validation="" title="Nombre" desc="">
        nombre
      </field>

      <field type="string" literal="Registrado" mandatory="yes" >
        status
      </field>

    </fields>
```

```

</command>

<command id ="1" type="agregar" table="productos_registro" >
<fields>

    <field type="datetime" mandatory="yes"
           function="seq from sqlite_sequence where
           name='productos'"   input="no">
productoid
</field>

    <field type="datetime" mandatory="yes"   function="datetime('now')"
           format="time_t" input="no">
fecha
</field>

    <field type="string" literal="_USERNAME" mandatory="yes" >
rol
</field>

    <field type="string" literal="Registrado" mandatory="yes" >
regstatus
</field>

</fields>
</command>
</operation>

```

## 2° SEGUNDO PASO

- Creamos el archivo .py por ejemplo **Script.py** en el directorio <HOME>.

```
$ touch Script.py
```

- Abrimos el archivo que creamos **Script.py**, insertamos el siguiente código:

```

# -*- coding: utf-8 -*-

# D(Borrar o eliminar)
# myconsult = u"operacion:borrar_producto id:5"

# U(Actualizar)
#myconsult = u"operacion:modificar_producto id:3 Nombre:Amoxicilina"

# +(Flujo de trabajo)
# myconsult = u"operacion:Actualizar_producto id:3
#                           Estado_producto: Pedido"

# Importación de la librería Safet y os
import Safet
import os

```

```
# Aqui obtengo mi home, media y url
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

# Constructor principal
myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

# Si no es un usuario registrado el metodo "login" retorna False
result = myinflow.login("admin","admin")

# Operación C(Insertar)
# Agregamos el producto a ingresar por ejemplo "Champu Olorin",
# "ComplejoB", "Aspirina", "Acetaminofén", "Ibuprofeno".

myconsult = u"operacion:agregar_producto Nombre: Champu Olorin"

if result:
    result = myinflow.toInputForm(myconsult)
else:
    print "\n ---Usuario autenticado---\n"
    exit()

if result:
    print "\n --Se Actualizo correctamente el producto---\n"
else:
    print "\n No se Actualizo el producto....!!!\n"

if not result:
    print "\nConsulta failed error: %s\n" % (myinflow.currentError())
    exit()

print u"  %s\n" % (myinflow.currentJSON())
```

---

**Nota:** Este Script contiene la operación:

- **Operación:** agregar\_producto
  - **Nombre:** El nombre del Producto agregar
- 

### 3° TERCER PASO

- Ejecutamos el archivo .py(**Script.py**), desde la consola de comando como usuario normal:

```
$ python $HOME/Script.py
```

**Nota:** Nos mostrara un mensaje como nos aparece en la siguiente *Figura 31: Insertar producto*

```
cenditel@debian:~$ python Script_Insertar_producto.py
QFSFileEngine::open: No file name specified
--Se inserto correctamente el producto---
{ "ticket": "n/t", "result": "false" }

cenditel@debian:~$ █
```

Figura 6.19: **Figura 31: Insertar producto**

---

#### 4° CUARTO PASO

Abrimos la base de datos (**mydb.db**) con el plugins **Sqlite-Manager** y verificamos si realizó la operación correctamente, como se muestra en la siguientes imagenes:

1.- **Observamos la tabla (productos):** *Figura 32: Tabla (productos)*

2.- **Observamos la tabla (productos\_registro):** *Figura 33: Tabla (productos\_registro)*

#### 6.2.4 2° Segunda operación CRUD+ “D(Borrar o Eliminar)”

Esta operación consiste en **Borrar** un producto de la base de datos para ello seguimos los siguientes pasos:

#### 1° PRIMER PASO

- Abrimos el archivo **desctrac.xml** del directorio <HOME>.safe/input/, insertamos la siguiente operación:

```
<operation name="borrar_producto" desc="Eliminar " icon="clear.png">
  <command id = "1" type="eliminar" table="productos">
    <fields>

      <field type="combolisttable"
        options="id:productos::id || ' - ' || nombre"
        mandatory="yes"
```

## Documentación de PySafet, Publicación

The screenshot shows the SQLite Manager interface with the database 'mydb.db' selected. The left sidebar lists tables: Master Table (1), Tables (3) including 'productos', Views (1), Indexes (0), and Triggers (0). The 'productos' table is currently selected. The main pane displays the 'productos' table structure with columns: id, mensaje, status, cantidad, and nombre. The data shows 5 rows of products: Champu Olorin, ComplejoB, Aspirina, Acetaminofén, and Ibuprofeno, all registered with status 'Registrado' and quantity 0.

TABLE productos				
	mensaje	status	cantidad	nombre
1		Registrado	0	Champu Olorin
2		Registrado	0	ComplejoB
3		Registrado	0	Aspirina
4		Registrado	0	Acetaminofén
5		Registrado	0	Ibuprofeno

Figura 6.20: Figura 32: Tabla (productos)

The screenshot shows the SQLite Manager interface with the database 'mydb.db' selected. The left sidebar lists tables: Master Table (1), Tables (3) including 'productos' and 'productos\_registro', Views (1), Indexes (0), and Triggers (0). The 'productos\_registro' table is currently selected. The main pane displays the 'productos\_registro' table structure with columns: id, productoid, fecha, rol, and regstatus. The data shows 5 rows of registration entries for products 1 through 5, all assigned to role 'admin' and status 'Registrado'. The dates are in a specific format (e.g., 1409160918).

TABLE productos_registro				
	productoid	fecha	rol	regstatus
1	1	1409160918	admin	Registrado
2	2	1409160948	admin	Registrado
3	3	1409160966	admin	Registrado
4	4	1409161128	admin	Registrado
5	5	1409161172	admin	Registrado

Figura 6.21: Figura 33: Tabla (productos\_registro)

```
        primarykey="yes"
        title="id"
        order="desc">

    id
</field>

    id
</field>

</fields>
</command>

<command id ="1" type="eliminar" table="productos_registro">
<fields>

    <field type="string" mandatory="yes" title="id" >
        productoid
    </field>

</fields>
</command>
</operation>
```

## 2° SEGUNDO PASO

- Abrimos el archivo **Script.py** y cambiamos la operación en la variable **myconsulta** como se muestra en el siguiente código:

```
# -*- coding: utf-8 -*-

# C(Insertar)
# Agregamos el producto a ingresar por ejemplo "Champu Olorin",
#           "ComplejoB", "Aspirina", "Acetaminofén", "Ibuprofeno".
#myconsult = u"operacion:agregar_producto Nombre: ComplejoB"

# U(Actualizar)
#myconsult = u"operacion:modificar_producto id:3 Nombre:Amoxicilina"

# +(Flujo de trabajo)
# myconsult = u"operacion:Actualizar_producto id:3
#                           Estado_producto: Pedido"

# Importación de la librería Safet y os
import Safet
import os

# Aquí obtengo mi home, media y url
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"
```

```
# Constructor principal
myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

# Si no es un usuario registrado el metodo "login" retorna False
result = myinflow.login("admin","admin")

# Operación D(Borrar o eliminar)
# Eliminamos el producto numero 5
myconsult = u"operacion:borrar_producto id:5"

if result:
    result = myinflow.toInputForm(myconsult)
else:
    print "\n ---Usuario autenticado---\n"
    exit()

if result:
    print "\n --Se borro correctamente el producto---\n"
else:
    print "\n No se borro el producto.....!!!\n"

if not result:
    print "\nConsulta failed error: %s\n" % (myinflow.currentError())
    exit()

print u"  %s\n" % (myinflow.currentJSON())
```

---

**Nota:** En este Script se utiliza la operación “**borrar\_producto**”:

- **Operación:** borrar\_producto
  - **id:** Aquí colocaremos el valor de **id** del producto por ejemplo borraremos el producto **Ibuprofeno** y su **id** es **5**. Observe la *Figura 32: Tabla (productos)*
- 

### 3º SEGUNDO PASO

- Ejecutamos el archivo **Script.py**, desde la consola de comando:

```
$ python $HOME/Script.py
```

---

**Nota:** Nos mostrara un mensaje como nos aparece en la siguiente *Figura 34: Eliminar producto*

---

```
cenditel@debian:~/cursoSafet$ python Eliminar_producto.py
QFSFileEngine::open: No file name specified
```

--Se borro correctamente el producto---

```
{ "ticket": "n/t", "result": "false" }
```

```
cenditel@debian:~/cursoSafet$ █
```

Figura 6.22: **Figura 34: Eliminar producto**

#### 4° CUARTO PASO

Abrimos la base de datos con el plugin **Sqlite-Manager** y verificamos si realizó la operación correctamente como se muestra en la siguientes imágenes:

**1.- Observamos la tabla (productos):** *Figura 35: Tabla (productos)*

TABLE productos				
	mensaje	status	cantidad	nombre
1		Registrado	0	Champu Olorin
2		Registrado	0	ComplejoB
3		Registrado	0	Aspirina
4		Registrado	0	Acetaminofén

Figura 6.23: **Figura 35: Tabla (productos)**

**2.- Observamos la tabla (productos\_registro):** *Figura 36: Tabla (productos\_registro)*

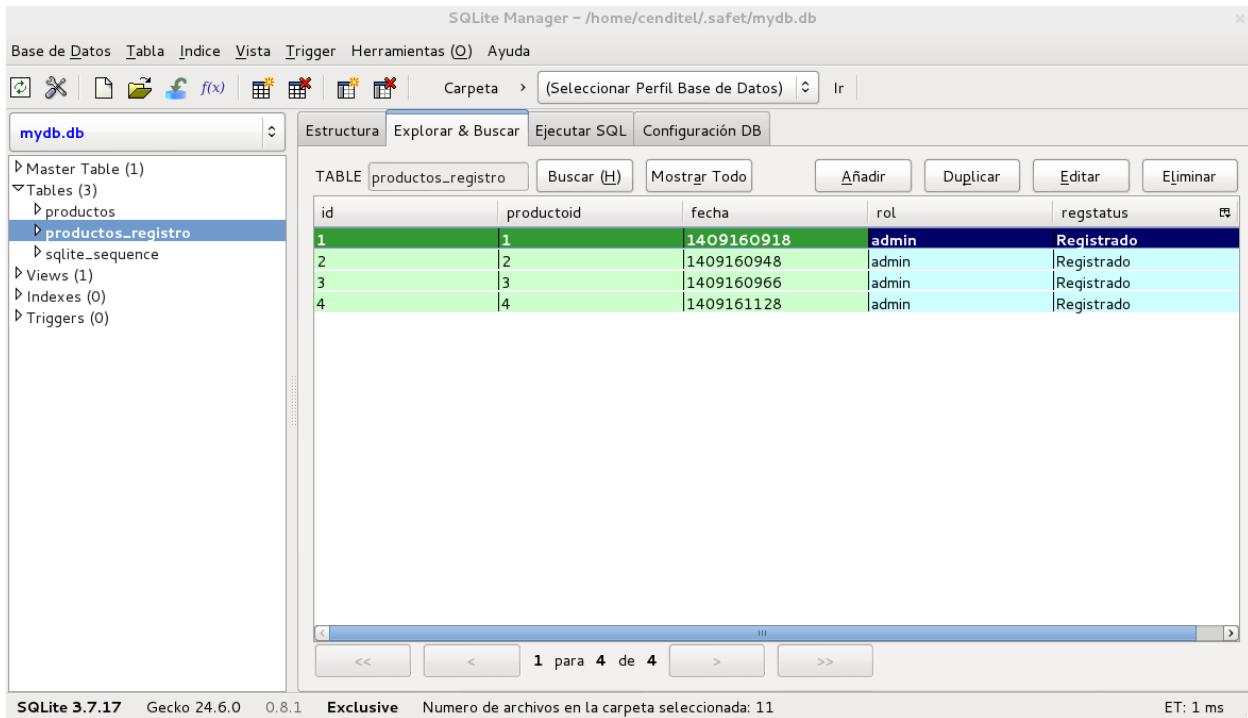


Figura 6.24: Figura 36: Tabla (productos\_registro)

### 6.2.5 3° Tercera operación CRUD+ “U(Actualizar)”

Esta operación **U(Actualizar)** consiste en actualizar o modificar el **nombre** de un producto de la base de datos, para ello seguimos los siguientes paso:

#### 1° PRIMER PASO

- Abrimos el archivo **desctrac.xml** del directorio <HOME>.safe/input/, insertamos la siguiente operación:

```

<operation name="modificar_producto" desc="" icon="plus.png">
  <command id ="1" type="actualizar" table="productos" >
    <fields>

      <field type="combolisttable"
        options="id:productos::id || ' - ' || nombre"
        mandatory="yes"
        primarykey="yes"
        title="id"
        order="desc">
        id
      </field>

      <field type="string" icon="resumen.png" mandatory="yes" validation=""
        title="Nombre" desc="">
        nombre
    
```

```

</field>

</fields>
</command>
</operation>
```

## 2° SEGUNDO PASO

- Abrimos el archivo **Script.py** y cambiamos la operación **myconsulta** como se muestra en el siguiente código:

```

# -*- coding: utf-8 -*-

# C(Insertar)
# Agregamos el producto a ingresar por ejemplo "Champu Olorin",
# "ComplejoB", "Aspirina", "Acetaminofén", "Ibuprofeno".
#myconsult = u"operacion:agregar_producto Nombre: ComplejoB"

# D(Borrar o eliminar)
# Eliminamos el producto numero 5
#myconsult = u"operacion:borrar_producto id:5"

# +(Flujo de trabajo)
# myconsult = u"operacion:Actualizar_producto id:3
#                                     Estado_producto: Pedido"

# Importación de la librería Safet y os
import Safet
import os

# Aquí obtengo mi home, media y url
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

# Constructor principal
myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

# Si no es un usuario registrado el metodo "login" retorna False
result = myinflow.login("admin", "admin")

# U(Actualizar)
# Se actualizara el nombre del producto número 3
myconsult = u"operacion:modificar_producto id:3 Nombre:Amoxicilina"
```

```
if result:
    result = myinflow.toInputForm(myconsult)
else:
    print "\n ---Usuario autenticado---\n"
    exit()

if result:
    print "\n --Se Actualizo correctamente el producto---\n"
else:
    print "\n No se Actualizo el producto....!!!\n"

if not result:
    print "\nConsulta failed error: %s\n" % (myinflow.currentError())
    exit()

print u"  %s\n" % (myinflow.currentJSON())
```

---

**Nota:** Seguidamente vamos a utilizar esta operación “Actualizar\_producto” en un Script de python como se muestra a continuación:

- **Operación:** Actualizar\_producto
  - **id:** Aqui colocaremos el valor del producto por ejemplo vamos a actualizar el producto **Aspirina** y su **id** es **3**. Observe la *Figura 32: Tabla (productos)*
  - **Nombre:** Aqui se coloca el nombre al que le vamos a modificar por ejemplo **Amoxacilina** por **Aspirina**.
- 

## 3° TERCER PASO

- Ejecutamos el archivo **Script.py**, desde la consola de comando:

```
$ python $HOME/Script.py
```

---

**Nota:** Nos mostrara un mensaje como nos aparece en la siguiente *Figura 37: Actualizar producto*

---

## 4° CUARTO PASO

Abrimos la base de datos (**mydb.db**) con el plugins **Sqlite-Manager** y verificamos si realizó la operación correctamente como se muestra en la siguiente: *Figura 38: Tabla (productos)*

```
cenditel@debian:~$ python Script.py
QFSFileEngine::open: No file name specified
```

--Se Actualizo correctamente el producto---

```
{ "ticket": "n/t", "result": "false" }
```

```
cenditel@debian:~$
```

Figura 6.25: Figura 37: Actualizar producto

The screenshot shows the SQLite Manager interface with the database 'mydb.db' selected. The 'productos' table is displayed in the main pane, showing the following data:

	id	mensaje	status	cantidad	nombre
4			Registrado	0	Acetaminofén
3			Pedido	0	Amoxicilina
1			Registrado	0	Champu Olorin
2			Registrado	0	ComplejoB

Figura 6.26: Figura 38: Tabla (productos)

## 6.2.6 4° Cuarta operación CRUD+ “+ (Flujo de trabajo (Estado))”

### 1° PRIMER PASO

- Abrimos el archivo **desctrac.xml** del directorio <HOME>.safe/input/, insertamos la siguiente operación:

```
<operation name="Actualizar_producto" desc="" icon="padlock.png">

<command id ="1" type="actualizar" table="productos">

<fields>

<field type="combolisttable"
    options="id:productos::id || ' - ' || nombre"
    mandatory="yes"
    primarykey="yes"
    title="id"
    order="desc">

    id
</field>

<field type="comboflow" mandatory="yes" options="next"
    path="/home/cenditel/.safet/flowfiles/productos.xml"
    title="Status_producto">
    status
</field>

</fields>
</command>

<command id ="1" type="agregar" table="productos_registro" >
<fields>

<field type="datetime" mandatory="yes"
    function="seq from sqlite_sequence where name='productos' "
    input="no">
productoid
</field>

<field type="datetime" mandatory="yes"  function="datetime('now')"
    format="time_t" input="no">
fecha
</field>

<field type="string" literal="_USERNAME" mandatory="yes" >
rol
</field>

<field type="string" title="Status" mandatory="yes" >
regstatus
</field>
```

```
</fields>
</command>
</operation>
```

## 2° SEGUNDO PASO

- Abrimos el archivo **Script.py** y cambiamos la operación **myconsulta** como se muestra en el siguiente código:

```
# -*- coding: utf-8 -*-

# C(Insertar)
# Agregamos el producto a ingresar por ejemplo "Champu Olorin",
#           "ComplejoB", "Aspirina", "Acetaminofén", "Ibuprofeno".
#myconsult = u"operacion:agregar_producto Nombre: ComplejoB"

# D(Borrar o eliminar)
# Eliminamos el producto numero 5
#myconsult = u"operacion:borrar_producto id:5"

# U(Actualizar)
# Se actualizara el nombre del producto número 3
#myconsult = u"operacion:modificar_producto id:3 Nombre:Amoxicilina"

# Importación de la librería Safet y os
import Safet
import os

# Aquí obtengo mi home, media y url
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

# Constructor principal
myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

# Si no es un usuario registrado el metodo "login" retorna False
result = myinflow.login("admin", "admin")

# +(Flujo de trabajo)
# Se actualizara su estado
myconsult = u"operacion:Actualizar_producto id:3 Estado_producto:Pedido"

if result:
```

```
        result = myinflow.toInputForm(myconsult)
else:
    print "\n ---Usuario autenticado---\n"
    exit()

if result:
    print "\n --Se actualizo el producto correctamente el producto---\n"
else:
    print "\n No se actualizo el producto.....!!!\n"

if not result:
    print "\nConsulta failed error: %s\n" % (myinflow.currentError())
    exit()

print u"  %s\n" % (myinflow.currentJSON())
```

---

**Nota:** Seguidamente vamos a utilizar esta operación “Actualizar\_producto” en un Script de python como se muestra a continuación:

- **Operación:** Actualizar\_producto
  - **id:** Aqui colocaremos el valor de **id** del producto por ejemplo vamos a actualizar el producto **Aspirina** y su **id** es **3**. Observe la *Figura 32: Tabla (productos)*
  - **Estado\_producto:** Aqui se coloca el estado del producto es decir si es por llegar,Agotarse,Pedido,En Espera. En este caso esta **Registrado** vamos a modificarlo a **pedido**.
- 

### 3° TERCER PASO

- Ejecutamos el archivo **Script.py**, desde la consola de comando:

```
$ python $HOME/Script.py
```

---

**Nota:** Nos mostrara un mensaje como nos aparece en la siguiente *Figura 39: Actualizar estado*

```
cenditel@debian:~/cursoSafet$ python Modificar_producto.py
QFSFileEngine::open: No file name specified

--Se Actualizo correctamente el producto---

{ "ticket": "n/t", "result": "false" }

cenditel@debian:~/cursoSafet$ □
```

Figura 6.27: **Figura 39: Actualizar estado**

## 4° CUARTO PASO

Abrimos la base de datos (**mydb.db**) con el plugin **Sqlite-Manager** y verificamos si realizó la operación correctamente como se muestra en la siguientes imágenes:

### 1.- Observamos la tabla (productos): *Figura 40: Tabla (productos)*

TABLE productos					
	id	mensaje	status	cantidad	nombre
1			Registrado	0	Champu Olorin
2			Registrado	0	ComplejoB
3			Pedido	0	Aspirina
4			Registrado	0	Acetaminofén

Figura 6.28: **Figura 40: Tabla (productos)**

### 2.- Observamos la tabla (productos\_registro): *Figura 41: Tabla (productos\_registro)*

#### 6.2.7 5° Quinta operación CRUD+ “(Siguiente estado)”

## 1° PRIMER PASO

- Abrimos el archivo **desctrac.xml** del directorio <HOME>.safe/input/, insertamos la siguiente operación:

```
<operation name="Siguiente_estado_producto" desc="" icon="padlock.png">
<command id = "1" type="actualizar" table="productos">
  <fields>
    <field type="combolisttable"
      options="id:productos::'('|| id || ')' || nombre"
      mandatory="yes"
      primarykey="yes"
```

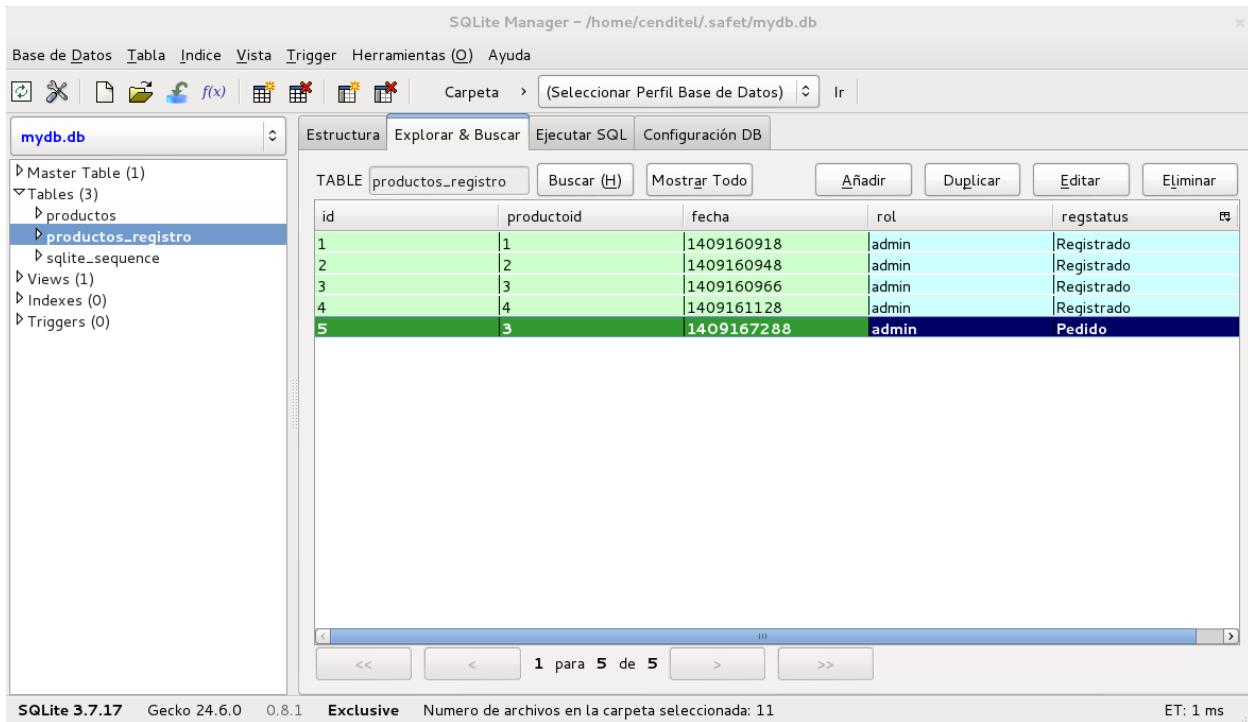


Figura 6.29: Figura 41: Tabla (productos\_registro)

```

changekey="yes"
title="id"
order="desc" >

    id
</field>

<field type="comboflow" mandatory="yes" options="next"
path="/home/debian/.safet/flowfiles/productos_siguiente_Estado.xml"
title="Siguiente_status" changefor="id">

    status
</field>

</fields>
</command>
<command id ="1" type="agregar" table="productos_registro" >
<fields>
    <field type="datetime" mandatory="yes"
function="seq from sqlite_sequence where name='productos'" input="no">

        productoid
</field>
    <field type="datetime" mandatory="yes"    function="datetime('now')"
format="time_t" input="no">

        fecha

```

```

</field>

<field type="string" literal="_USERNAME" mandatory="yes" >
    rol
</field>

<field type="string" title="Siguiente_status" mandatory="yes" >
    regstatus
</field>

</fields>
</command>

</operation>

```

## 2º SEGUNDO PASO

- Abrimos el archivo **Script.py** y cambiamos la operación **myconsulta** como se muestra en el siguiente código:

```

# -*- coding: utf-8 -*-

# C(Insertar)
# Agregamos el producto a ingresar por ejemplo "Champu Olorin",
#           "ComplejoB", "Aspirina", "Acetaminofén", "Ibuprofeno".
#myconsult = u"operacion:agregar_producto Nombre: ComplejoB"

# D(Borrar o eliminar)
# Eliminamos el producto numero 5
#myconsult = u"operacion:borrar_producto id:5"

# U(Actualizar)
# Se actualizara el nombre del producto número 3
#myconsult = u"operacion:modificar_producto id:3 Nombre:Amoxicilina"

# Importación de la librería Safet y os
import Safet
import os

# Aqui obtengo mi home,media y url
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

# Constructor principal
myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

```

```
# Si no es un usuario registrado el metodo "login" retorna False
result = myinflow.login("admin","admin")

# +(Flujo de trabajo)
# Se actualizara su estado
myconsult = u"operacion:Siguiente_Estado id:3 \
                                         Siguiente_status: Disponible"

if result:
    result = myinflow.toInputForm(myconsult)
else:
    print "\n ---Usuario autenticado---\n"
    exit()

if result:
    print "\n --Se cambio el estado producto---\n"
else:
    print "\n No se cambio de estado...!!!\n"

if not result:
    print "\nConsulta failed error: %s\n" % (myinflow.currentError())
    exit()

print u"  %s\n" % (myinflow.currentJSON())
```

---

**Nota:** Seguidamente vamos a utilizar esta operación “Actualizar\_producto” en un Script de python como se muestra a continuación:

- **Operación:** Siguiente\_estado
  - **id:** Aqui colocaremos el valor de **id** del producto por ejemplo vamos a actualizar el producto **Aspirina** y su **id** es **3**. Observe la *Figura 32: Tabla (productos)*
  - **Siguiente\_status:** Aqui se coloca el estado del producto es decir si es por llegar,Agotarse,Pedido,En Espera. En este caso esta **Registrado** vamos a modificarlo a **pedido**.
- 

### 3° TERCER PASO

- Ejecutamos el archivo **Script.py**, desde la consola de comando:

```
$ python $HOME/Script.py
```

### 4° CUARTO PASO

Abrimos la base de datos (**mydb.db**) con el plugins **Sqlite-Manager** y verificamos si realizó la operación correctamente como se muestra en la siguientes imagenes:

### 1.- Observamos la tabla (productos): *Figura 42: Tabla (productos)*

The screenshot shows the SQLite Manager interface with the database 'mydb.db' open. On the left, the tree view shows 'Tables (3)' with 'productos' selected. The main area displays the 'productos' table structure and data:

id	mensaje	status	cantidad	nombre
1		Registrado	0	Champu Olorin
2		Registrado	0	ComplejoB
3		Pedido	0	Aspirina
4		Registrado	0	Acetaminofén

At the bottom, the status bar indicates: SQLite 3.7.17 Gecko 24.6.0 0.8.1 Exclusive Número de archivos en la carpeta seleccionada: 11 ET: 0 ms.

Figura 6.30: **Figura 42: Tabla (productos)**

---

**Nota:** Si durante el tutorial les ocurrió algún error en la ejecución, podemos descargar el archivo (**desctrac.xml**) y el **Script.py** de python, para ello damos click al siguiente enlace:

## 6.3 Crear el flujo de trabajo utilizando un archivo XML

El flujo de trabajo (workflow) es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas. Más información [AQUÍ](#).

A continuación seguimos los siguientes paso:

### 6.3.1 A.- Creación del archivo (XML) de nuestro flujo de trabajo

#### 1° PRIMER PASO

- Desde la consola de comando como usuario (**normal**), accedemos al <HOME>.safet/flowfiles/ con el comando (**cd**) como se muestra en el siguiente código:



Figura 6.31: Click aquí (deftrac.xml) —Y— Click aquí (Script.py)

```
$ cd $HOME/.safet/flowfiles/
```

## 2° SEGUNDO PASO

- Con el comando (**touch**) creamos el archivo (XML) por ejemplo (**productos.xml**), como se muestra en siguiente código:

```
.safet/flowfiles $ touch productos.xml
```

### 6.3.2 B.- Encabezado del archivo XML (**productos.xml**)

#### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** que creamos en el paso anterior, insertamos el siguiente encabezado (**XML**), como se muestra en el siguiente código:

```
<?xml version='1.0' encoding='UTF-8'?>  
  
<!DOCTYPE yawl SYSTEM  
      'file:///home/cenditel/.safet/dtd/yawlworkflow.dtd'>  
    <!--  
Documento : productos.xml  
Creado   : 16/10/08 09:27 AM  
Autor     : nombre_autor  
Descripcion: Archivo generado por plantilla de la Libreria SAFET  
-->
```

---

**Nota:** El comentario es opcional.

---

### 6.3.3 C.- Etiquetas principales del flujo de trabajo en el archivo xml (productos.xml)

En el archivo **productos.xml** se utilizaran las siguientes **etiquetas principales (XML)**:

- **<yawl>**: Etiqueta principal yawl.
- **<workflow>**: Nombre del flujo.
- **<token>**: Ficha: Nombre de la tabla y Nombre del campo.

#### 1° PRIMER PASO

- Con un editor de texto abrimos el archivo **productos.xml** creado en el paso anterior.

#### 2° SEGUNDO PASO

- Debajo del *B.- Encabezado del archivo XML (productos.xml)*, insertamos la etiquetas principales (XML), como se muestra en el siguiente código:

```

<yawl version="0.01">
  <workflow id="productos">
    <token keysource="productos" key="id"/>

    <!-- **CUERPO** 
      Aqui vamos a colocar todos los estados o tareas.
      ##### 
      ## - INICIAL      ##
      ## - REGISTRADO   ##
      ## - PEDIDO        ##
      ## - POR_LLEGAR    ##
      ## - DISPONIBLE    ##
      ## - POR_AGOTARSE  ##
      ## - AGOSTADO      ##
      ## - FINAL         ##
      ##### 
    -->

  </workflow>
</yawl>

```

---

**Nota: Cuerpo de archivo (xml):** EL cuadro comentado es opcional, es solo para visualizar como van hacer las tareas o procesos en su desarrollo.

---

### 6.3.4 D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo

En las **condiciones principal (inicial y final)** se utilizaran las siguientes **etiquetas (XML)**:

- **<condition>inicial**: Condición inicial
- **<condition>final**: Condición final

- **<port> inicial:** Dentro de esta etiqueta van las conexiones continene una conexión.
- **<port> final:** No tiene a quien conectarse.
- **<connection> inicial:** Como no hay nada registrado inicio apunta a final
- **<connection> final:** final no apunta a nada

Ahora ejecutaremos los siguiente pasos:

### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y dentro de las *C.- Etiquetas principales del flujo de trabajo en el archivo xml (productos.xml)*, insertamos el siguiente código (**XML**):

```
<!--
#####
# Condición inicial #
#####
-->
<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="true" options="" source="final"/>
  </port>
</condition>

<!--
#####
# Condición final #
#####
-->
<condition id="final">
  <port side="forward" type="split">
    <connection source="" />
  </port>
</condition>
```

### 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**normal**), creamos una carpeta llamada **tmp** en nuestro **<HOME>**, para ello utilizaremos el comando (**mkdir**), como se muestra en el siguiente código:

```
$ mkdir $HOME/tmp
```

### 3° TERCE PASO

- Desde la consola de comando como usuario (**normal**), Creamos un archivo de python por ejemplo **Script\_graficos.py** ya sea en cualquier directorio en este caso en nuestro directorio **\$HOME**, para ello utilizaremos el comando (**touch**), como se muestra en la siguiente código:

```
$ touch $HOME/Script_graficos.py
```

## 4° CUARTO PASO

- Abrimos el archivo de python **Script\_graficos.py**, la cual se creo en el paso anterior. Insertamos el siguiente código:

```
# -*- coding: utf-8 -*-

import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia )
myinflow.setHostURL(myurl)

result = myinflow.login("admin","admin")

myconsult = u"operacion:Generar_gráfico_coloreado \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml" % (myhome)

if not result:
    print "Authentication failed"
    exit()

result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

print u"%s" % (myinflow.currentJSON())
```

## 5° QUINTO PASO

- Desde la consola de comando como usuario (**normal**), ejecutamos el archivo de python (**Script\_graficos.py**), como se muestra en el siguiente código:

```
$ python Script_graficos.py
```

---

**Nota:** Al ejecutar el archivo de python (**Script\_graficos.py**) nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales**:

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
```

```
'/home/cenditel/.safet/mydb.db'  
is still in use, all queries will cease to work.  
.....wheretokens: on  
.....newnode: |inicial| # Nueva condición inicial.  
.....newnode: |final| # Nueva condición final.  
qt_temp.XM6827.svg # Obtenemos la nueva imagen con su nombre  
# la cual contiene el gráfico.
```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constantemente.**

---

## 6° SEXTO PASO

- Desde la consola de comando como usuario (**normal**), accedemos directorio **\$HOME/tmp/**, para ello utilizaremos el comando (**cd**) como se muestra en el siguiente código.

```
$ cd $HOME/tmp
```

**Nota:** En el directorio **tmp/** se almacenan los gráficos (**svg ó png**), usted puede definir el directorio de escritura y temporal utilizando los siguientes parámetros en el archivo **safet.conf** que se encuentra en el directorio **\$HOME/.safet/**, como se muestra en el siguiente código:

```
# directorio para archivos temporales  
plugins.graphviz.infile = /home/fulano/tmp  
  
# directorio de salida para archivos de grafo (svg ó png)  
plugins.graphviz.outfile = /home/fulano/tmp
```

---

## 7° SEPTIMO PASO

- En la consola de comando escribimos el comando **ls** para ver las imágenes (**.svg**), como se muestra en el siguiente código:

```
tmp$ ls  
qt_temp.XM7929.svg
```

## 8° OCTAVO PASO

- Desde la consola de comando como usuario (**normal**), visualizamos las dos condiciones principales en la imagen **.svg (qt\_temp.XM6827.svg)**, como se muestra en el siguiente código:

```
tmp$ eog qt_temp.XM6827.svg
```

**Nota:** Si realizó los pasos correctos se mostrara el gráfico, en el cual se define lo siguiente:

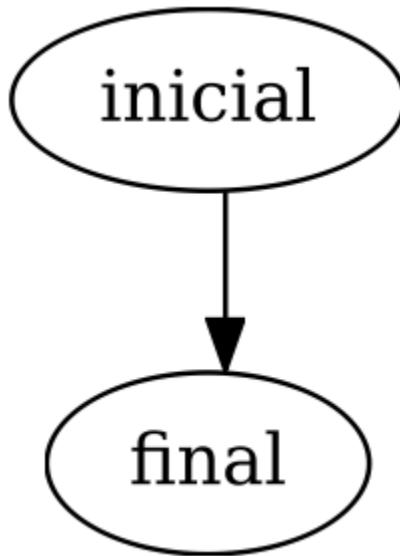


Figura 6.32: **Figura 42: Condiciones (inicial y final).**

En esta imagen no se muestran ninguna tarea ya que no hemos realizado ninguna, para ello pasamos al *E.- Tarea y procesos del flujo de trabajo (Registrado)* donde comenzaremos a realizar las tareas.

**Descripción de la Figura 42: Condiciones (inicial y final).:**

- El circulo **inicial** apunta al circulo **final**.

### 6.3.5 E.- Tarea y procesos del flujo de trabajo (Registrado)

En la primera tarea (**Registrado**) se utilizaran las siguientes **etiquetas (XML)**:

- **<connection>**: **Registrado** apunta a **final**, esto varia.
- **<task>**: Nombre de la tarea (**Registrado**) y su mensaje.
- **<port>**: Registrado apunta a una opción.
- **<variable>**: Variable **vRegistro** donde me aparecerá un mensaje la hora en el cual se hizo ese registro.

Ahora ejecutaremos los siguiente pasos:

#### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y insertamos el código (**xml**) de la **tarea Registrado** que aparece en el siguiente block:

```

<!--
*****
/ Condición inicial /
  
```

```
*****
-->
<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Registrado" source="Registrado"/>
  </port>
</condition>

<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">
  <port side="forward" type="split">
    <connection query="true" options="" source="final"/>
  </port>
  <variable
    config="1"
    documentsource="select id,nombre,status from productos"
    type="sql" tokenlink=""
    id="vRegistrado" rolfield="(select rol from productos_registro
      where productoid=productos.id and regstatus='Registrado') as rol"
    scope="task"
    timestampfield="(select fecha from productos_registro
      where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>

<!--
*****
/ Condición final /
*****
-->
<condition id="final">
  <port side="forward" type="split">
    <connection source="" />
  </port>
</condition>
```

---

### Nota: Archivo XML:

1. La condición **inicial** se modifico en la etiqueta **<connection>** la cual apuntará la tarea **Registrado**.
  2. En el *D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo* creamos la carpeta llamada **tmp** y el archivo llamado **.py (Script\_graficos.py)**, la cual se utilizaran en este paso a seguir.
-

## 2° SEGUNDO PASO

- Ejecutamos el mismo archivo .py (**Script\_graficos.py**) con el mismo contenido, desde la consola de comando como usuario normal.

```
$ python $HOME/Script_graficos.py
```

---

**Nota:** Al ejecutar el archivo .py (**Script\_graficos.py**) nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales** y la primera tarea (**Registrado**):

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
          '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Registrado| # Nueva tarea Registrado
.....newnode: |inicial|   # Condición inicial
.....newnode: |final|     # Condición final
qt_temp.XM6827.svg           # Obtenemos la nueva imagen con su
                             # nombre la cual contiene el gráfico.
```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constantemente.**

---

## 3° SEGUNDO PASO

- Nos vamos al directorio **\$HOME/tmp**,desde la consola de comando.

```
$ cd $HOME/tmp
```

## 4° CUARTO PASO

- Escribimos el comando **ls** para ver las 2 imágenes **.svg** que hemos obtenido ,desde la consola de comando.

```
tmp$ ls
qt_temp.XM6827.svg , qt_temp.XM6970.svg
```

## 5° QUINTO PASO

- Con el comando **eog** vemos la la segunda imagen **.svg(qt\_temp.XM6827.svg )** ,desde la consola de comando.

```
tmp$ eog qt_temp.XM6970.svg
```

---

**Nota:** Si realizó los pasos correctos, se mostrara el grafo como en la imagen siguiente *Figura 43: Registrado*:

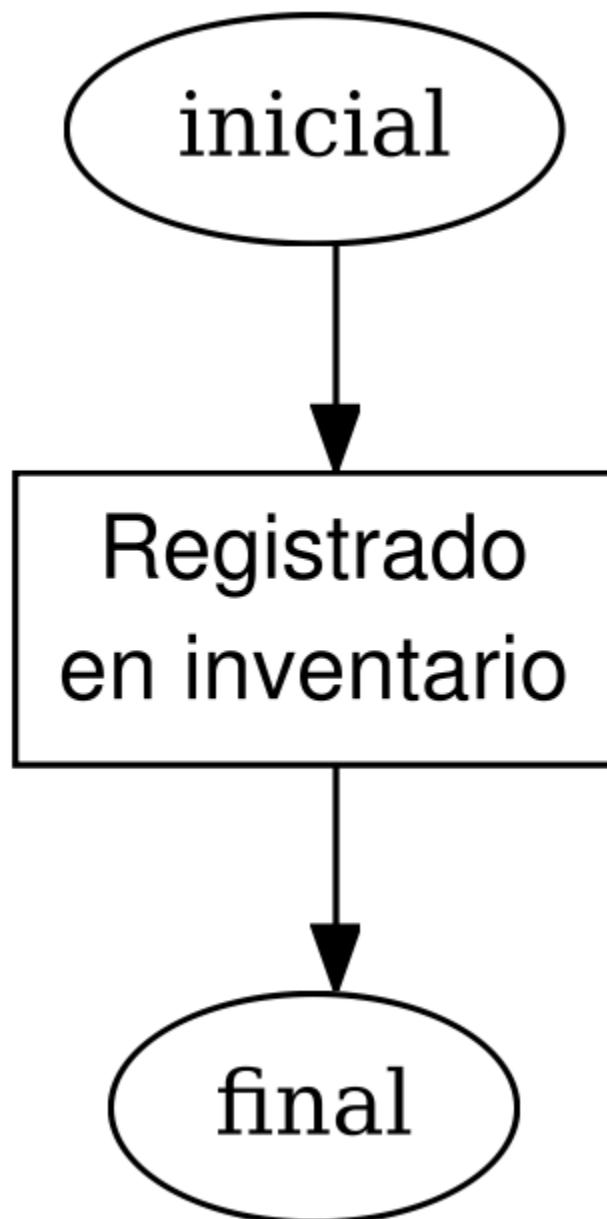


Figura 6.33: Figura 43: Registrado

Descripción de la *Figura 43: Registrado*:

- La condición del circulo **inicial** apunta a la primera tarea de cuadro **Registrado**.
  - La primera tarea **Registrado** apunta a la condición de circulo **final**
- 

### 6.3.6 F.- Tarea y procesos del flujo de trabajo (Pedido)

En la segunda tarea (**Pedido**) se utilizaran las siguientes **etiquetas (XML)**:

- **<connection>**: **Pedido** apunta a **final**, esto varia.
- **<task>**: Nombre de la tarea (**Pedido**)
- **<port>**: **Pedido** apunta a dos opciones con el operador **OR**.
- **<variable>**: Variable **vPedido** donde me aparecerá un mensaje la hora en el cual se hizo ese pedido.

Ahora ejecutaremos los siguiente pasos:

#### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y insertamos el código (**xml**) de la **tarea Pedido** que aparece en el siguiente block:

```

<!--
*****
/ Condición inicial /
*****
-->
<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="select status from productos"
      options="Registrado" source="Registrado"/>
  </port>
</condition>

<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">

  <port side="forward" type="split">
    <connection query="select status from productos"
      options="Pedido" source="Pedido"/>
  </port>

  <variable config="1"
    documentsource="select id,nombre,status from productos"

```

```
type="sql" tokenlink=""
id="vRegistrado" rolfield="(select rol from productos_registro
where productoid=productos.id and regstatus='Registrado') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>

<!---
*****
/ Pedido /
*****
--&gt;
&lt;task title="" id="Pedido" textualinfo=""&gt;

&lt;port pattern="none" side="forward" type="split"&gt;
  &lt;connection query="true" options="" source="final"/&gt;
&lt;/port&gt;

&lt;variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPedido" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Pedido') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Pedido') as fecha"/&gt;
&lt;/task&gt;

<!---
*****
/ Condición final /
*****
--&gt;
&lt;condition id="final"&gt;
  &lt;port side="forward" type="split"&gt;
    &lt;connection source="" /&gt;
  &lt;/port&gt;
&lt;/condition&gt;</pre>
```

---

### Nota: Archivo XML:

1. La tarea **Registrado** se modifico en la etiqueta **<connection>** la cual apuntará a la tarea **Pedido**.
  2. En el *D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo* creamos la carpeta llamada **tmp** y el archivo llamado **.py (Script\_graficos.py)**, la cual se utilizaran en este paso a seguir.
-

## 2° SEGUNDO PASO

- Ejecutamos el mismo archivo .py (**Script\_graficos.py**) con el mismo contenido, desde la consola de comando como usuario normal.

```
$ python $HOME/Script_graficos.py
```

---

**Nota:** Al ejecutar el archivo .py (**Script\_graficos.py**) nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales** y la dos tarea (**Registrado**),(**Pedido**):

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
                                '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Pedido|      # Nueva tarea Pedido
.....newnode: |Registrado|  # Tarea Registrado
.....newnode: |inicial|    # Condición inicial
.....newnode: |final|      # Condición final
qt_temp.XM5792.svg           # Obtenemos la nueva imagen con el
                             # nombre, la cual contiene el gráfico
```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constantemente.**

---

## 3° SEGUNDO PASO

- Nos vamos al directorio \$HOME/tmp,desde la consola de comando.

```
$ cd $HOME/tmp
```

## 4° CUARTO PASO

- Escribimos el comando ls para ver las 3 imágenes .svg que hemos obtenido ,desde la consola de comando.

```
tmp$ ls
qt_temp.XM6827.svg, qt_temp.XM6970.svg, qt_temp.XM5792.svg
```

## 5° QUINTO PASO

- Con el comando eog vemos la tercera imagen .svg(**qt\_temp.XM5792.svg**) ,desde la consola de comando.

```
tmp$ eog qt_temp.XM5792.svg
```

---

**Nota:** Si realizó los pasos correctos, se mostrara el grafo como en la imagen siguiente *Figura 44: Pedido*:

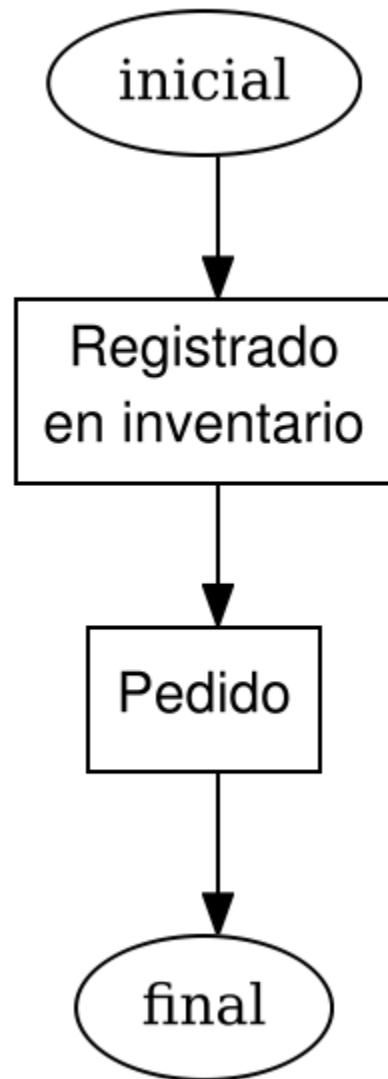


Figura 6.34: **Figura 44: Pedido**

Descripción de la *Figura 44: Pedido*:

- La condición del circulo **inicial** apunta a la primera tarea de cuadro **Registrado**.
- La segunda tarea **Pedido** apunta a la condición de circulo **final**

### 6.3.7 G.- Tarea y procesos del flujo de trabajo (Disponible)

En la tercera tarea (**Disponible**) se utilizaran las siguientes **etiquetas (XML)**:

- **<task>**: Nombre de la tarea (**Disponible**).
- **<port>**: Disponible apunta a una opción.
- **<connection>**: **Disponible** apunta a **final**, esto varia.
- **<variable>**: Variable **vDisponible** donde nos aparecerá (**nombre,id,status,fecha y hora**) de esa acción.

#### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y insertamos el código (**xml**) de la **tarea Disponible** que aparece en el siguiente block:

```

<!--
*****
/ Condición inicial /
*****
-->
<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Registrado" source="Registrado"/>
  </port>
</condition>

<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">

  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Pedido" source="Pedido"/>
  </port>

  <variable config="1"

```

```
documentsource="select id,nombre,status from productos"
type="sql" tokenlink=""
id="vRegistrado" rolfield="(select rol from productos_registro
where productoid=productos.id and regstatus='Registrado') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>

<!--
*****
/ Pedido /
*****
-->
<task title="" id="Pedido" textualinfo="">

<port pattern="or" side="forward" type="split">
<connection query="select status from productos"
options="Disponible" source="Disponible"/>
</port>

<variable config="1"
documentsource="select id,nombre,status from productos"
type="sql" tokenlink=""
id="vPedido" rolfield="(select rol from productos_registro
where productoid=productos.id and regstatus='Pedido') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Pedido') as fecha"/>
</task>

<!--
*****
/ Disponible /
*****
-->
<task title="" id="Disponible" textualinfo="">

<port pattern="none" side="forward" type="split">
<connection query="true" options="" source="final"/>
</port>

<variable config="1"
documentsource="select id,nombre,status from productos"
type="sql" tokenlink=""
id="vDisponible" rolfield="(select rol from productos_registro
where productoid=productos.id and regstatus='Disponible') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Disponible') as fecha"/>
</task>

<!--
*****
/ Condición final /
*****
```

```
*****
-->
<condition id="final">
  <port side="forward" type="split">
    <connection source="" />
  </port>
</condition>
```

---

**Nota:** Archivo XML:

1. La tarea **Pedido** se modifco las etiqueta **<port>** donde indicará que habran 2 opciones a ocurrir y la etiqueta **<connection>** la cual apuntará a la tarea **Disponible**.
  2. En el *D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo* creamos la carpeta llamada **tmp** y el archivo llamado **.py (Script\_graficos.py)**, la cual se utilizaran en este paso a seguir.
- 

## 2° SEGUNDO PASO

- Ejecutamos el mismo archivo **.py (Script\_graficos.py)** con el mismo contenido, desde la consola de comando como usuario normal.

```
$ python $HOME/Script_graficos.py
```

---

**Nota:** Al ejecutar el archivo **.py (Script\_graficos.py)** nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales** y la 3 tarea (**Registrado**),(**Pedido**),(**Disponible**):

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
          '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Disponible| # Nueva tarea
                  # (Primera opción) Disponible.
.....newnode: |Pedido|   # Tarea Pedido
.....newnode: |Registrado| # Tarea Registrado
.....newnode: |inicial|  # Condición inicial
.....newnode: |final|    # Condición final
qt_temp.XM6088.svg      # Obtenemos la nueva imagen con el
                         # nombre, la cual contiene el gráfico.
```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constante mente.**

---

## 3° SEGUNDO PASO

- Nos vamos al directorio **\$HOME/tmp**,desde la consola de comando.

```
$ cd $HOME/tmp
```

## 4° CUARTO PASO

- Escribimos el comando **ls** para ver las 4 imágenes **.svg** que hemos obtenido ,desde la consola de comando.

```
tmp$ ls  
qt_temp.XM6827.svg, qt_temp.XM6970.svg, qt_temp.XM5792.svg,  
qt_temp.XM6088.svg
```

## 5° QUINTO PASO

- Con el comando **eog** vemos la tercera imagen **.svg(qt\_temp.XM6088.svg)**, desde la consola de comando.

```
tmp$ eog qt_temp.XM6088.svg
```

---

**Nota:** Si realizó los pasos correctos, se mostrara el grafo como en la imagen siguiente *Figura 45: Disponible*

Descripción de la *Figura 45: Disponible*:

- La condición del circulo **inicial** apunta a la primera tarea de cuadro **Registrado**.
  - La primera tarea **Registrado** apunta a la segunda tarea **Pedido**.
  - La tercera tarea **Disponible** apunta a la condición de circulo **final**
- 

### 6.3.8 H.- Tarea y procesos del flujo de trabajo (Por\_llegar)

En la cuarta tarea (**Por\_llegar**) se utilizaran las siguientes **etiquetas (XML)**:

- **<connection>**: **Por\_llegar** apunta a **final**, esto varia.
- **<task>**: Nombre de la tarea (**Por\_llegar**).
- **<port>**: **Por\_llegar** apunta a la tarea (**Disponible**).
- **<variable>**: Variable **vPor\_llegar** donde nos aparecerá (**nombre,id,status,fecha y hora**) de esa acción.

Ahora ejecutaremos los siguiente pasos:

## 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y insertamos el código (**xml**) de la **tarea Por\_llegar** que aparece en el siguiente block:

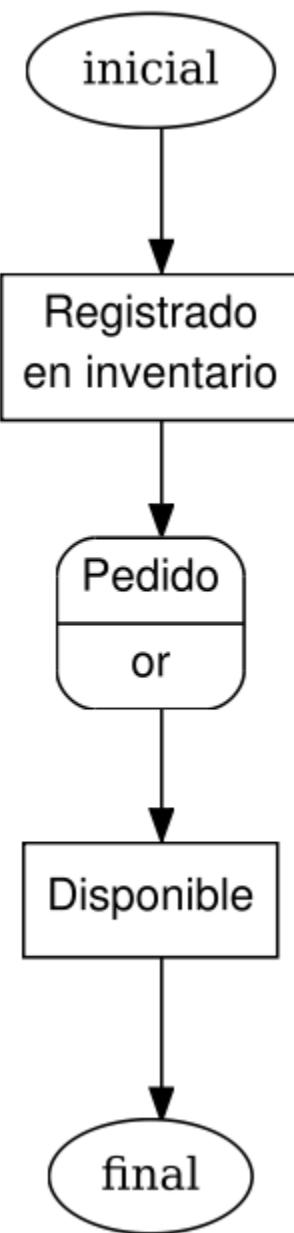


Figura 6.35: Figura 45: Disponible

```
<!--
*****
/ Condición inicial /
*****
-->
<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Registrado" source="Registrado"/>
  </port>
</condition>

<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">
  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Pedido" source="Pedido"/>
  </port>
  <variable config="1"
            documentsource="select id,nombre,status from productos"
            type="sql" tokenlink=""
            id="vRegistrado" rolfield="(select rol from productos_registro
                                         where productoid=productos.id and regstatus='Registrado') as rol"
            scope="task" timestampfield="(select fecha from productos_registro
                                         where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>

<!--
*****
/ Pedido /
*****
-->
<task title="" id="Pedido" textualinfo="">

  <port pattern="or" side="forward" type="split">
    <connection query="select status from productos"
                 options="Disponible" source="Disponible"/>
    <connection query="select status from productos"
                 options="Por_llegar" source="Por_llegar"/>
  </port>

  <variable config="1"
            documentsource="select id,nombre,status from productos"
            type="sql" tokenlink=""
            id="vPedido" rolfield="(select rol from productos_registro
                                         where productoid=productos.id and regstatus='Pedido') as rol"
            scope="task" timestampfield="(select fecha from productos_registro
                                         where productoid=productos.id and regstatus='Pedido') as fecha"/>
```

```
</task>

<!--
*****
/ Por_llegar /
*****
-->
<task title="" id="Por_llegar" textualinfo="">

<port pattern="none" side="forward" type="split">
  <connection query="select status from productos"
               options="Disponible" source="Disponible"/>
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPor_llegar" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Por_llegar') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Por_llegar') as fecha"/>
</task>

<!--
*****
/ Disponible /
*****
-->
<task title="" id="Disponible" textualinfo="">

<port pattern="none" side="forward" type="split">
  <connection query="true" options="" source="final"/>
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vDisponible" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Disponible') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Disponible') as fecha"/>
</task>

<!--
*****
/ Condición final /
*****
-->
<condition id="final">
  <port side="forward" type="split">
    <connection source="" />
  </port>
</condition>
```

---

### Nota: Archivo XML:

1. En la tarea **Pedido** se agrega otra etiqueta **<connection>** que seria segunda opción de tarea.
  2. En el *D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo* creamos la carpeta llamada **tmp** y el archivo llamado **.py (Script\_graficos.py)** ,la cual se utilizaran en este paso a seguir.
- 

## 2° SEGUNDO PASO

- Ejecutamos el mismo archivo **.py (Script\_graficos.py)** con el mismo contenido, desde la consola de comando como usuario normal.

```
$ python $HOME/Script_graficos.py
```

---

**Nota:** Al ejecutar el archivo **.py (Script\_graficos.py)** nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales** y la 4 tarea (**Registrado),(Pedido),(Disponible [OR] Por\_llegar**):

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
                               '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Disponible| # Tarea (Primera opción)
                           # Disponible.
.....newnode: |Pedido|    # Tarea Pedido
.....newnode: |Por_llegar| # Nueva tarea
                           # (Segunda opción) Por_llegar.
.....newnode: |Registrado| # Tarea Registrado
.....newnode: |inicial|   # Condición inicial
.....newnode: |final|     # Condición final
qt_temp.XM6368.svg        # Obtenemos la nueva imagen con el
                           # nombre, la cual contiene el gráfico
```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constante mente.**

---

## 3° SEGUNDO PASO

- Nos vamos al directorio **\$HOME/tmp**,desde la consola de comando.

```
$ cd $HOME/tmp
```

## 4° CUARTO PASO

- Escribimos el comando **ls** para ver las 5 imágenes **.svg** que hemos obtenido ,desde la consola de comando.

```
tmp$ ls  
qt_temp.XM6827.svg, qt_temp.XM6970.svg, qt_temp.XM5792.svg,  
qt_temp.XM6088.svg, qt_temp.XM6368.svg
```

## 5° QUINTO PASO

- Con el comando **eog** vemos la tercera imagen **.svg(qt\_temp.XM6368.svg)** ,desde la consola de comando.

```
tmp$ eog qt_temp.XM6368.svg
```

---

**Nota:** Si realizó los pasos correctos, se mostrara el grafo como en la imagen siguiente *Figura 46: Por\_llegar*

Descripción de la *Figura 46: Por\_llegar*:

- La condición del circulo **inicial** apunta a la primera tarea de cuadro **Registrado**.
  - La primera tarea **Registrado** apunta a la segunda tarea **Pedido**.
  - la cuarta tarea **Por\_llegar** apunta a la tercera tarea **Disponible**.
  - La tarcera tarea **Disponible** apunta a la condición **final**
- 

### 6.3.9 H.- Tarea y procesos del flujo de trabajo (Por\_agotarse)

En la quinta tarea (**Por\_agotarse**) se utilizaran las siguientes **etiquetas (XML)**:

- **<connection>**: **Por\_agotarse** apunta a **final**, esto varia.
- **<task>**: Nombre de la tarea (**Por\_agotarse**).
- **<port>**: **Por\_agotarse** apunta a una opción.
- **<variable>**: Variable **vPor\_agotarse** donde nos aparecerá (**nombre,id,status,fecha y hora**) de esa acción.

Ahora ejecutaremos los siguiente pasos:

## 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y insertamos el código (**xml**) de la **tarea Por\_agotarse** que aparece en el siguiente block:

```
<!--  
*****  
/ Condición inicial /  
*****  
-->
```

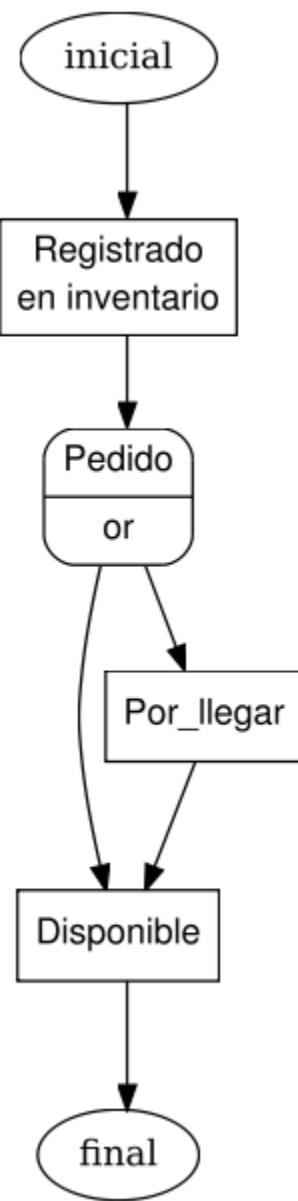


Figura 6.36: **Figura 46: Por\_llegar**

```

<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Registrado" source="Registrado"/>
  </port>
</condition>

<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">

  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Pedido" source="Pedido"/>
  </port>

  <variable config="1"
            documentsource="select id,nombre,status from productos"
            type="sql" tokenlink=""
            id="vRegistrado" rolfield="(select rol from productos_registro
                                         where productoid=productos.id and regstatus='Registrado') as rol"
            scope="task" timestampfield="(select fecha from productos_registro
                                         where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>

<!--
*****
/ Pedido /
*****
-->
<task title="" id="Pedido" textualinfo="">

  <port pattern="or" side="forward" type="split">
    <connection query="select status from productos"
                 options="Disponible" source="Disponible"/>
    <connection query="select status from productos"
                 options="Por_llegar" source="Por_llegar"/>
  </port>

  <variable config="1"
            documentsource="select id,nombre,status from productos"
            type="sql" tokenlink=""
            id="vPedido" rolfield="(select rol from productos_registro
                                         where productoid=productos.id and regstatus='Pedido') as rol"
            scope="task" timestampfield="(select fecha from productos_registro
                                         where productoid=productos.id and regstatus='Pedido') as fecha"/>
</task>

<!--

```

```
*****
/ Por_llegar /
*****
-->
<task title="" id="Por_llegar" textualinfo="">

<port pattern="none" side="forward" type="split">
    <connection query="select status from productos"
                  options="Disponible" source="Disponible"/>
</port>

<variable config="1"
    documentsource="select id,nombre,status from productos"
    type="sql" tokenlink=""
    id="vPor_llegar" rolfield="(select rol from productos_registro
    where productoid=productos.id and regstatus='Por_llegar') as rol"
    scope="task" timestampfield="(select fecha from productos_registro
    where productoid=productos.id and regstatus='Por_llegar') as fecha"/>
</task>

<!--
*****
/ Disponible /
*****
-->
<task title="" id="Disponible" textualinfo="">

<port pattern="none" side="forward" type="split">
    <connection query="select status from productos"
                  options="Por_agotarse" source="Por_agotarse"/>
</port>

<variable config="1"
    documentsource="select id,nombre,status from productos"
    type="sql" tokenlink=""
    id="vDisponible" rolfield="(select rol from productos_registro
    where productoid=productos.id and regstatus='Disponible') as rol"
    scope="task" timestampfield="(select fecha from productos_registro
    where productoid=productos.id and regstatus='Disponible') as fecha"/>
</task>

<!--
*****
/ Por_agotarse /
*****
-->
<task title="" id="Por_agotarse" textualinfo="">

<port pattern="none" side="forward" type="split">
    <connection query="true" options="" source="final"/>
</port>
```

```

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPor_agotarse" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Por_agotarse') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Por_agotarse')
  as fecha"/>
</task>

<!--
*****
/ Condición final /
*****
-->
<condition id="final">
  <port side="forward" type="split">
    <connection source="" />
  </port>
</condition>

```

**Nota:** Archivo XML:

1. En las opciones de tareas **Disponible** se modificará la etiqueta **<connection>** que apuntaran a la siguiente tarea **Por\_agotarse**.
2. En el *D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo* creamos la carpeta llamada **tmp** y el archivo llamado **.py (Script\_graficos.py)**, la cual se utilizaran en este paso a seguir.

**2° SEGUNDO PASO**

- Ejecutamos el mismo archivo **.py (Script\_graficos.py)** con el mismo contenido, desde la consola de comando como usuario normal:

```
$ python $HOME/Script_graficos.py
```

**Nota:** Al ejecutar el archivo **.py (Script\_graficos.py)** nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales** y la 4 tarea (**Registrado),(Pedido),(Por\_llegar [OR] Disponible**):

```

QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
          '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Disponible|      # Tarea (Primera opción) Disponible
.....newnode: |Pedido|          # Tarea Pedido
.....newnode: |Por_agotarse|    # Nueva tarea Por_agotarse
.....newnode: |Por_llegar|      # Tarea (Segunda opción) Por_llegar
.....newnode: |Registrado|     # Tarea Registrado
.....newnode: |inicial|        # Condición inicial

```

```
.....newnode: |final|
qt_temp.XM6667.svg
# Condición final
# Obtenemos la nueva imagen con el
# nombre, la cual contiene el gráfico.
```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constante mente.**

---

### 3° SEGUNDO PASO

- Nos vamos al directorio **\$HOME/tmp**,desde la consola de comando.

```
$ cd $HOME/tmp
```

### 4° CUARTO PASO

- Escribimos el comando **ls** para ver las 5 imágenes **.svg** que hemos obtenido ,desde la consola de comando.

```
tmp$ ls
qt_temp.XM6827.svg, qt_temp.XM6970.svg, qt_temp.XM5792.svg,
qt_temp.XM6088.svg, qt_temp.XM6368.svg, qt_temp.XM6667.svg
```

### 5° QUINTO PASO

- Con el comando **eog** vemos la tercera imagen **.svg(qt\_temp.XM6667.svg)** ,desde la consola de comando.

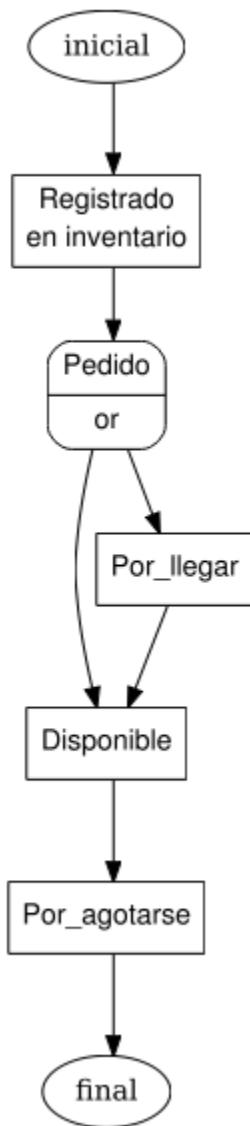
```
tmp$ eog qt_temp.XM6667.svg
```

---

**Nota:** Si realizó los pasos correctos, se mostrara el grafo como en la imagen siguiente *Figura 47: Por\_agotarse*

Descripción de la *Figura 47: Por\_agotarse*:

- La condición del circulo **inicial** apunta a la primera tarea de cuadro **Registrado**.
  - La primera tarea **Registrado** apunta a la segunda tarea **Pedido**.
  - la cuarta tarea **Por\_llegar** apunta a la tercera tarea **Disponible**.
  - La tarcera tarea **Disponible** apunta a la quinta tarea **Por\_agotarse**
  - La quinta tarea **Por\_agotarse** apunta a la condición **final**
-

Figura 6.37: **Figura 47: Por\_agotarse**

### 6.3.10 I.- Tarea y procesos del flujo de trabajo (Agotado)

En la sexta tarea (**Agotado**) se utilizaran las siguientes **etiquetas (XML)**:

- **<connection>**: Agotado apunta a (final, Pedido), esto varia.
- **<task>**: Nombre de la tarea (**Agotado**).
- **<port>**: Agotado apunta a dos opción.
- **<variable>**: Variable vAgotado donde nos aparecerá (nombre,id,status,fecha y hora) de esa acción.

Ahora ejecutaremos los siguiente pasos:

#### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** y insertamos el código (**xml**) de la tarea **Agotado** que aparece en el siguiente block:

```
<!--
*****
/ Condición inicial /
*****
-->
<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Registrado" source="Registrado"/>
  </port>
</condition>

<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">

  <port side="forward" type="split">
    <connection query="select status from productos"
                 options="Pedido" source="Pedido"/>
  </port>

  <variable config="1"
            documentsource="select id,nombre,status from productos"
            type="sql" tokenlink=""
            id="vRegistrado" rolfield="(select rol from productos_registro
                                         where productoid=productos.id and regstatus='Registrado') as rol"
            scope="task" timestampfield="(select fecha from productos_registro
                                         where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>

<!--
```

```

*****
/ Pedido /
*****
-->
<task title="" id="Pedido" textualinfo="">

<port pattern="or" side="forward" type="split">
  <connection query="select status from productos"
               options="Disponible" source="Disponible"/>
  <connection query="select status from productos"
               options="Por_llegar" source="Por_llegar"/>
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPedido" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Pedido') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Pedido') as fecha"/>
</task>

<!--
*****
/ Por_llegar /
*****
-->
<task title="" id="Por_llegar" textualinfo="">

<port pattern="none" side="forward" type="split">
  <connection query="select status from productos"
               options="Disponible" source="Disponible"/>
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPor_llegar" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Por_llegar') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Por_llegar') as fecha"/>
</task>

<!--
*****
/ Disponible /
*****
-->
<task title="" id="Disponible" textualinfo="">

<port pattern="none" side="forward" type="split">
  <connection query="select status from productos"
               options="Por_agotarse" source="Por_agotarse"/>

```

```
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vDisponible" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Disponible') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Disponible') as fecha"/>
</task>

<!--
*****
/ Por_agotarse /
*****
-->
<task title="" id="Por_agotarse" textualinfo="">

<port pattern="none" side="forward" type="split">
  <connection query="select status from productos"
               options="Agotado" source="Agotado"/>
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPor_agotarse" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Por_agotarse') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Por_agotarse')
  as fecha"/>
</task>

<!--
*****
/ Agotado /
*****
-->
<task title="" id="Agotado" textualinfo="">

<port pattern="none" side="forward" type="split">
  <connection query="true" options="" source="final"/>
  <connection query="select status from productos"
               options="Pedido" back="yes" source="Pedido"/>
</port>

<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vAgotado" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Agotado') as rol"
  scope="task" timestampfield="(select fecha from productos_registro
  where productoid=productos.id and regstatus='Agotado') as fecha"/>
```

```

</task>

<!--
*****
/ Condición final /
*****
-->
<condition id="final">
  <port side="forward" type="split">
    <connection source="" />
  </port>
</condition>

```

**Nota:** Archivo XML:

1. En la tarea **Por\_agotarse** se modificará la etiqueta **<connection>** que apuntaran a la siguiente tarea **Agotado**.
2. En el *D.- Condiciones principales (INICIAL Y FINAL) del flujo de trabajo* creamos la carpeta llamada **tmp** y el archivo llamado **.py (Script\_graficos.py)**,la cual se utilizaran en este paso a seguir.

**2° SEGUNDO PASO**

- Ejecutamos el mismo archivo **.py (Script\_graficos.py)** con el mismo contenido, desde la consola de comando como usuario normal.

```
$ python $HOME/Script_graficos.py
```

**Nota:** Al ejecutar el archivo **.py (Script\_graficos.py)** nos mostrara un mensaje donde salen reflejadas las 2 **Condiciones principales** y la 4 tarea (**Registrado),(Pedido),(Disponible [OR] Por\_llegar**):

```

QFSFileEngine::open: No file name specified
QSqlDatabasePrivate::removeDatabase: connection
      '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Agotado|      # Nueva tarea Agotado
.....newnode: |Disponible|    # Tarea (Primera opción) Disponible
.....newnode: |Pedido|        # Tarea Pedido
.....newnode: |Por_agotarse|  # Tarea Por_agotarse
.....newnode: |Por_llegar|    # Tarea (Segunda opción) Por_llegar
.....newnode: |Registrado|   # Tarea Registrado
.....newnode: |inicial|      # Condición inicial
.....newnode: |final|        # Condición final
qt_temp.XM6954.svg          # Obtenemos la nueva imagen con el
                             # nombre, la cual contiene el gráfico.

```

**El nombre de la imagen (.svg) es temporal, es decir su nombre varia constantemente.**

### 3° SEGUNDO PASO

- Nos vamos al directorio \$HOME/tmp, desde la consola de comando.

```
$ cd $HOME/tmp
```

### 4° CUARTO PASO

- Escribimos el comando ls para ver las 6 imágenes .svg que hemos obtenido ,desde la consola de comando.

```
tmp$ ls  
qt_temp.XM6827.svg, qt_temp.XM6970.svg, qt_temp.XM5792.svg,  
qt_temp.XM6088.svg, qt_temp.XM6368.svg, qt_temp.XM6667.svg,  
qt_temp.XM6954.svg
```

### 5° QUINTO PASO

- Con el comando eog vemos la tercera imagen .svg(qt\_temp.XM6954.svg) ,desde la consola de comando.

```
tmp$ eog qt_temp.XM6954.svg
```

---

**Nota:** Si realizó los pasos correctos, se mostrara el grafo como en la imagen siguiente *Figura 48: Agotado*

Descripción de la *Figura 48: Agotado*:

- La condición del circulo **inicial** apunta a la primera tarea de cuadro **Registrado**.
  - La primera tarea **Registrado** apunta a la segunda tarea **Pedido**.
  - la cuarta tarea **Por\_llegar** apunta a la tercera tarea **Disponible**.
  - La tarcera tarea **Disponible** apunta a la quinta tarea **Por\_agotarse**
  - La quinta tarea **Por\_agotarse** apunta a la sexta tarea **Agotado**
  - La sexta tarea **Agotado** apunta a la condición **final**
- 

**Nota:** Si durante el tutorial les ocurrió algún error, podemos descargar el archivo (**producto.xml**) en el siguiente enlace:

---

## 6.4 Ver datos usando flujos de trabajo

En esta sección de explicar las acciones necesarias para obtener reportes asociados a la configuración de datos del directorio **.safet/ (PySafet)**. Tomando como ejemplo el caso inventario explicaremos las acciones

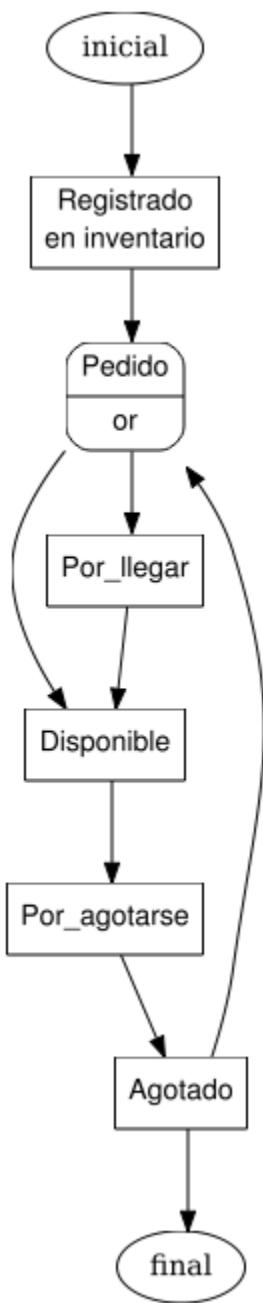


Figura 6.38: Figura 48: Agotado



Figura 6.39: Click Aquí (productos.xml)

a realizar para obtener un listado de todos los datos.

**A continuaciones mostraremos las siguientes acciones:**

#### 6.4.1 A.- PRIMERA ACCION (Tarea “vRegistrado”)

Con esta tarea se lista los productos que estén **registrados** o en estado **registrado**, la cual conocimos en los pasos anteriores. **Observen el siguiente código:**

```
<!--
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">

<port side="forward" type="split">
<connection query="true" options="" source="final"/>
</port>

<!-- nombre de las columnas(id,nombre,status),
     variable(id="vRegistrado") -->
<variable config="1"
documentsource="select id,nombre,status from productos"
type="sql" tokenlink="" id="vRegistrado"
rolfield="(select rol from productos_registro
where productoid=productos.id and regstatus='Registrado') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>
```

**A continuación seguimos los siguientes paso:**

## 1° PRIMER PASO

- Desde la consola de comando como usuario (**normal**), creamos un directorio en nuestro <HOME>, por ejemplo (**JsonInterfaz/**), como se muestra en el siguiente código:

```
$ mkdir $HOME/JsonInterfaz
```

## 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**normal**), accedemos al directorio **JsonInterfaz/** con el comando (**cd**), como se muestra en el siguiente código:

```
$ cd $HOME/JsonInterfaz
```

## 3° TERCER PASO

- Desde la consola de comando como usuario (**normal**), creamos un archivo **.py**, por ejemplo **Listar-Producto.py** con el comando (**touch**), como se muestra en el siguiente código:

```
JsonInterfaz$ touch ListarProducto.py
```

## 4° CUARTO PASO

- Abrimos el archivo **ListarProducto.py**, insertamos el siguiente código:

```
# -*- coding: utf-8 -*-

import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia )
myinflow.setHostURL(myurl)

myconsult = u"operacion:Listar_datos_general \
Cargar_archivo_flujo: /home/cenditel/.safet/flowfiles/productos.xml \
Variable: vRegistrado"

result = myinflow.login("admin","admin")

if not result:
    print "Authentication failed"
```

```
    exit()

result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

print u"%s" % (myinflow.currentJSON())
```

---

**Nota:** La operación seleccionada es “**Listar\_datos**”, el archivo donde se realiza la operación es “**productos.xml**” y el directorio donde se encuentra es <HOMR>.safet/flowfiles/.

La variable seleccionada “**vRegistrado**” se define los siguiente:

- La variable debe estar definida en el archivo “**productos.xml**”.
  - La operación **Listar\_datos** esta definida en el archivo **defconsole.xml** que se encuentra en el directorio <HOME>.safet/input/
- 

## 5° QUINTO PASO

- Ejecutamos el archivo **ListarProducto.py** desde la consola de comando como usuario normal, como se muestra en el siguiente código:

```
JsonInterfaz$ python ListarProducto.py
```

---

**Nota:** Al llamar el método **toInputConsole** con las opciones dicha anteriormente se generaría el siguiente mensaje **Json**.

```
QFSFileEngine::open: No file name specified

# Aquí nos indica que se esta conectando a la base de datos
# que esta en el directorio <HOME>.safet/
 QSqlDatabasePrivate::removeDatabase:
connection '/home/cenditel/.safet/mydb.db' is still in use,
          all queries will cease to work.

 QSqlDatabasePrivate::removeDatabase:
connection '/home/cenditel/.safet/mydb.db' is still in use,
          all queries will cease to work.

 QSqlDatabasePrivate::removeDatabase:
connection '/home/cenditel/.safet/mydb.db' is still in use,
          all queries will cease to work.

# La variable que esta utilizando
{s "safetvariable" : "vRegistrado",

"safetkey" : "",

"safettitle" : "vRegistrado",
"safetreport" : "operacion:Listar_datos_general \\\\"
```

```
Cargar_archivo_flujo:/home/cenditel/.safet/flowfiles/productos.xml \
Variable: vRegistrado",

# Aquí nos dice que hay dos registro en la db
"safetlistcount" : "2",

# Nos mostrara los datos en una lista
"safetlist" :
[
    # Aqui nos aparece el nombre de las columnas(id,nombre,status) y
    # sus datos
    {"id":"1", "id":"1", "nombre":"Jabón Protex", "status":"Registrado"}, 
    {"id":"2", "id":"2", "nombre":"Pañales Pamper", "status":"Registrado"}
], 

#Nos mostrara la columnas es decir el keys
"safetcolumns" :
[
    { "key": "id", "label": "id", "width": "10",
      "resizeable": "true", "sortable": "true"},

    { "key": "id", "label": "id", "width": "10",
      "resizeable": "true", "sortable": "true"},

    { "key": "nombre", "label": "nombre", "width": "90",
      "resizeable": "true", "sortable": "true"},

    { "key": "status", "label": "status", "width": "90",
      "resizeable": "true", "sortable": "true"}
]
```

---

#### 6.4.2 B.- Combinación de Json con HTML(javascript) con la variable (vRegistrado)

Listaremos los datos con interfaz gráfica de nuestro **Json**, con los siguientes pasos:

##### 1° PRIMER PASO

- Con el comando (**touch**) Creamos un archivo **.py** por ejemplo (**ScriptJson.py**), como se muestra en el siguiente código:

```
JsonInterfaz$ touch ScriptJson.py
```

##### 2° SEGUNDO PASO

- Abrimos el archivo que creamos **.py(****ScriptJson.py****)** insertamos el siguiente código:

```
#!/usr/bin python
# -*- coding: utf-8 -*-

# importamos la librerías a utilizar
import Safet
import os
import json

#función para convertir el json en datos con listas
def jsonToJquery(myoldarray):

    # Variables que me van a almacenar listar o arreglos
    mynewarray = []
    currcolumns = []
    currkeys = []

    # Una variable para almacenar
    stringcolumn = ""

    # Estructura de repetición para obtener
    # Las keys es decir los dato de
    #     las columnas (id,nombre,status)
    for reg in myoldarray:
        # reg.keys() me obtiene una en una la columnas
        currkeys = reg.keys()
        myvalue = ""
        regarray = []

        # Estructura de repetición anidada
        # Se obtienen los datos de la columnas
        #     es decir del (id,nombre,status)
        for currkey in currkeys:
            # optienes el datos
            myvalue = reg[currkey]
            # Insertamos el dato dentro de la lista regarray[]
            regarray.insert(0,myvalue)

        # Se añade la lista regarray[] es decir datos de las tablas
        mynewarray.append(regarray)

    #Estructura de repetición para obtener las columnas para javascript
    for currkey in currkeys:
        # Optenemos las columnas que se almacenan en un diccionario
        currcolumn = { "sTitle" : currkey }
        # Se inserta en la lista currcolumn[] el diccionario
        currcolumns.insert(0,currcolumn)

    # Retornamos la función con 2 listas
    # mynewarray[] que son los datos de la tablas o columnas
    # currcolumns[] que es un diccionario que contiene las columnas de
    #
    # la tablas de los datos
    return (mynewarray,currcolumns)
```

```

# función para escribir los datos obtenido en un archivo .js(javascript)
def writeJsonData(data,columns,Variable,filename):

    # u"%s" para almacenan los datos y las columnas
    stringnewarray = u"%s" % data
    stringcolumns = u"%s" % columns

    # se abre el archivo de tipo escritura
    file = open(filename, "w")

    # Escribimos el nombre de lista para datos y para las columnas
    datatowrite = u"dataInventory \
        = %s\n\ncolumnInventory = %s" % (stringnewarray, stringcolumns)

    # Se obtiene el valor de la variable
    variablewrite = "Variable = %s" % (Variable)

    # datatowrite.replace para que en el archivo
    #                 .js se me eliminar "u'"por "'"
    datatowrite = datatowrite.replace("u'", "'")
    variablewrite = variablewrite.replace("u'", "'")

    # salto de linea
    datatowrite = datatowrite + "\n"

    # Escribimos en archivo las dos listas
    file.write(datatowrite)
    file.write(variablewrite)

    # Cerramos el archivo
    file.close()

# Mi función principal
def Principal():

    # Se localiza en el directorio <HOME>
    myhome = os.getenv("HOME")
    mymedia = myhome + "/tmp"
    myurl = "http://localhost"

    # Mi constructor MainWindow
    myinflow = Safet.MainWindow(myhome)

    myinflow.setMediaPath(mymedia )
    myinflow.setHostURL(myurl)

    # Mi consulta de vRegistrado y su operación listar datos
    #que se encuentra en el directorio <HOME>.safe/flowfiles/
    myconsult = u"operacion:Listar_datos \
        Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml \
        Variable: vRegistrado " % (myhome)

```

```
# Nombre de usuario y si password
result = myinflow.login("admin", "admin")

if not result:
    print "Authentication failed"
    exit()

# Obtiene los el json de archivo toInputConsole.xml
result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

# Se obtiene mis datos json
mystringdata = u"%s" % myinflow.currentJSON();

# Me carga los datos json
mydata = json.loads(mystringdata)

# Llámanos a la función con el nombre jsonToJquery()
# para modificar el json
# Se le pasa 1 parámetro
# primero mydata con el nombre de la lista

# Se almacenan los datos en 2 variable
# La cual la función esta retornando 2 resultados
#           el array y las columnas del json
(mynewarray, currcolumns) = jsonToJquery(mydata["safetlist"])

#Llámanos a la función que la llámanos writeJsonData()
#           para insertar los datos en un archivo
#Se le pasan 3 parámetro:
#primero mynewarray
#Segundo currcolumns
#Tercero el nombre del archivo .js
writeJsonData(mynewarray, currcolumns,
              [mydata["safetvariable"]], "dataInventory.js")

if __name__ == "__main__":
    # Mi función principal
    Principal()
```

---

**Nota:** Este Script se utiliza para poder ver los datos con la librería jquery de manera más dinámica.

---

### 3º TERCER PASO

- Ejecutamos el archivo .py(**ScriptJson.py**) desde la consola de comando como usuario normal:

```
JsonInterfaz$ python ScriptJson.py
```

---

**Nota: Ejecución del ScriptJson.py:**

1. Nos mostrara el siguiente mensaje en la consola de comando, significando que funciono correctamente el Script:

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection '/home/cenditel/
 .safet/mydb.db' is still in use, all queries will cease to work.
 QSqlDatabasePrivate::removeDatabase: connection '/home/cenditel/
 .safet/mydb.db' is still in use, all queries will cease to work.
 QSqlDatabasePrivate::removeDatabase: connection '/home/cenditel/
 .safet/mydb.db' is still in use, all queries will cease to work.
```

2. AL ejecutar Script (**ScriptJson.py**) se genera un archivo **javascript** llamado **dataInventory.js** que contiene las columnas de la tabla y sus datos.
- 

## 4° CUARTO PASO

- Con el comando (**touch**) Creamos un archivo **.html** por ejemplo (**InventarioJson.html**), en el directorio **JsonInterfaz/**:

```
JsonInterfaz$ touch InventarioJson.html
```

## 5° QUINTO PASO

- Abrimos el archivo que creamos **.html(InventarioJson.html)**, insertamos el siguiente código:

```
<html>

<head>
    <link rel="stylesheet" type="text/css"
        href="http://ajax.aspnetcdn.com/ajax/
            jquery.dataTables/1.9.4/css/jquery.dataTables.css">
</head>

<body>

<!-- Aquí se crea la tabla dentro de la etiqueta &lt;body&gt;--&gt;
&lt;table cellpadding="0" cellspacing="0" border="0"
    class="display" name="safettable0" id="safettable0"&gt;
&lt;/table&gt;

&lt;script src="dataInventory.js"&gt; &lt;/script&gt;

&lt;script type="text/javascript" charset="utf8"
    src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.8.2.min.js"&gt;
&lt;/script&gt;</pre>
```

```
<script type="text/javascript" charset="utf8"
src="http://ajax.aspnetcdn.com/ajax/
jquery.dataTables/1.9.4/jquery.dataTables.min.js">
</script>

<script>

$(function () {
    $("#safetable0").dataTable({
        "aaData": dataInventory,
        "aoColumns": columnInventory
    });
})
</script>

</body>
</html>
```

**Nota:** En este código HTML se utilizó la librería (jquery) y la librería (jquery.dataTables) de manera interna. Para ver los datos mas dinámicos utilizando (javascript y json).

---

### 6° SEXTO PASO

- Abrimos con cualquier navegador web el archivo Json.html y no mostrara de la siguiente forma:  
*Figura 49: Json*

**Nota:**

Show 10 entries			Search:
id	nombre	status	
1	Jabón Protex	Registrado	
2	Pañales Pamper	Registrado	
3	Vitamina C	Registrado	
4	Vitamina D	Registrado	
5	Vitamina B12	Registrado	
6	Vitamina K	Registrado	
7	Vitamina B1	Registrado	
8	Vitamina B8	Registrado	

Showing 1 to 8 of 8 entries

◀ Previous Next ▶

Figura 6.40: **Figura 49: Json**

La siguiente *Figura 49: Json* indica lo siguiente:

- Nos mostrara el nombre de las columnas.
- Nos mostrara los datos de cada columna.

- Nos mostrara un buscador **Search**.
  - Nos mostrara **Show** que nos indicara cuantos datos quiere que se muestren en la tabla.
- 
- Aquí esta el ejemplo de (**vRegistrado**). Damos **Clik** a vRegistrado\_JSON.HTML

#### 6.4.3 C.- Nota variable (vPedido)

En el script de **Combinación de Json con HTML(javascript)** del *2º SEGUNDO PASO* en la función principal tenemos la consulta, la cual nos sirve para consultar los datos de las más variables, es decir, de las variables (“**vPedido**”, “**vDisponible**”, “**vPor\_llegar**”, “**vPor\_agotarse**”, “**vAgotado**”), con solo cambiarle el nombre de la variable.

Por ejemplo habíamos colocado anteriormente la variable “**vRegistrado**”, la cambiamos por “**vDisponible**” o la que sea.

Ya cambiado la variable solo se ejecuta el **Script** como en el *C.- Etiquetas principales del flujo de trabajo en el archivo xml (productos.xml)*, la cual al ejecutarlo nos generara el archivo **.js(SCRIPTJson.py)** con los datos y las columnas, como lo hacia con la variable “**vRegistrado**”, igualmente vemos en el navegador con el mismo código **HTML** del archivo (**InventarioJson.html**), que nos muestra el contenido con el mismo formato de las librerías (**jquery** y **jquery.dataTables**).

---

**Nota:** Si vamos a utilizar la variable “**vPedido**” tómese en cuenta lo siguiente:

---

#### 1º PRIMER PASO

- 1.- La variable (**vPedido**) necesita saber la **cantidad** de pedidos, para poder mostrar los datos.
- 2.- Abrimos el archivo **.xml(productos.xml)** de directorio <HOME>.safet/flowfiles/ y vemos los atributos (**id,nombre,status**):

```
<!--
*****
/ Pedido /
*****
-->
<task title="" id="Pedido" textualinfo="">
<port pattern="none" side="forward" type="split">
  <connection query="select status from productos"
               options="Disponible" source="Disponible"/>
  <connection query="select status from productos"
               options="Por_llegar" source="Por_llegar"/>
</port>

<!-- atributos (id,nombre,status) -->
<variable config="1"
  documentsource="select id,nombre,status from productos"
  type="sql" tokenlink=""
  id="vPedido" rolfield="(select rol from productos_registro
  where productoid=productos.id and regstatus='Pedido') as rol"
```

```
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Pedido') as fecha"/>
</task>
```

### 2° SEGUNDO PASO

- Agregamos el atributo **cantidad** en la etiquete (**variable**), como se muestra en el siguiente código:

```
<!--
*****
/ Pedido /
*****
-->
<task title="" id="Pedido" textualinfo="">
<port pattern="none" side="forward" type="split">
<connection query="select status from productos"
options="Disponible" source="Disponible"/>
<connection query="select status from productos"
options="Por_llegar" source="Por_llegar"/>
</port>

<!-- atributos (id,nombre,cantidad,status) -->
<variable config="1"
documentsource="select id,nombre,cantidad,status from productos"
type="sql" tokenlink=""
id="vPedido" rolfield="(select rol from productos_registro
where productoid=productos.id and regstatus='Pedido') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Pedido') as fecha"/>
</task>
```

### 3° TERCER PASO

- Listo ya podemos utilizar la variable “**vPedido**” para consultar, por ejemplo cuantos productos se han pedido. Claro colocamos la varible **vPedido** y luego ejecutando el Script del *2° SEGUNDO PASO*, como se muestra en siguiente *Figura 43: Registrado*

---

#### Nota:

Show 10 entries

cantidad	id	nombre	status
25	8	Vitamina B8	Pedido
30	6	Vitamina K	Pedido
30	4	Vitamina D	Pedido
220	2	Pañales Pamper	Pedido
300	5	Vitamina B12	Pedido
600	7	Vitamina B1	Pedido
1000	1	Jabón Protex	Pedido
1200	3	Vitamina C	Pedido

Showing 1 to 8 of 8 entries

Search:  Previous Next ▶

Figura 6.41: Figura 50: Json

La siguiente *Figura 50: Json* indica lo siguiente:

- Nos mostrara el nombre de las columnas, más el atributo **cantidad**.
  - Nos mostrara los datos de cada columna.
  - Nos mostrara un buscador **Search**.
  - Nos mostrara **Show** que nos indicara cuantos datos quiere que se muestren en la tabla.
- 
- Aquí esta el ejemplo de **(vPedido)**. Damos **Clik** a vPedido\_JSON.(HTML)

## 6.5 Ver fichas usando flujos de trabajo

En los pasos anteriores obtenemos los datos de dos **acciones** **vRegistrado** y **vPedido**, la cual utilizamos como ejemplo como se muestra en la siguiente imágenes:

**Nota: vRegistrado**

Show 10 entries			Search:
id	nombre	status	
1	Jabón Protex	Registrado	
2	Pañales Pamper	Registrado	
3	Vitamina C	Registrado	
4	Vitamina D	Registrado	
5	Vitamina B12	Registrado	
6	Vitamina K	Registrado	
7	Vitamina B1	Registrado	
8	Vitamina B8	Registrado	

Showing 1 to 8 of 8 entries

◀ Previous Next ▶

Figura 6.42: **Figura 51: vRegistrado**

La siguiente *Figura 51: vRegistrado* indica lo siguiente:

- Nos mostrara el nombre de las columnas.
- Nos mostrara los datos de cada columna.
- Nos mostrara un buscador **Search**.
- Nos mostrara **Show** que nos indicara cuantos datos quiere que se muestren en la tabla.

**Nota: vPedido**

La siguiente *Figura 52: vPedido* indica lo siguiente:

- Nos mostrara el nombre de las columnas, más el atributo **cantidad**.

Show 10 entries					Search:	
cantidad		id	nombre		status	
25		8	Vitamina B8		Pedido	
30		6	Vitamina K		Pedido	
30		4	Vitamina D		Pedido	
220		2	Pañales Pamper		Pedido	
300		5	Vitamina B12		Pedido	
600		7	Vitamina B1		Pedido	
1000		1	Jabón Protex		Pedido	
1200		3	Vitamina C		Pedido	

Showing 1 to 8 of 8 entries

◀ Previous Next ▶

Figura 6.43: Figura 52: vPedido

- Nos mostrara los datos de cada columna.
  - Nos mostrara un buscador **Search**.
  - Nos mostrara **Show** que nos indicara cuantos datos quiere que se muestren en la tabla.
- 

A continuación vamos a demostrar, como obtener fichas de esa tabla de datos:

### 6.5.1 Ficha (vRegistrado) (vPedido)

Los paso para obtener una ficha que vamos a seguir mas adelante, nos sirven para las demás acciones menos una, es decir, no solo para la acción **vRegistrado** sino tambien para las acciones (**vDisponible, vPor\_llegar, vPor\_agotarse, vAgotado**). Para la acción **vPedido** los pason son diferentes ya que ah esta acción se agrega el el atributo **cantidad**, porque para mostrar los datos depende del valor cantidad.

**Nota: Ficha:**

1.- La ficha (**vRegistrado**) sirve para las tareas (**vDisponible, vPor\_llegar, vPor\_agotarse, vAgotado**), la cual tienen los mismos atributos: **Observen la siguiente Figura 53: Ficha vRegistrado**

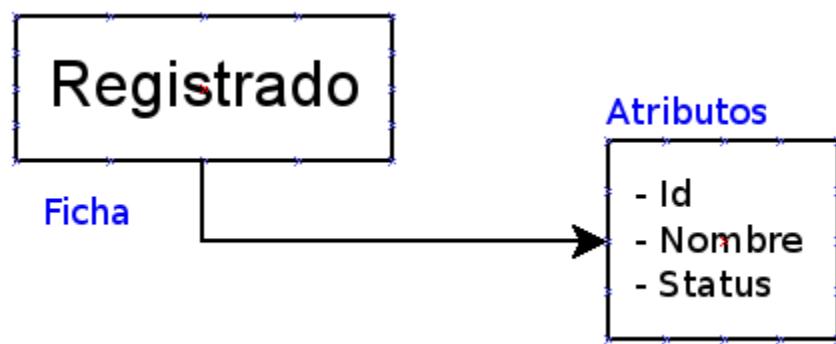


Figura 6.44: Figura 53: Ficha vRegistrado

2.- La ficha (**vPedido**) tiene ademas el atributo **cantidad**: **Observen la siguiente Figura 54: Ficha vPedido**

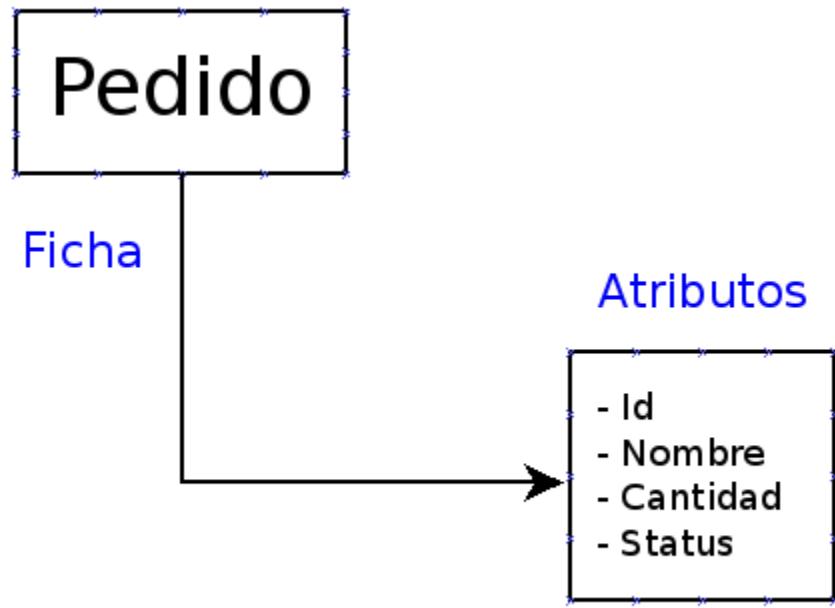


Figura 6.45: Figura 54: Ficha vPedido

---

A continuación seguimos los siguientes paso:

### 1° PRIMER PASO

- Nos vamos directorio que hemos creado anteriormente que se llama **JsonInterfaz**

```
$ cd $HOME/JsonInterfaz
```

### 2° SEGUNDO PASO

- Ejecutamos el Script que insertamos en el archivo que creamos anteriormente **.py(ScriptJson.py)**, con la variable (**vRegistrado**) o con la la variable **vPedido** o la que guste.

```
JsonInterfaz$ python ScriptJson.py
```

### 3° TERCER PASO

- Con el comando (**touch**) creamos un archivo **.css** por ejemplo (**style.css**), como se muestra en el siguiente código:

```
JsonInterfaz$ touch style.css
```

### 4° CUARTO PASO

- Abrimos el archivo que creamos .css(style.css), insertamos el siguiente código:

```
body {  
    background-repeat: no-repeat;  
    background-position: top center;  
    background-color: #657077;  
    margin: 40px;  
}  
  
#tabbed_box_1 {  
    margin: 0px auto 0px auto;  
    width: 300px;  
}  
.tabbed_box h4 {  
    font-family: Arial, Helvetica, sans-serif;  
    font-size: 23px;  
    color: #ffffff;  
    letter-spacing: -1px;  
    margin-bottom: 10px;  
}  
.tabbed_box h4 small {  
    color: #e3e9ec;  
    font-weight: normal;  
    font-size: 9px;  
    font-family: Verdana, Arial, Helvetica, sans-serif;  
    text-transform: uppercase;  
    position: relative;  
    top: -4px;  
    left: 6px;  
    letter-spacing: 0px;  
}  
.tabbed_area {  
    border: 1px solid #494e52;  
    background-color: #636d76;  
    padding: 8px;  
}  
  
ul.tabs {  
    margin: 0px; padding: 0px;  
    margin-top: 5px;  
    margin-bottom: 6px;  
}  
ul.tabs li {  
    list-style: none;  
    display: inline;  
}  
ul.tabs li a {  
    background-color: #464c54;
```

```
color:#ffebb5;
padding:8px 14px 8px 14px;
text-decoration:none;
font-size:9px;
font-family:Verdana, Arial, Helvetica, sans-serif;
font-weight:bold;
text-transform:uppercase;
border:1px solid #464c54;
background-repeat:repeat-x;
background-position:bottom;
}
ul.tabs li a:hover {
background-color:#2f343a;
border-color:#2f343a;
}
ul.tabs li a.active {
background-color:#ffffff;
color:#282e32;
border:1px solid #464c54;
border-bottom: 1px solid #ffffff;
background-repeat:repeat-x;
background-position:top;
}
.content {
background-color:#ffffff;
padding:10px;
border:1px solid #464c54;
font-family:Arial, Helvetica, sans-serif;
background-repeat:repeat-x;
background-position:bottom;
}
#content_2, #content_3 { display:none; }

.content ul {
margin:0px;
padding:0px 20px 0px 20px;
}
.content ul li {
list-style:none;
border-bottom:1px solid #d6dde0;
padding-top:15px;
padding-bottom:15px;
font-size:13px;
}
.content ul li:last-child {
border-bottom:none;
}
.content ul li a {
text-decoration:none;
color:#3e4346;
}
.content ul li a small {
color:#8b959c;
```

```
    font-size:9px;
    text-transform:uppercase;
    font-family:Verdana, Arial, Helvetica, sans-serif;
    position:relative;
    left:4px;
    top:0px;
}
.content ul li a:hover {
    color:#a59c83;
}
.content ul li a:hover small {
    color:#baae8e;
}
```

## 5° QUINTO PASO

- Con el comando (**touch**) creamos un archivo **.html** por ejemplo (**Fichas.html**), como se muestra en el siguiente código:

```
JsonInterfaz$ touch FichavRegistrado.html
```

## 6° SEXTO PASO

- Abrimos el archivo que creamos **.html(Fichas.html)**, insertamos el siguiente código:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
      xml:lang="en" lang="en">
<head>

<meta http-equiv="Content-Type" content="text/html;
      charset=ISO-8859-1">
<title>Fichas</title>
<link rel="stylesheet" type="text/css" media="screen">

<script src="fichas/dataInventory.js"> </script>

<script>

function Fichas() {

    var valor = columnInventory.length;
    var valor1 = dataInventory.length-1;
    var j = 0, flecha = false;

    if (columnInventory[j].sTitle == "cantidad") {
```

```

flecha = true;

for (var i = 0; i < valor; i++ ){

if( flecha == true && i == 2){

document.write("<li> <font color = 'blue'>" +
+ columnInventory[j].sTitle + ": </font>" +
+ dataInventory[valor1][j]+ "</li>");

document.write("<li> <font color = 'blue'>" +
+ columnInventory[i+1].sTitle + ": </font>" +
+ dataInventory[valor1][i+1]+ "</li>");

}

else{

document.write("<li> <font color = 'blue'>" +
+ columnInventory[i+1].sTitle + ": </font>" +
+ dataInventory[valor1][i+1]+ "</li>");

}

}

}

else{

for (var i = 0; i < valor; i++){

document.write("<li> <font color = 'blue'>" +
+ columnInventory[i].sTitle + ": </font>" +
+ dataInventory[j][i]+ "</li>");

}

}

}

</script>
</head>
<body>

<div id="tabbed_box_1" class="tabbed_box">
<h4>Ficha de la tabla
<script> document.write(Variable[0]);
</script></h4>

<div class="tabbed_area">
<ul class="tabs">
<li><a class="ta ctive">Ficha</a></li>

</ul>

<div id="content_1" class="content">
<ul>
<script> Fichas();</script>
</ul>

</div>
</div>

```

```
</div>  
</body>  
</html>
```

---

**Nota:** Este código sirve para todas la variables, se utilizo estilo css y se muestran los datos del archivo dataInventory.js utilizando código javascript.

---

## 7° SEPTIMO PASO

- Abrimos el archivo **FichavRegistrado.html** con cualquier navegador **Web** y se nos mostrara de la siguiente fichas:

---

**Nota: Fichas:**

- 1.- La *Figura 55: Ficha de un registro* nos muestra los datos de una fila de la tabla **vRegistrado**, por ejemplo los datos de la primera fila en forma de una ficha:

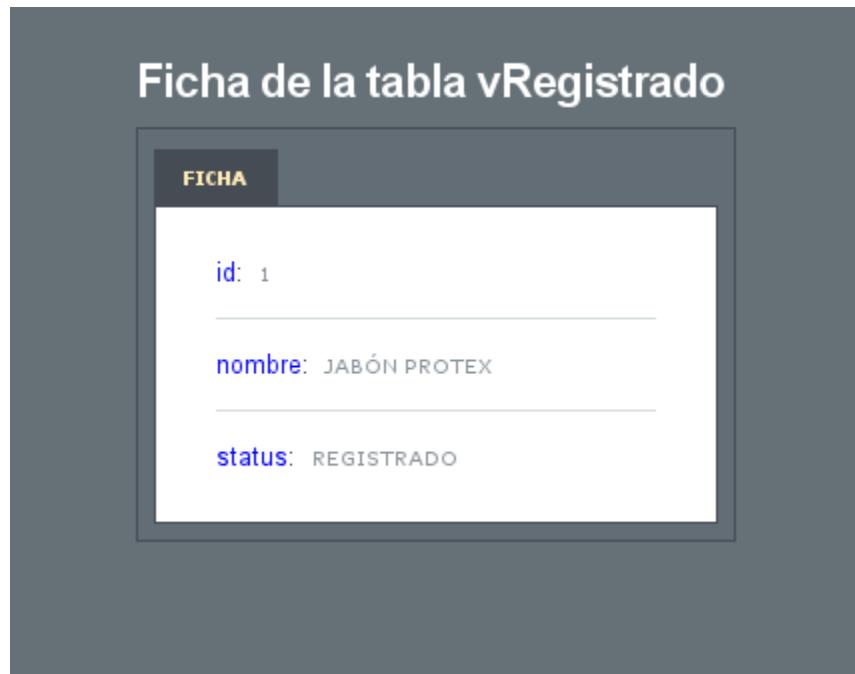


Figura 6.46: **Figura 55: Ficha de un registro**

La *Figura 55: Ficha de un registro* indica lo siguiente:

- Nos mostrara el **id** y su valor
- Nos mostrara el **Nombre** y su valor.
- Nos mostrara el **Status** y su valor.

2.- La *Figura 56: Ficha de un pedido* nos muestra los datos de una fila de la tabla **vPedido**, por ejemplo los datos de la primera fila en forma de una ficha:



Figura 6.47: **Figura 56: Ficha de un pedido**

La *Figura 56: Ficha de un pedido* indica lo siguiente:

- Nos mostrara el **id** y su valor
- Nos mostrara el **Nombre** y su valor.
- Nos mostrara el **Cantidad** y su valor.
- Nos mostrara el **Status** y su valor.

---

Aquí esta los 2 ejemplos de las fichas le damos (click) algunas de ellas:

1.- FichavRegistrado.html

2.- FichavPedido.html

## 6.6 Ver gráficos coloreado usando flujos de trabajo

Los gráficos coloreados se definen de dos manera, la primera es que se puede obtener de modo general y la segunda que se puede obtener por clave, es decir, un valor en específico.

En el punto anterior del tutorial *Crear el flujo de trabajo utilizando un archivo XML* habíamos creado un directorio **tmp** y un archivo **.py**(**Script\_graficos.py**) en nuestro <HOME> la cual el directorio se ubicaban las (**imagenes o gráficos**) y el **Script** para generarlos, de modo que vamos a utilizar ese directorio y ese archivo para esta parte del tutorial.

A continuaciones comenzaremos a obtener los dos tipos de gráficos:

### 6.6.1 A.- Gráficos coloreado general

Este gráfico nos mostrara todo los estados en general con todo el contenido y de forma más dinamica para ello seguimos los siguientes pasos:

#### 1° PRIMER PASO

- Abrimos el archivo **.py**(**Script\_graficos.py**), insertamos la operación **Generar\_gráfico\_coloreado** como aparece en el siguiente código:

```
# -*- coding: utf-8 -*-

import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia )
myinflow.setHostURL(myurl)

result = myinflow.login("admin","admin")

# Operación Generar_gráfico_coloreado
myconsult = u"operacion:Generar_gráfico_coloreado \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml" % (myhome)

if not result:
    print "Authentication failed"
    exit()

result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

print u"%s" % (myinflow.currentJSON())
```

---

**Nota:** Para obtener el gráfico general debe tener la siguiente consulta:

- **Operacion:** Generar\_gráfico\_coloreado
  - **Cargar\_archivo\_flujo:** %s/.safet/flowfiles/productos.xml
- 

## 2° SEGUNDO PASO

- Ejecutamos el archivo ya mencionado anteriormente **.py**(**Script\_graficos.py**) desde la consola de comando:

```
$ python $HOME/Script_graficos.py
```

---

**Nota:** Al ejecutar el archivo **.py** (**Script\_graficos.py**) nos mostrara el siguiente mensaje con todas las **tareas** y la nueva imagen generada:

```
QFSFileEngine::open: No file name specified
QSqlDatabasePrivate::removeDatabase: connection
                                '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Agotado|      # Tarea Agotado
.....newnode: |Disponible|    # Tarea (Primera opción) Disponible
.....newnode: |Pedido|        # Tarea Pedido
.....newnode: |Por_agotarse|  # Tarea Por_agotarse
.....newnode: |Por_llegar|    # Tarea (Segunda opción) Por_llegar
.....newnode: |Registrado|   # Tarea Registrado
.....newnode: |inicial|      # Condición inicial
.....newnode: |final|        # Condición final
qt_temp.XM5838.svg          # Obtenemos la nueva imagen con el
                            # nombre, la cual contiene el gráfico.
```

---

## 3° TERCER PASO

- Nos vamos al directorio **tmp** para ver la (**imagen o gráfico**):

```
$ cd $HOME/tmp/
```

## 4° CUARTO PASO

- Abrimos la (**imagen o gráfico**) que obtuvimos cuando ejecutamos el **Script\_graficos.py** con el comando **eog** :

```
tmp$ eog qt_temp.XM5838.svg
```

---

**Nota:** Aquí nos muestra en la siguiente imagen el: *Figura 57: Gráficos coloreado general.*:

La imagen *Figura 57: Gráficos coloreado general.* se define lo siguiente:

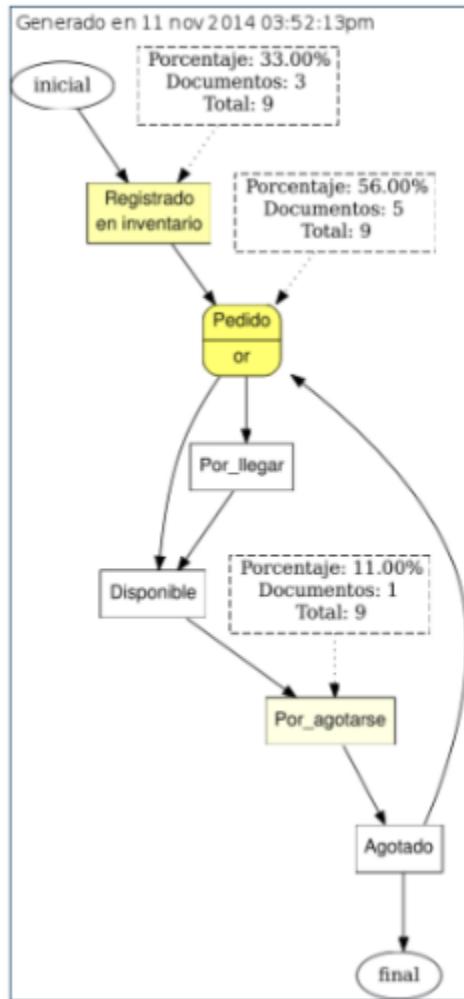


Figura 6.48: **Figura 57: Gráficos coloreado general.**

1. Cada estado que este ocurriendo algun evento se remarcara con un color en este caso en color amarillo, ese color puede cambiarse.
  2. Cada **estado** se refleja un mensaje indicando especificaciones de los eventos.
  3. Los estado se muestran de forma rectangular esto puede cambiar.
- 

## 6.6.2 B.- Gráficos por clave

Este gráfico nos mostrara una tarea en especifica la cual nos indicara por cuales tarea a recorrido el producto como tal para ello seguimos los siguientes pasos:

### 1° PRIMER PASO

- Abrimos el archivo **.py**(**Script\_graficos.py**), insertamos la operación **Generar\_gráfico\_para\_clave** y su **clave**, como aparece en el siguiente código:

```
# -*- coding: utf-8 -*-

import Safet
import os

myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia)
myinflow.setHostURL(myurl)

result = myinflow.login("admin", "admin")

# Operación Generar_gráfico_para_clave
myconsult = u"operacion::Generar_gráfico_para_clave \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml \
    Clave: 1" % (myhome)

if not result:
    print "Authentication failed"
    exit()

result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

print u"%s" % (myinflow.currentJSON())
```

---

**Nota:** Para obtener el gráfico general debe tener la siguiente consulta:

- **Operacion:** enerar\_gráfico\_para\_clave
  - **Cargar\_archivo\_flujo:** %s/.safet/flowfiles/productos.xml
  - **Clave:** (1) Es decir el número de producto.
- 

### 2° SEGUNDO PASO

- Ejecutamos el archivo .py(**Script\_graficos.py**) desde la consola de comando:

```
$ python $HOME/Script_graficos.py
```

**Nota:** Al ejecutar el archivo .py (**Script\_graficos.py**) nos mostrara el siguiente mensaje con todas las **tareas** y la nueva imagen generada:

```
QFSFileEngine::open: No file name specified
QSqlDatabasePrivate::removeDatabase: connection
          '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Agotado|      # Tarea Agotado
.....newnode: |Disponible|    # Tarea (Primera opción) Disponible
.....newnode: |Pedido|       # Tarea Pedido
.....newnode: |Por_agotarse| # Tarea Por_agotarse
.....newnode: |Por_llegar|   # Tarea (Segunda opción) Por_llegar
.....newnode: |Registrado|  # Tarea Registrado
.....newnode: |inicial|     # Condición inicial
.....newnode: |final|       # Condición final
qt_temp.XM8417.svg          # Obtenemos la nueva imagen con el
                            # nombre, la cual contiene el gráfico.
```

---

### 3° TERCER PASO

- Nos vamos al directorio **tmp** para ver la (**imagen o gráfico**):

```
$ cd $HOME/tmp/
```

### 4° CUARTO PASO

- Abrimos la (**imagen o gráfico**) que obtuvimos cuando ejecutamos el **Script\_graficos.py** con el comando **eog** :

```
tmp$ eog qt_temp.XM8417.svg
```

**Nota:** Aquí nos muestra en la siguiente imagen el: *Figura 58: Gráficos por clave.*:

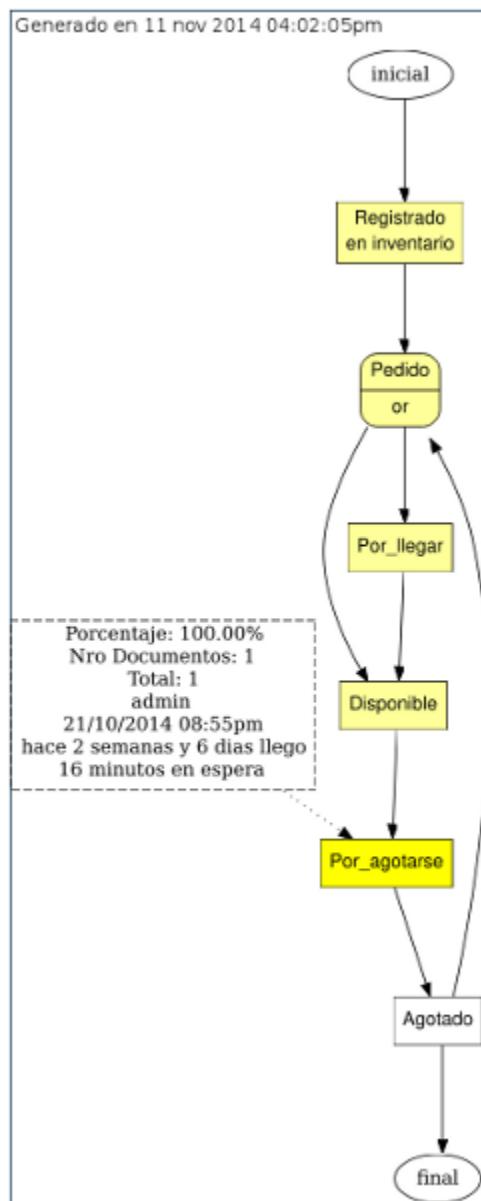


Figura 6.49: Figura 58: Gráficos por clave.

La imagen *Figura 58: Gráficos por clave.* se define lo siguiente:

1. Aparece algunas tareas remarcados con color amarillo significando que pasado por esos eventos.
  2. Aparece un mensaje con ciertas especificaciones del evento.
- 

### 6.6.3 C.- Opciones de Gráficos (general o por “clave”)

En el *3º TERCER PASO* de **gráficos general** vimos el gráfico en forma normal la cual se puede cambiar de diferentes formas, para ello debemos cambiar algunas opciones de **[Plugins.Graphviz]** en el archivo **safet.conf** que esta en el directorio <HOME>.safet/, seguimos los siguientes pasos:

#### 1º PRIMER PASO

- Abrimos el archivo **safet.conf** y nos vamos a las opciones de **[Plugins.Graphviz]** como se muestra en el siguiente código:

```
# Opciones particulares de cada complemento
[Plugins.Graphviz]

plugins.graphviz.graphtype = svg

# Información a mostrar en cuadro de información extra
# (Porc, Tokens, Total, InfoText, InfoDate)
plugins.graphviz.extrainfo.show = Porc,Tokens,Total,InfoText,InfoDate

#Color activo de para ser utilizado en los grficos
plugins.graphviz.task.fillcolor = #f1f1f1

#Color activo de la linea para ser utilizado en los grficos
plugins.graphviz.task.color = black

# Atributo utilizado en la estadstica Opciones posibles
# (Color/Size/Line/Custom)
plugins.graphviz.stats.attribute = Color

# Tamao mximo para la estadstica de (Tamano Maximo)
plugins.graphviz.stats.sizemax = 2

# Tamao mnimo para la estadstica de (Tamano Minimo)
plugins.graphviz.stats.sizemin = 1

# Color para dibujar estadistica
plugins.graphviz.stats.colorgradient = yellow

# Color del texto para cuadro de estadística
plugins.graphviz.stats.fontcolor = black

# Color de fondo para cuadro de estadística
```

```

plugins.graphviz.stats.fillcolor = antiquewhite

# Estilo de la linea del cuadro de estadística
plugins.graphviz.stats.style = dashed

# Mostrar cuadro de estadística
plugins.graphviz.showstats = yes

#Dirección del grafo TB (Arriba-Abajo) LR (Izquierda-Derecha)
plugins.graphviz.graph.rankdir = TB

# Tamaño del fuente para todos los nodos
plugins.graphviz.graph.fontsize = 12

# Separador del rank
plugins.graphviz.graph.ranksep = 0.5 equally

# Figura para la tarea (Task)
plugins.graphviz.task.shape = box

# Estilo de la Figura para la tarea (Task) filled,bold,dotted,empty
plugins.graphviz.task.style = filled

#Color activo de para ser utilizado en los gráficos
plugins.graphviz.condition.fillcolor = #FFFFFF

#Color activo de la linea para ser utilizado en los gráficos
plugins.graphviz.condition.color = black

# Figura para la (Condition) (box, ellipse, circle, etc.)
plugins.graphviz.condition.shape = ellipse

# Estilo de la Figura para la tarea (Condition)
plugins.graphviz.condition.style = filled

# Mostrar la información extra solo donde existan tokens (fichas)

plugins.graphviz.extrainfo.showwheretokens = on

```

## 2° SEGUNDO PASO

Cambiamos algunas opciones archivo **safet.conf** por ejemplo:

1. **Dirección del grafo:** se nos muestra el gráfico de (**Arriba-Abajo**) con el comando **TB**, cambiamos el comando a **LR** que es (**Izquierda-Derecha**) como nos muestra en el siguiente código:

```

#Dirección del grafo TB (Arriba-Abajo) LR (Izquierda-Derecha)
plugins.graphviz.graph.rankdir = LR

```

---

**Nota:** Al cambiar la opción, se nos mostrara el resultado como en la siguiente *Figura 59: LR(Izquierda-Derecha)*:

---

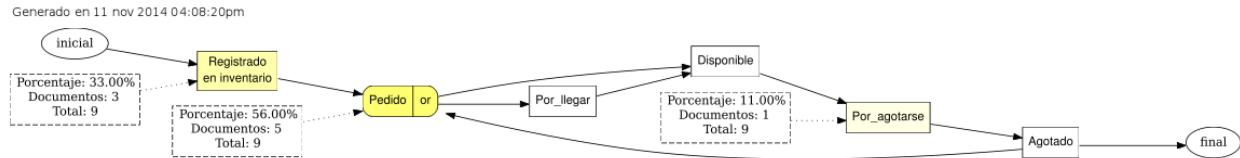


Figura 6.50: Figura 59: LR(Izquierda-Derecha)

- Color activo de la línea para ser utilizado en los gráficos por ejemplo (**black, Blue, mediumseagreen, etc.**):

```
plugins.graphviz.task.color = Blue
```

**Nota:** Al cambiar la opción, se nos mostrara el resultado como en la siguiente *Figura 60: Color activo de la línea*:

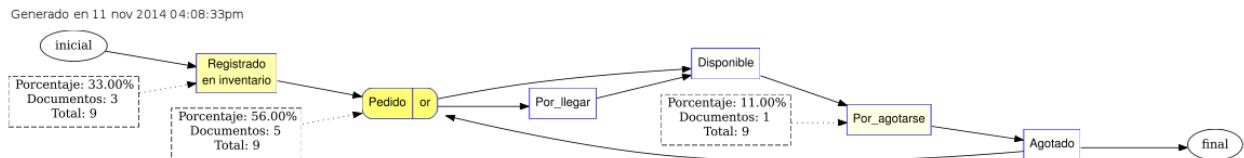


Figura 6.51: Figura 60: Color activo de la línea

- Color del texto para cuadro de estadística por ejemplo (**black, Blue, chocolate**):

```
plugins.graphviz.stats.fontcolor = chocolate
```

**Nota:** Al cambiar la opción, se nos mostrara el resultado como en la siguiente *Figura 61: Color del texto estadístico*:

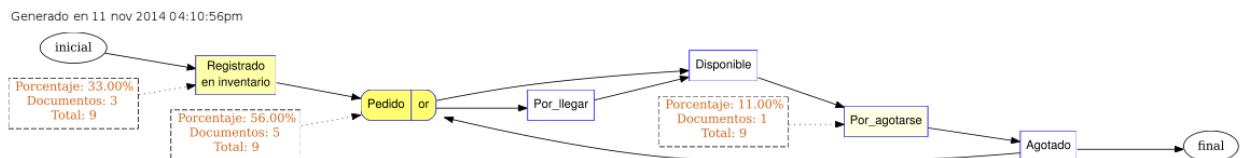


Figura 6.52: Figura 61: Color del texto estadístico

- Figura para la tarea (Task) por ejemplo (**box, ellipse, circle, etc.**):

```
plugins.graphviz.task.shape = ellipse
```

**Nota:** Al cambiar la opción, se nos mostrara el resultado como en la siguiente *Figura 62: Tareas en forma de circulo*:

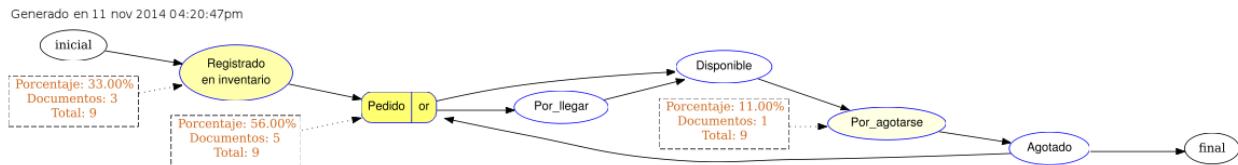


Figura 6.53: Figura 62: Tareas en forma de circulo

5. Color para dibujar estadistica por ejemplo (**salmon,yellow, blue, #FFFFFF , etc.**):

```
plugins.graphviz.stats.colorgradient = salmon
```

**Nota:** Al cambiar la opción, se nos mostrara el resultado como en la siguiente *Figura 63: Color de fondo de las tareas*:

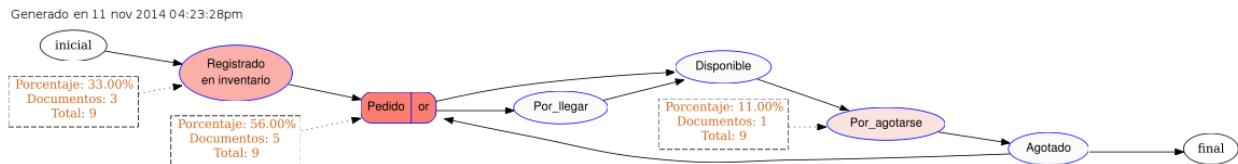


Figura 6.54: Figura 63: Color de fondo de las tareas

6. Figura para la (Condition) por ejemplo (**box, ellipse, circle, etc.**):

```
plugins.graphviz.condition.shape = box
```

**Nota:** Al cambiar la opción, se nos mostrara el resultado como en la siguiente *Figura 64: Condiciones en forma de cuadrado*:

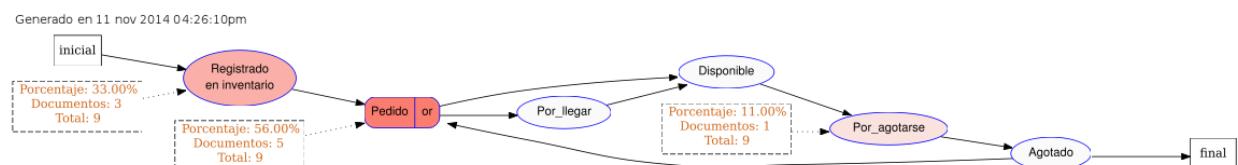


Figura 6.55: Figura 64: Condiciones en forma de cuadrado

A media que se cambie alguna configuración el archivo (**safet.conf**) realizamos los pasos (3, 4 y 5):

### 3° TERCER PASO

- Ejecutamos de nuevo el Script (**Script\_graficos.py**) y obtenemos la imagen con los gráfico:

```
$ python $HOME/Script_graficos.py
```

### 4° CUARTO PASO

- Nos vamos al directorio **tmp** para ver la (**imagen o gráfico**):

```
$ cd $HOME/tmp/
```

### 5° QUINTO PASO

- Abrimos la (**imagen o gráfico**) que obtuvimos cuando ejecutamos el **Script\_graficos.py** con el comando **eog** :

## 6.7 Parámetros usando flujos de trabajo

El uso de parámetros son útiles en una base de datos que contiene muchos datos la cual te permite buscar datos, con solo colocar un parámetro a buscar, por ejemplo una letra o un numero. También te permite opciones gráficas de [Plugins.Graphviz] del archivo **safet.conf**, que sean mas dinámicas.

A continuación vemos la configuración del ejemplo del inventario con dos parámetros:

### 6.7.1 A.- Parámetro de configuración

Al utilizar este parámetros de configuración se hace más dinámico para el usuario, es decir, no necesita accediendo al archivo **safet.conf** en el directorio <HOME>.safet/ para realizar las configuraciones de apariencia gráficas, con solo ejecutar un **Script python**.

A continuación se mostraran algunos ejemplo usando parámetro de configuración:

### 1° Dirección del gráfico

**A.- Nombre de la apariencia y su valor:**

```
# TB (Arriba-Abajo) LR (Izquierda-Derecha)
plugins.graphviz.graph.rankdir = LR
```

**B.- Código (xml) para el archivo (productos.xml):**

```
<parameter
    title      = "Direccion_Grafico_de_Flujo"
    options    = "TB::La direccin del grfico es arriba-abajo::TB,\n              LR::La direccin del grfico es izquierda-derecha::LR"
    configurekey = "Plugins.Graphviz/plugins.graphviz.graph.rankdir"
    type       = "combolistliteral"
    mandatory   = "no"
/>
```

**C.- Código del Script (python):**

```
myconsult = u"operacion:Generar_gráfico_coloreado \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml\
    configurekey.Plugins.Graphviz/plugins.graphviz.stats.fontcolor:LR \
    " % (myhome)
```

**D.- Resultado de la apariencia Figura 65: Gráficos.:**

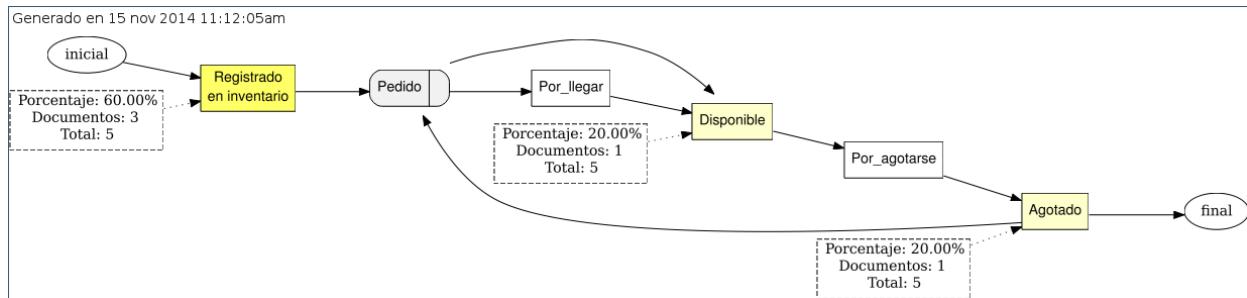


Figura 6.56: **Figura 65: Gráficos.**

**2º Color del texto para cuadro de estadística**

**A.- Nombre de la apariencia y su valor:**

```
# Color del texto
plugins.graphviz.stats.fontcolor = chocolate
```

**B.- Código (xml) para el archivo (productos.xml):**

```
<parameter
    title      = "Direccion_Grafico_de_Flujo"
    options    = ""
    configurekey = "Plugins.Graphviz/plugins.graphviz.stats.fontcolor"
    type       = "string"
    mandatory  = "no"
/>
```

**C.- Código del Script (python):**

```
myconsult = u"operacion:Generar_gráfico_coloreado \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml\
    configurekey.Plugins.Graphviz/plugins.graphviz.stats.fontcolor: \
    chocolate " % (myhome)
```

**D.- Resultado de la apariencia Figura 66: Gráficos Estadístico.:**

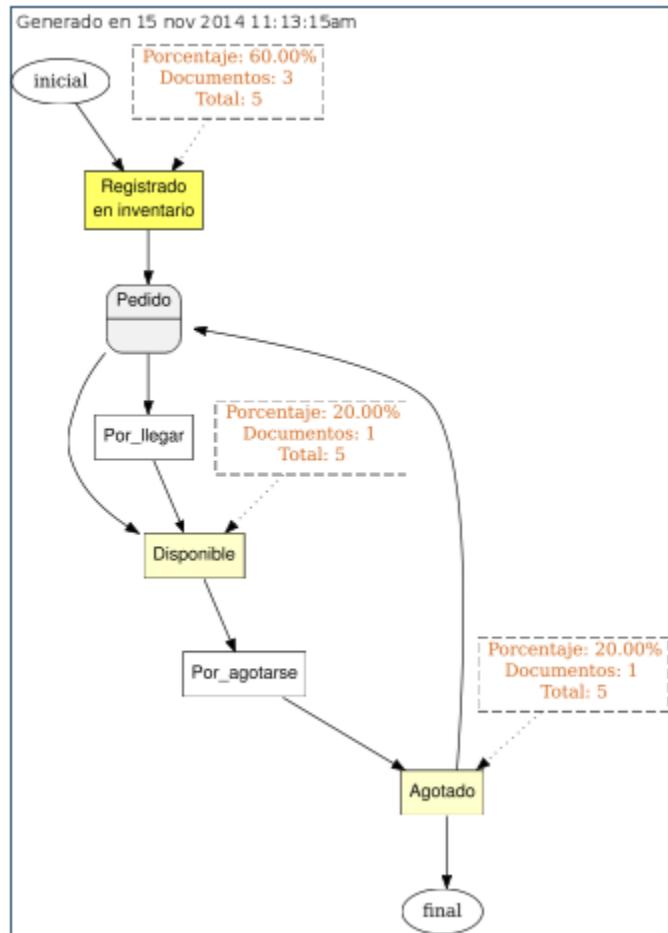


Figura 6.57: **Figura 66: Gráficos Estadistico.**

### 3º Figura para la (Condition)

#### A.- Nombre de la apariencia y su valor:

```
#Color activo
plugins.graphviz.task.shape = ellipse
```

#### B.- Código (xml) para el archivo (productos.xml):

```
<parameter
    title      = "Direccion_Grafico_de_Flujo"
    options    = ""
    configurekey = "Plugins.Graphviz/plugins.graphviz.task.shape"
    type       = "string"
    mandatory   = "no"
/>
```

#### C.- Código del Script (python):

```
myconsult = u"operacion:Generar_gráfico_coloreado \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml \
    configurekey.Plugins.Graphviz/plugins.graphviz.task.shape: \
    ellipse " % (myhome)
```

#### D.- Resultado de la apariencia *Figura 67: Gráficos condición.*:

## 6.7.2 B.- Ejecución de los parámetros de flujo de trabajo

### 1º PRIMER PASO

- Abrimos el archivo **productos.xml** del directorio <HOME>.safet/flowfiles/, insertamos la etiqueta **<parameter>** debajo de la etiqueta **<token>** como aparece en el siguiente código:

```
<yawl version="0.01">
  <workflow id="productos">
    <token keysource="productos" key="id"/>

    <!-- Insertamos este código
    ****
    / Parámetros de apariencias /
    ****
-->

<parameter
    title      = "Direccion_Grafico_de_Flujo"
    options    = "TB::La direccin del grfico es arriba-abajo::TB, \
                LR::La direccin del grfico es izquierda-derecha::LR"
    configurekey = "Plugins.Graphviz/plugins.graphviz.graph.rankdir"
    type       = "combolistliteral"
```

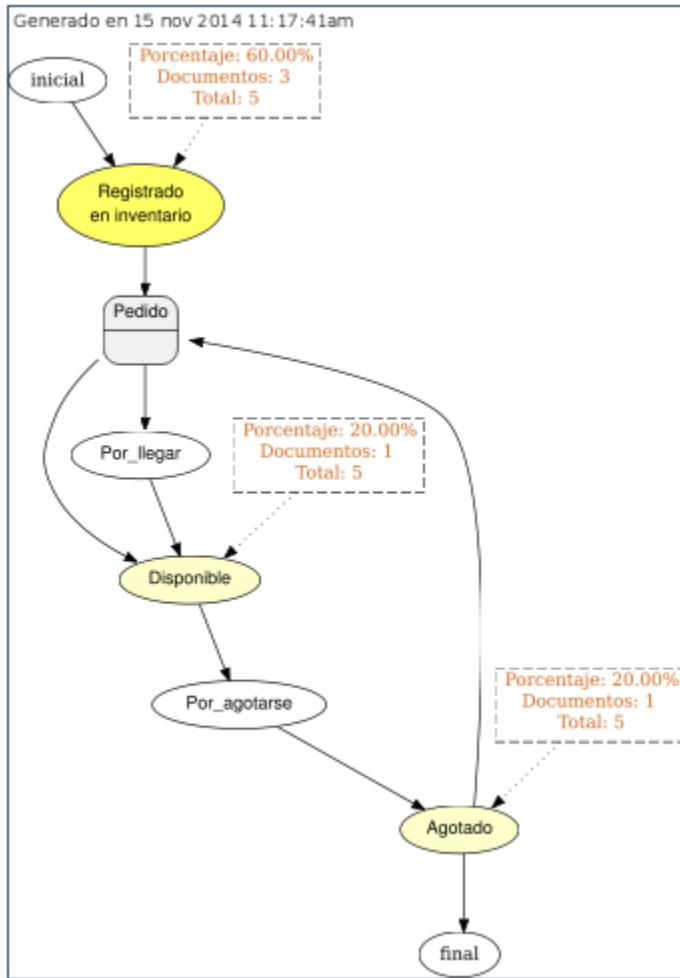


Figura 6.58: Figura 67: Gráficos condición.

```
mandatory = "no"
/>
```

**Nota:** Esta etiqueta “<parameter>” de configuración de apariencia de gráficos se define lo siguiente:

1. **title:** En este parámetro se le agrega cualquier nombre la cual va a almacena la operación a realizar.
2. **options:** Aquí se validan los valores que van a utilizar.
3. **configurekey:** En este parámetro es la configuración en el archivo **safet.conf**, la cual vamos a utilizar la apariencia de gráfico. Para agregar la configuración del gráficos en **configurekey** se debe hacer lo siguiente:
  - Se coloca el nombre de la opción a usar [**Plugins.Graphviz**] del archivo **safe.conf**.
  - Luego se coloca un “/”
  - Se coloca el nombre de la del estilo a utilizar en este ejemplo usamos el estilo **plugins.graphviz.graph.rankdir**, que es la Dirección del grafo.
4. **type:** “combolistliteral”, que admite solo valores.
5. **mandatory:** Parámetro si es obligatorio o no.

## 2° SEGUNDO PASO

- Abrimos el mismo archivo python de parámetros de datos (**Script\_parametro.py**), cambiamos la consulta como se muestra en el siguiente código:

```
import Safet
import os
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia )
myinflow.setHostURL(myurl)

# Consulta con el parametro
myconsult = u"operacion:Generar_gráfico_coloreado \
Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml\
configurekey.Plugins.Graphviz/plugins.graphviz.graph.rankdir:LR \
" % (myhome)

result = myinflow.login("admin", "admin")

if not result:
    print "Authentication failed"
```

```
exit()

result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

print u"%s" % (myinflow.currentJSON())
```

---

**Nota:** En este código python **myconsult** tiene los siguiente parámetros:

- **operacion:Generar\_gráfico\_coloreado:** Operación para obtener los gráficos.
  - **Cargar\_archivo\_flujo: %s/.safet/flowfiles/productos.xml:** Ruta del archivo donde esta el **parametro**
  - **configurekey.Plugins.Graphviz/plugins.graphviz.graph.rankdir:LR:** (configurekey) es la configuración que va a tomar del archivo **safet.conf** de apariencias, puede ser cualquier apariencia con su valor.
- 

### 3° TERCER PASO

- Ejecutamos el archivo python (**Script\_parametro.py**), desde la consola de comando como usuario normal:

```
$ python $HOME/Script_parametro.py
```

---

**Nota:** Al ejecutar el archivo **.py (Script\_parametro.py)** nos mostrara el siguiente mensaje con todas las **tareas** y la nueva imagen generada:

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
                                '/home/cenditel/.safet/mydb.db'
is still in use, all queries will cease to work.
.....wheretokens: on
.....newnode: |Agotado|      # Tarea Agotado
.....newnode: |Disponible|    # Tarea (Primera opción) Disponible
.....newnode: |Pedido|        # Tarea Pedido
.....newnode: |Por_agotarse|  # Tarea Por_agotarse
.....newnode: |Por_llegar|    # Tarea (Segunda opción) Por_llegar
.....newnode: |Registrado|   # Tarea Registrado
.....newnode: |inicial|      # Condición inicial
.....newnode: |final|        # Condición final
qt_temp.T16691.svg          # Obtenemos la nueva imagen con el
                            # nombre, la cual contiene el gráfico.
```

---

## 4° CUARTO PASO

- Nos vamos al directorio **tmp** para ver la (**imagen o gráfico**):

```
$ cd ${HOME}/tmp/
```

- Abrimos la (**imagen o gráfico**) que obtuvimos cuando ejecutamos el **Script\_parametro.py** con el comando **eog**:

```
tmp$ eog qt_temp.T16691.svg
```

**Nota:** Aquí nos muestra en la siguiente imagen el: *Figura 68: Gráficos.*:

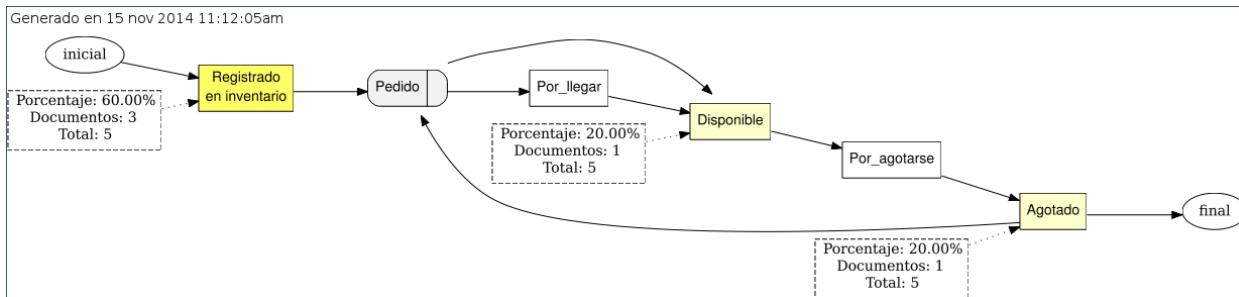


Figura 6.59: *Figura 68: Gráficos.*

La imagen *Figura 68: Gráficos.* se define lo siguiente:

- Vamos la imagen en la dirección contraria es decir de (**Izquierda-Derecha**).
- Nos muestra en banco pro no le especificamos ningún dato.

### 6.7.3 C.- Ejecución de parámetro para reportes de datos

Este parámetro te permite buscar cualquier dato de una base de datos especificando su **estado**. Para manejar este parámetro debemos seguir los siguientes paso:

## 1° PRIMER PASO

- Abrimos el archivo **productos.xml** del directorio **<HOME>.safet/flowfiles/**, Borramos la condición inicial como se muestra en el siguiente código:

```
<!--
*****
/ Condición inicial /
*****
-->
<condition type="start" id="inicial">
<port side="forward" type="split">
```

```
<connection
    query="select status from productos"
    options="Registrado"
    source="Registrado"
  />
</port>
</condition>
```

### 2° SEGUNDO PASO

- Insertamos el siguiente código donde estaba la condición inicial:

```
<!--
*****
/ Parametros de datos /
*****
-->
<parameter title="Empieza_por" options="" type="string" mandatory="no"/>

<condition type="start" id="inicial">
  <port side="forward" type="split">
    <connection
      query="select nombre from productos"
      options="LIKE '{#Empieza_por}%'"
      source="Por_nombre"/>
  </port>
</condition>

<task title="Por_nombre" id="Por_nombre">
  <port side="forward" type="split">
    <connection
      query="select status from productos"
      options="Registrado"
      source="Registrado"/>
  </port>
</task>

<variable config="1" documentsource="select nombre,status
from productos" type="sql" tokenlink="" id="vEmpieza"
rolfield = "(select rol from productos_registro where
  productoid = productos.id and regstatus='Registrado') as rol"
scope = "task"
timestampfield="(select fecha from productos_registro where
  productoid=productos.id and regstatus='Registrado') as fecha"
/>
</task>
```

---

**Nota:** Ejemplo del parámetro, que afecte a los datos del flujo (ej. Empieza\_por)

1. La etiquete **<parameter>** se define los siguiente:

- **title:** Nombre del parámetro en este ejemplo le colocamos **Emperzar\_por**, es opcional.

- **options:** En este caso no va realizar ningún opción.
- **type:** Tipo de dato que se desea que se muestre o retorne en este caso cadena, puede ser cualquiera.
- **mandatory:** Es para que el campo se obligatorio (“**no**”, “**yes**”), en este ejemplo colocamos “no”

2. Condición inicial: **options=”LIKE ‘{#Empieza\_por} %’”:**

- **LIKE(COMO):** Permite que empiece por cualquier letra, es decir, que se pareca pero que no sea igual a la palabra. Si no se le coloca entonces tienes que colocar tal cual como es la palabra a buscar para poder que la busque.
- ‘**{#Empieza\_por}**’: El parámetro que se le paso que es **Emperzar\_por**, tiene que entre comillas simples y entre llaves con el # y el nombre el parámetro.
- “**%**”: Se le coloca este porcentaje, que quiere decir, busca todo las palabras que comienzan por parámetro que le estas pasando. También esta = igual, la cual tiene que decirle que busque todo igual al parámetro que le estas pasando.

### 3° TERCER PASO

- Creamos una archivo python por ejemplo (**Script\_parametro.py**):

```
$ touch $HOME/Script_parametro.py
```

- Abrimos el archivo python creado (**Script\_parametro.py**), insertamos el siguiente código:

```
import Safet
import os
myhome = os.getenv("HOME")
mymedia = myhome + "/tmp"
myurl = "http://localhost"

myinflow = Safet.MainWindow(myhome)

myinflow.setMediaPath(mymedia )
myinflow.setHostURL(myurl)

# Consulta con el parametro
myconsult = u"operacion>Listar_datos \
    Cargar_archivo_flujo: %s/.safet/flowfiles/productos.xml\
    Variable: vPedido\
    parameters.Empieza_por: v " % (myhome)

result = myinflow.login("admin","admin")

if not result:
    print "Authentication failed"
    exit()
```

```
result = myinflow.toInputConsole(myconsult)

if not result:
    print "Consult failed error: %s" % (myinflow.currentError())
    exit()

print u"%s" % (myinflow.currentJSON())
```

---

**Nota:** En este código python **myconsult** tiene los siguiente parámetros:

- **operacion:Listar\_datos:** Vamos a listar datos con la operación **Listar\_datos**
  - **Cargar\_archivo\_flujo: %s/.safet/flowfiles/productos.xml:** Ruta del archivo donde esta el **parametro**
  - **Variable: vPedido:** Vamos a tomar como ejemplo la variable **Pedido** para listar sus datos a buscar. Esta variable puede ser cualquiera (**vRegistrado,vDisponible,vPor\_agotarse,vPor\_llegar,vAgotado**).
  - **parameters.Empieza\_por: v:** tomando como ejemplo de búsqueda, le decimos que nos busque todos los productos pedidos que comiencen por la letra v.
- 

## 4° CUARTO PASO

- Ejecutamos el archivo python (**Script\_parametro.py**), desde la consola de comando como usuario normal:

```
$ python $HOME/Script_parametro.py
```

---

**Nota:** Al ejecutar el archivo python nos dice que hay 3 productos pedidos que comienzan con la letra v, en caso de que no allá ningún producto que comience con la letra v no habrá ningún resultado, vemos el resultado del script:

```
QFSFileEngine::open: No file name specified
 QSqlDatabasePrivate::removeDatabase: connection
 '/home/cenditel/.safet/mydb.db' is still in use,
 all queries will cease to work.
 QSqlDatabasePrivate::removeDatabase: connection
 '/home/cenditel/.safet/mydb.db' is still in use,
 all queries will cease to work.
 QSqlDatabasePrivate::removeDatabase: connection
 '/home/cenditel/.safet/mydb.db' is still in use,
 all queries will cease to work.
{
    "safetvariable" : "vRegistrado",
    "safetkey" : "",
    "safetttitle" : "vRegistrado",
    "safetreport" : "operacion:Listar_datos
Cargar_archivo_flujo: /home/cenditel/.safet/flowfiles/productos.xml
Variable: vRegistrado",
```

```
"safetlistcount" : "3",
"safetlist" : [
    {"id" : "3", "nombre" : "Vitamina C", "cantidad" : "", "status" : "Registrado"}, 
    {"id" : "5", "nombre" : "Vitamina B12", "cantidad" : "", "status" : "Registrado"}, 
    {"id" : "6", "nombre" : "Vitamina K", "cantidad" : "", "status" : "Registrado"}],
"safetcolumns" : [
    { "key": "id", "label": "id", "width": "10", "resizeable": "true", "sortable": "true"}, 
    { "key": "nombre", "label": "nombre", "width": "90", "resizeable": "true", "sortable": "true"}, 
    { "key": "cantidad", "label": "cantidad", "width": "10", "resizeable": "true", "sortable": "true"}, 
    { "key": "status", "label": "status", "width": "90", "resizeable": "true", "sortable": "true"}]
}
```

---



## 7.- Tutorial del “ejemplo de inventario” usando la interfaz gráfica (Inflow)

---

### 7.1 Introducción a inflow

#### 7.1.1 A.- Iniciar la interfaz gráfica de inflow

##### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

##### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 69: Autenticación de Inflow*.

#### 7.1.2 B.- Barra de menú (Inflow)

##### 1.- Botón (Archivo)

En el botón (**Archivo**) tenemos las siguientes opciones:

1. **Cargar configuración:** Permite cargar un nuevo proyecto (.tar).
2. **Guardar configuración:** Permite guardar nuestro proyecto en formato (.tar).
3. **Cargar configuración FTP:** Permite cargar FTP.
4. **Guardar configuración FTP:** Permite guardar FTP en formato (.tar).
5. **Cambiar usuario:** Permite cambiarnos de usuario.



Figura 7.1: **Figura 69:** Autenticación de Inflow.

6. **Imprimir:** Permite guardar la captura en formato pdf.
7. **Vista preliminar:** Permite ver la información de manera amplia.
8. **Cerrar:** Salir de la herramienta Inflow.

Observe la siguiente [Figura 70: Menú \(Archivo\)](#).

## 2.- Botón (Editar)

En el botón (Editar) tenemos las siguientes opciones:

1. **Deshacer (Ctrl+Z):** Permite deshacer la operación que está ejecutando.
2. **Rehacer (Ctrl+shift+Z):** Permite rehacer la operación que está ejecutando.
3. **Cortar (Ctrl+X):** Permite cortar la operación.
4. **Copiar (Ctrl+C):** Permite copiar la operación.
5. **Pegar (Ctrl+V):** Permite pegar la operación.
6. **Borrar Campo (Ctrl+P):** Permite borrar campos de la operación.

Observe la siguiente [Figura 71: Menú \(Editar\)](#).

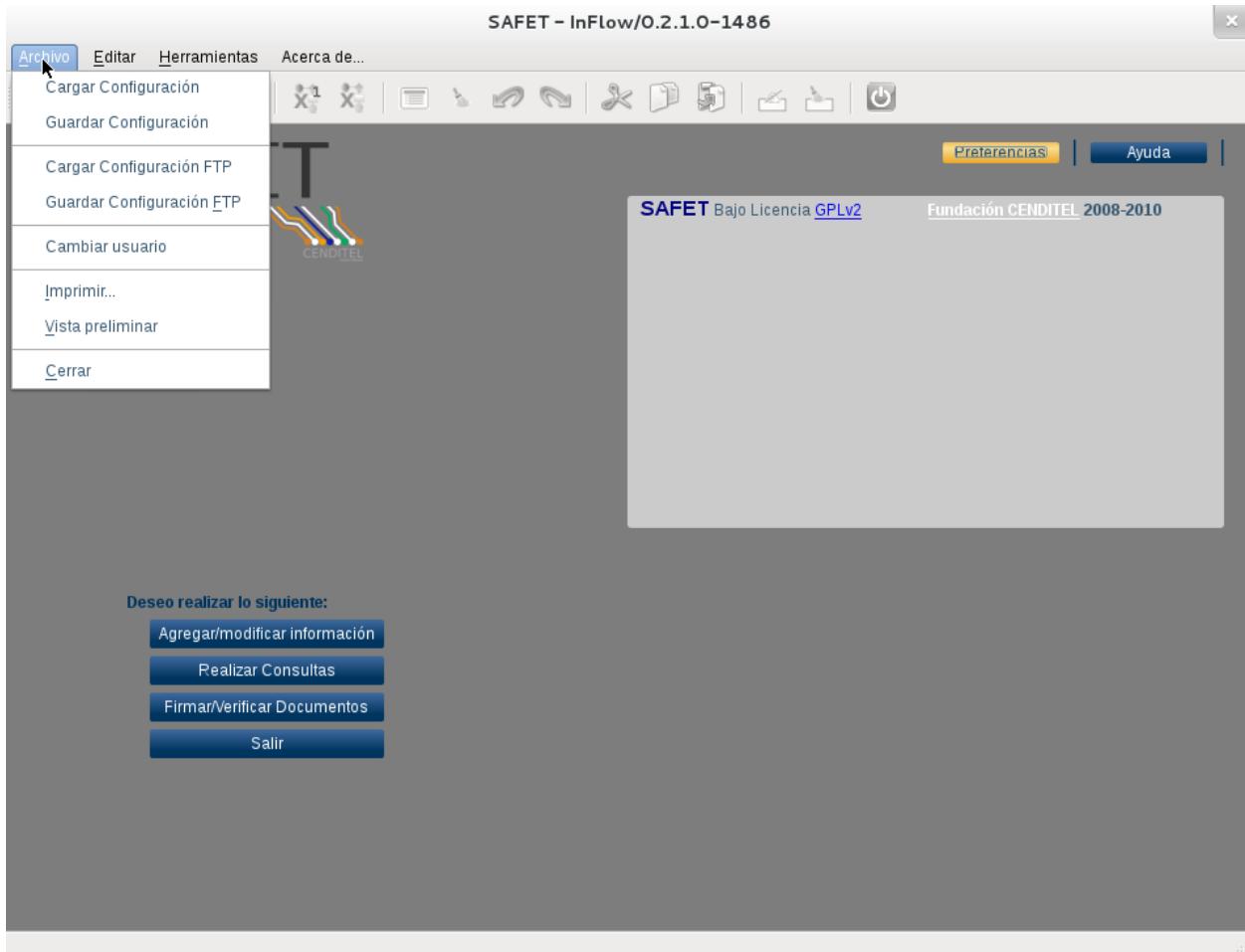


Figura 7.2: Figura 70: Menú (Archivo).

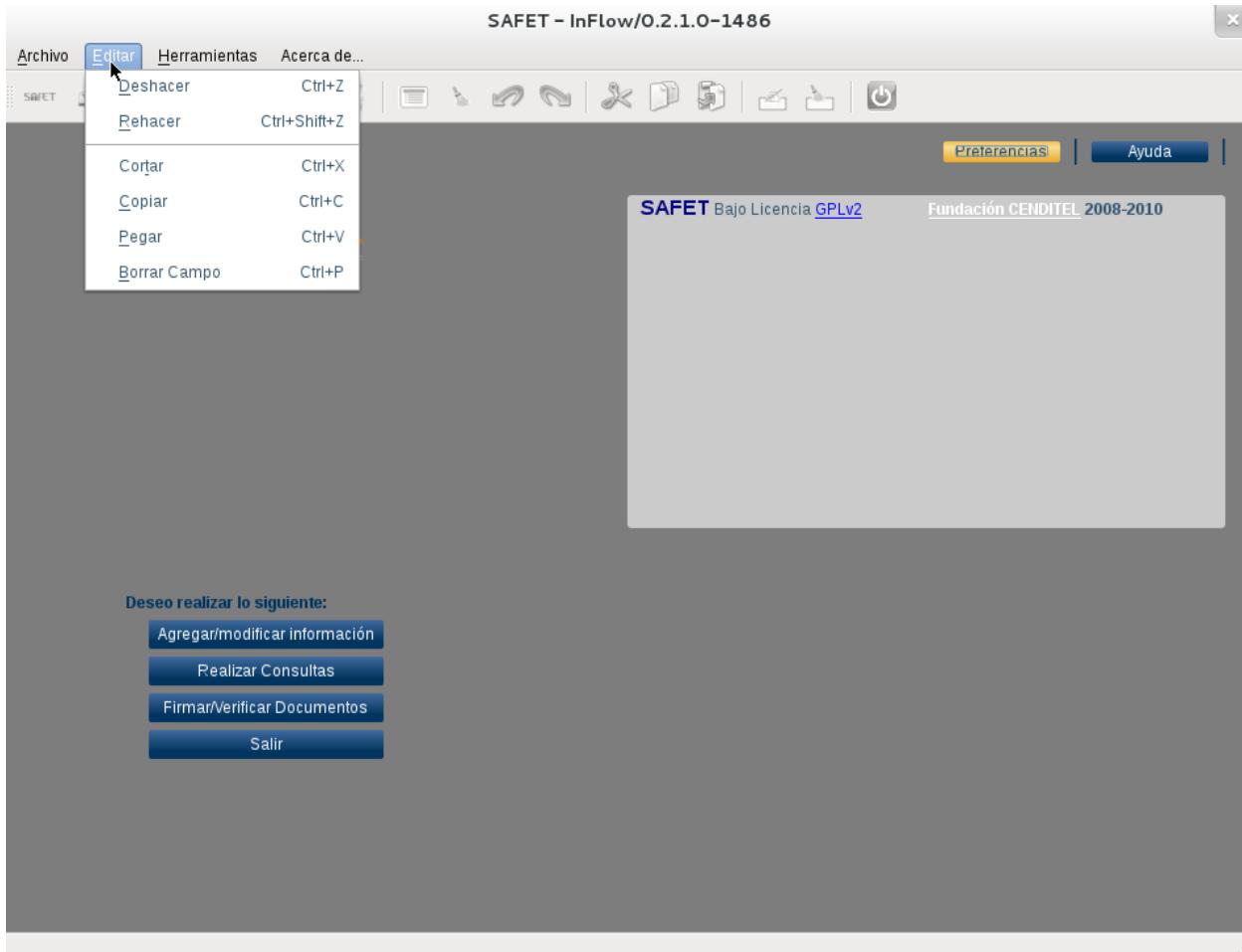


Figura 7.3: **Figura 71: Menú (Editar).**

### 3.- Botón (Herramientas)

En el botón (Herramientas) tenemos las siguientes opciones:

1. Opciones de widgets
2. Fuente de datos
3. Incluir documento firmado
4. Guardar/Enviar Documento firmado
5. Guardar gráfico seleccionado
6. Restaurar gráficos
7. Comparar gráficos

Observe la siguiente *Figura 72: Menú (Herramientas)*

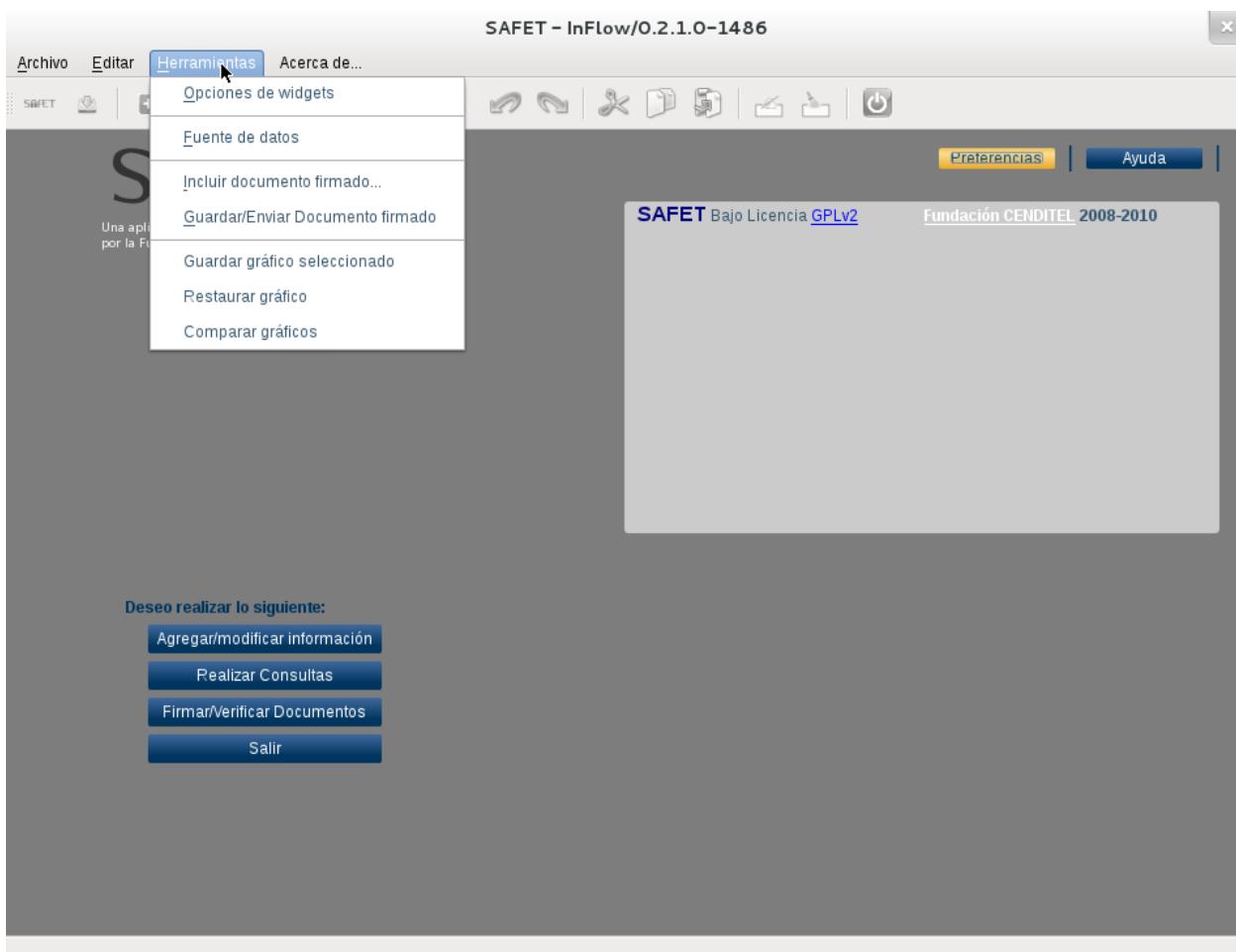


Figura 7.4: Figura 72: Menú (Herramientas)

### 4.- Botón (Acerca de)

En el botón (Acerca de) tenemos la siguientes opciones:

1. Ayuda
2. Acerca de...
3. Acerca de Qt

Observe la siguiente *Figura 73: Menú (Acerca de)*

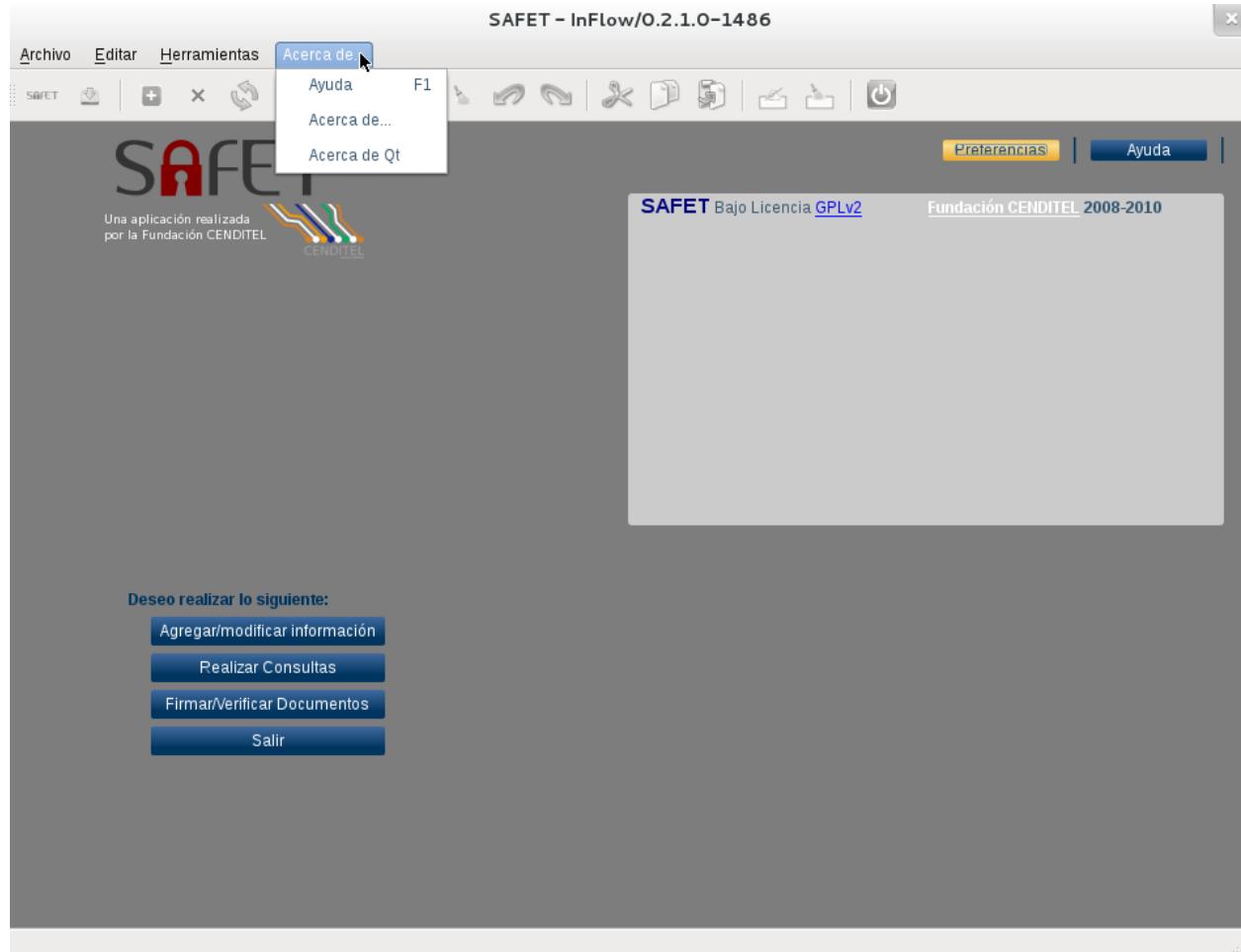


Figura 7.5: **Figura 73: Menú (Acerca de)**

### 7.1.3 C.- Opciones principales

En las opciones principales tenemos lo siguientes:

1. **Agregar/modificar información:** Permite entrar a las operaciones del formularios.
2. **Realizar consultar:** Permite entrar a las operaciones del consulta.

3. **Firmar/verificar Documento:** Permite entrar a las operaciones del Diseño.
4. **Salir:** Permite salir de la herramienta.

Observe la siguiente *Figura 74: Opciones principal*



Figura 7.6: **Figura 74: Opciones principal**

#### 7.1.4 D.- Barra de herramienta Inflow



Figura 7.7: **Figura 75: Herramientas Inflow.**

##### 1.- Botón (Principal):

Permite dirigirse a la entrada principal. Ver *Figura 76: Principal*



Figura 7.8: **Figura 76: Principal**

##### 2.- Botón (Enviar):

Su función es enviar el formulario de la operación o también usando la tecla (Ctrl+S). Ver *Figura 77: Botón (Enviar)*



Figura 7.9: **Figura 77: Botón (Enviar)**

### 3.- Botón (Aregar nodo):

Agregar nodos al gráfico seleccionado de flujo de trabajo. Ver *Figura 78: Botón (Aregar nodo)*



Figura 7.10: **Figura 78: Botón (Aregar nodo)**

### 4.- Botón (Eliminar nodo):

Eliminar nodos al gráfico seleccionado de flujo de trabajo. Ver *Figura 79: Botón (Eliminar nodo)*



Figura 7.11: **Figura 79: Botón (Eliminar nodo)**

### 5.- Botón (Actualizar):

Permite actualizar la consulta del último gráfico. Ver *Figura 80: Botón (Actualizar)*

### 6.- Botón (Borrar gráfico):

Permite borrar gráficos de flujo de trabajo seleccionado del cuadro actual. Ver *Figura 81: Botón (Borrar un solo gráfico)*

### 7.- Botón (Borrar gráficos):

Permite borrar todos los gráficos de flujo de trabajo del cuadro actual. Ver *Figura 82: Botón (Borrar todos los gráficos)*



Figura 7.12: **Figura 80: Botón (Actualizar)**



Figura 7.13: **Figura 81: Botón (Borrar un solo gráfico)**

#### 8.- Botón (Panel):

Permite ir al panel de edición actual o también con la tecla (Ctrl+W). Ver *Figura 83: Botón (Panel)*

#### 9.- Botón (Borrar cuadro):

Permite borrar el cuadro de edición actual o también con la tecla (Ctrl+D). Ver *Figura 84: Botón (Borrar cuadro)*

#### 10.- Botón (Deshacer):

Permite deshacer la operación que se está ejecutando o también con la tecla (Ctrl+Z). Ver *Figura 85: Botón (Deshacer)*

#### 11.- Botón (Rehacer):

Permite rehacer la operación que se está ejecutando o también con la tecla (Ctrl+shift+Z). Ver *Figura 86: Botón (Deshacer)*



Figura 7.14: **Figura 82: Botón (Borrar todos los gráficos)**



Figura 7.15: **Figura 83: Botón (Panel)**



Figura 7.16: **Figura 84: Botón (Borrar cuadro)**

#### 12.- Botón (Cortar):

Permite cortar la operación que se está ejecutando o también con la tecla (Ctrl+X). Ver *Figura 87: Botón (Cortar)*

#### 13.- Botón (Copiar):

Permite copiar la operación que se está ejecutando o también con la tecla (Ctrl+C). Ver *Figura 88: Botón (Copiar)*

#### 14.- Botón (Pegar):

Permite pegar la operación que se está ejecutando o también con la tecla (Ctrl+V). Ver *Figura 89: Botón (Pegar)*

#### 15.- Botón (Modificar campo):

Permite modificar campos de la operación que se está ejecutando o también con la tecla (Ctrl+M). Ver *Figura 90: Botón (Modificar campo)*



Figura 7.17: **Figura 85: Botón (Deshacer)**



Figura 7.18: **Figura 86: Botón (Deshacer)**



Figura 7.19: **Figura 87: Botón (Cortar)**



Figura 7.20: **Figura 88: Botón (Copiar)**



Figura 7.21: **Figura 89: Botón (Pegar)**



Figura 7.22: **Figura 90: Botón (Modificar campo)**

### 16.- Botón (Borrar campo):

Permite borrar campos de la operación que se está ejecutando o también con la tecla (Ctrl+P). Ver *Figura 91: Botón (Borrar campo)*



Figura 7.23: **Figura 91: Botón (Borrar campo)**

### 17.- Botón (Cerrar o salir):

Permite cerrar o salir de la herramienta. Ver *Figura 92: Botón (Cerrar o salir)*



Figura 7.24: **Figura 92: Botón (Cerrar o salir)**

### 7.1.5 E.- Menú de operaciones



**Figura 93: Menú de Operaciones**

- 1.- **Botón (Formulario):** Permite ejecutar las operaciones **CRUD+**.
- 2.- **Botón (Consulta):** Permite ejecutar consultar de datos y gráficos de flujo de trabajos.
- 3.- **Botón (Diseño):** Permite diseñar los modelos de flujos de trabajo.
- 4.- **Botón (Reporte):** Permite ver los reportes de datos.
- 5.- **Botón (Flujo):** Permite ver los reportes de gráficos de flujo de trabajo.
- 6.- **Botón (Configuración):** Permite realizar configuraciones de todas las operaciones generales.

**7.- Botón (Salida):** Permite mostrar la información de la operación que se está ejecutando.

**8.- Botón (Usuario/Roles):** Permite realizar las operaciones de usuario y roles.

### 7.1.6 F.- Opciones de Campos de la Operación



Figura 7.25: **Figura 94: Opciones de campos de la operación**

**1.- Botón (Examinar):** Permite cargar el archivo XML del directorio (**/home/usuario/.safet/flowfile/**) en el campo de la operación. Ver [Figura 95: Botón \(Examinar\)](#)



Figura 7.26: **Figura 95: Botón (Examinar)**

**2.- Botón (Cancelar):** Permite cancelar el campo de la operación. Ver [Figura 96: Botón \(Cancelar\)](#)



Figura 7.27: **Figura 96: Botón (Cancelar)**

**3.- Botón (Finalizar):** Permite la finalizar el campo de la operación. Ver [Figura 97: Botón \(Cancelar\)](#)

### 7.1.7 G.- Botones de Ejecución

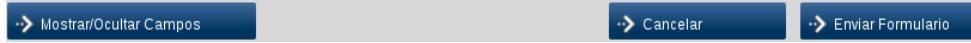


Figura 98: **Botones de Ejecución**

**1.- Botón (Mostrar/Ocultar Campos):** Permite mostrar o ocultar las operaciones y campos a ejecutar.

**2.- Botón (Cancelar):** Permite cancelar la operación.

**3.- Botón (Enviar Formulario):** Permite enviar los datos del formulario o de la operación.



Figura 7.28: **Figura 97: Botón (Cancelar)**

## 7.2 Ejecución de operaciones CRUD+

En este tutorial explicaremos el **ejemplo de inventario** utilizando las operaciones **CRUD** de manera mas gráfica, para ello debemos tener instalado inflow, sino damos un click al siguiente enlace. Instalación de la interfaz gráfica (inflow).

A continuación seguimos los siguientes pasos para abrir inflow y obtener las acciones CRUD:

### 7.2.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 99: Autenticación de Inflow*.

#### 3° TERCER PASO

- Damos click a la primera opción (**Agregar/modificar información**), como se muestra en la siguiente *Figura 100: Agregar/modificar información*

#### 4° CUARTO PASO

- Damos click a la opción (**Mostrar/Ocultar Campos**), la cual se nos mostrara las acciones **CRUD**, como se muestra en la siguiente *Figura 101: Acciones CRUD*



Figura 7.29: Figura 99: Autenticación de Inflow.

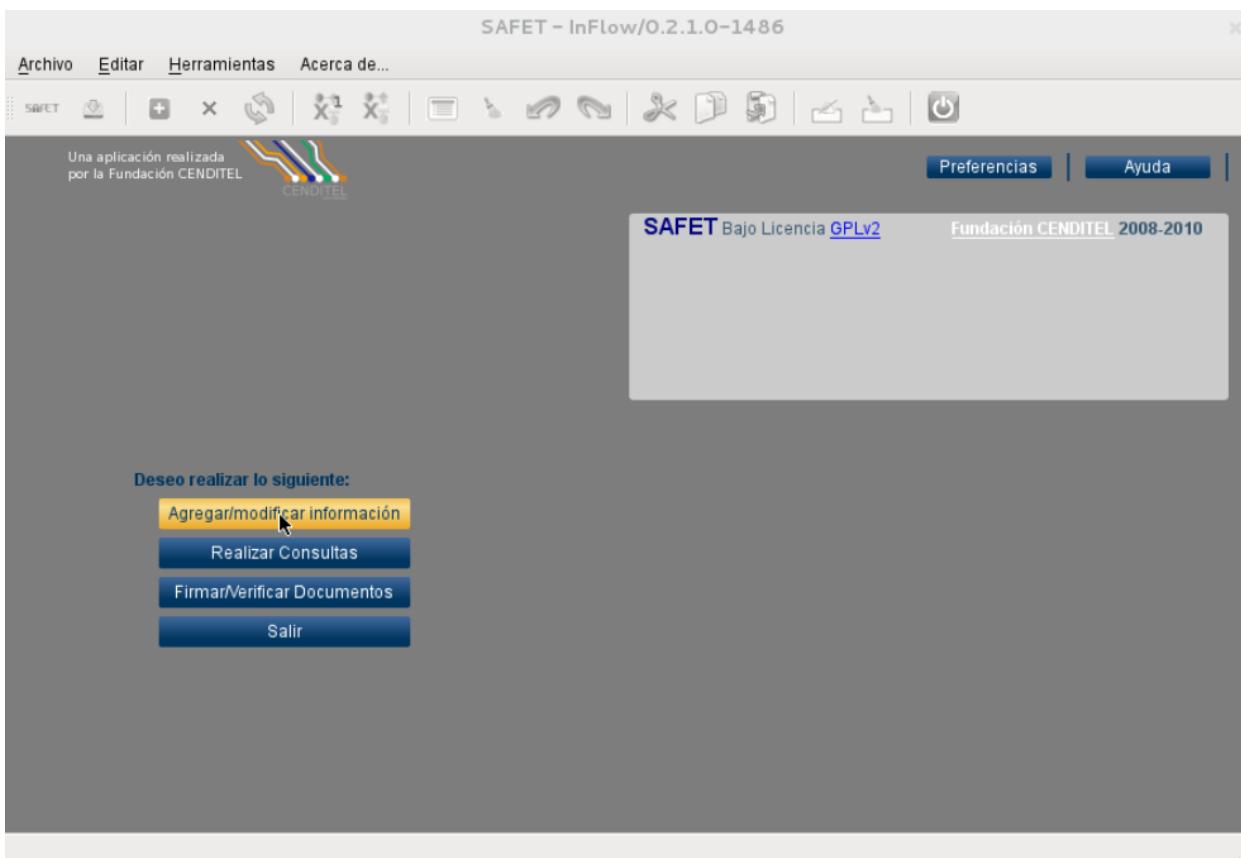


Figura 7.30: Figura 100: Agregar/modificar información

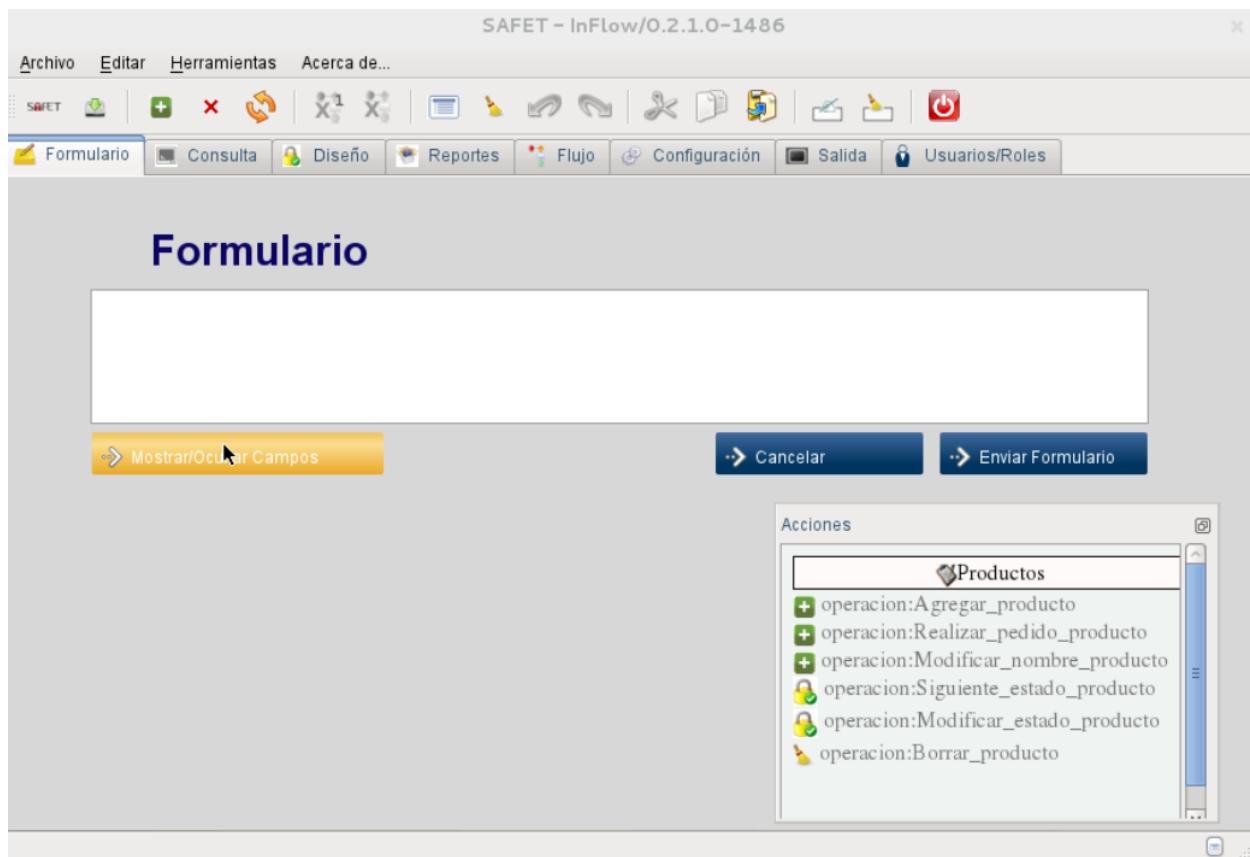


Figura 7.31: **Figura 101: Acciones CRUD**

## 7.2.2 1° PRIMERA OPERACIÓN CRUD+ (Agregar o registrar producto)

### 1° PRIMER PASO

- Damos click a la operación (**operacion:Agregar\_producto**), como se muestra en la siguiente *Figura 102: Acciones CRUD*

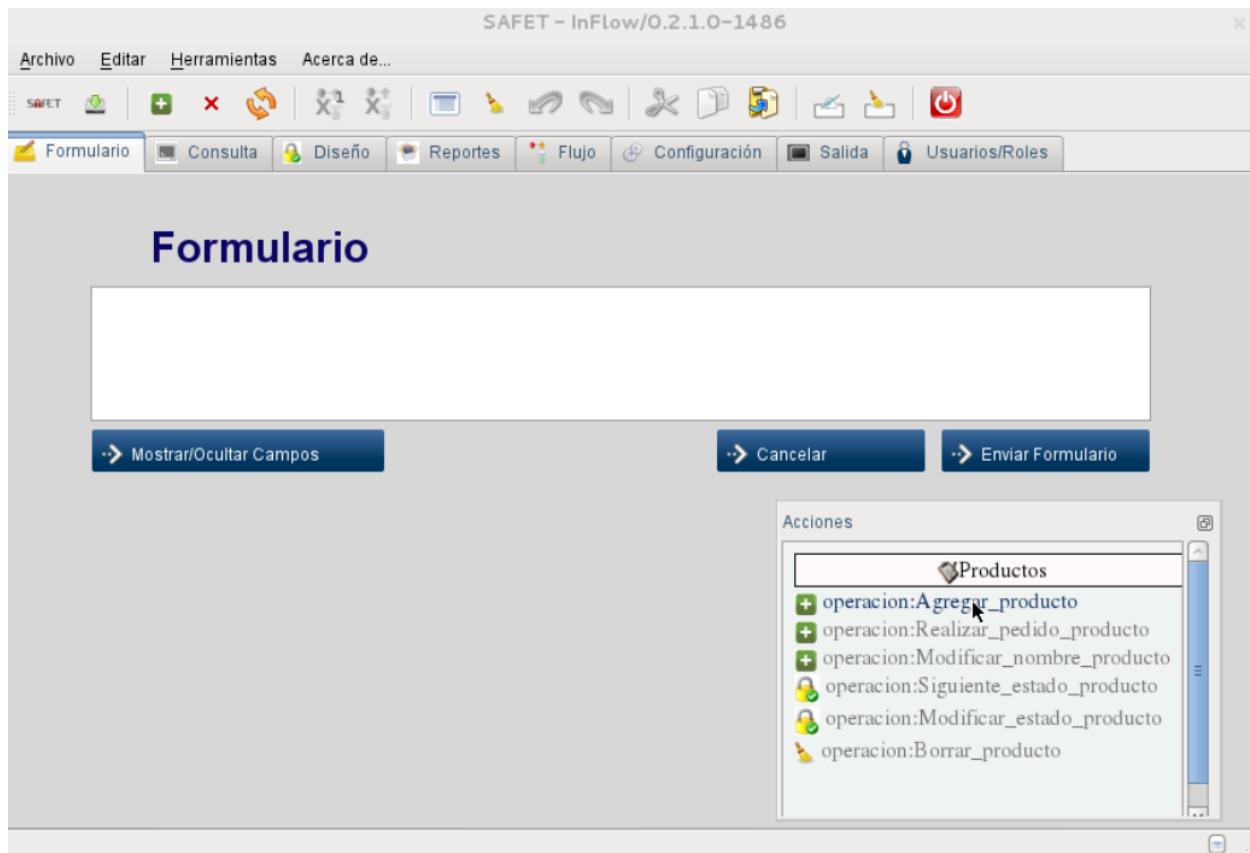


Figura 7.32: **Figura 102: Acciones CRUD**

### 2° SEGUNDO PASO

- En esta (**operacion:Agregar\_producto**) tenemos un solo campo llamado (**Nombre**), la cual damos un click a ese campo (**Nombre**), como se muestra en la siguiente *Figura 103: Campo (Nombre)*

---

**Nota:** Al darle click campo (**Nombre**), se nos mostrara en formulario junto con la (**operacion:agregar\_producto**), como se muestra en la siguiente *Figura 104: Campo en el formulario*

---

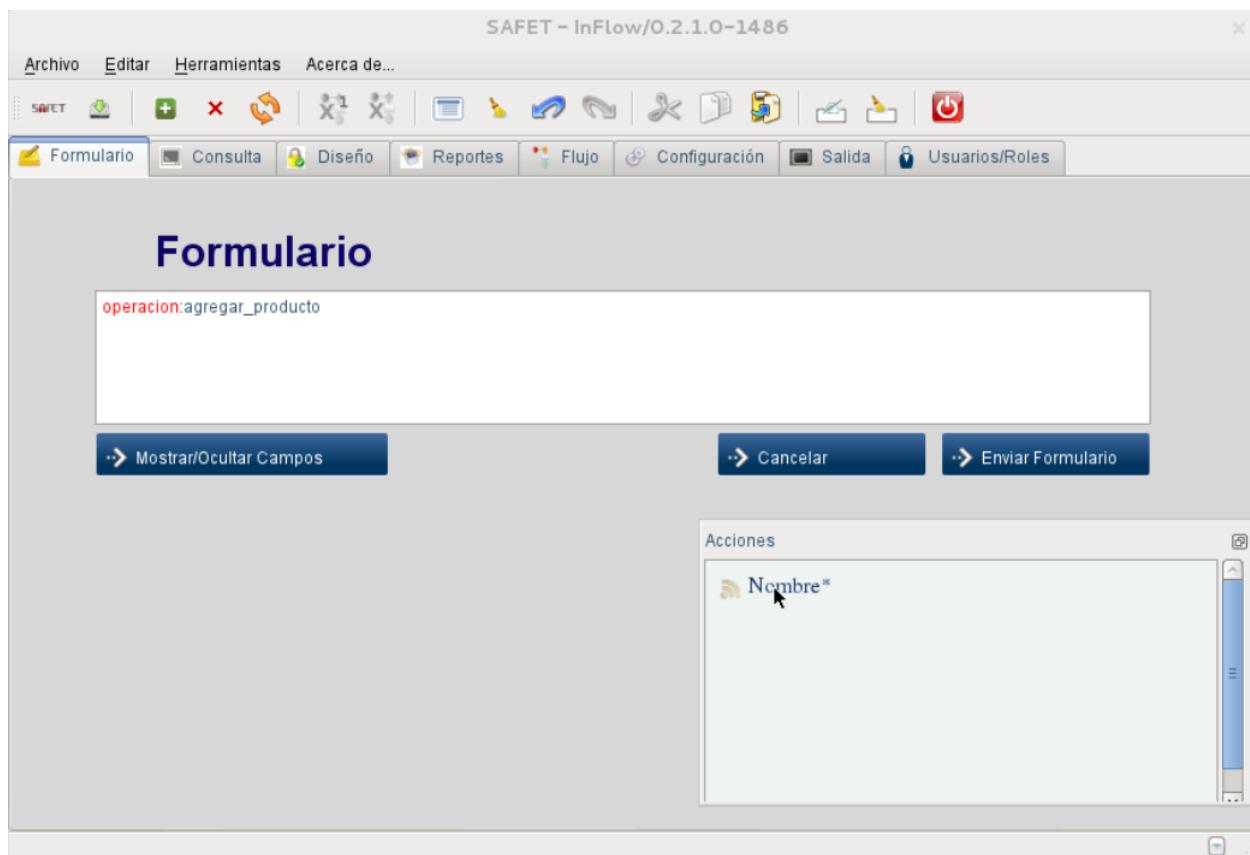


Figura 7.33: **Figura 103: Campo (Nombre)**

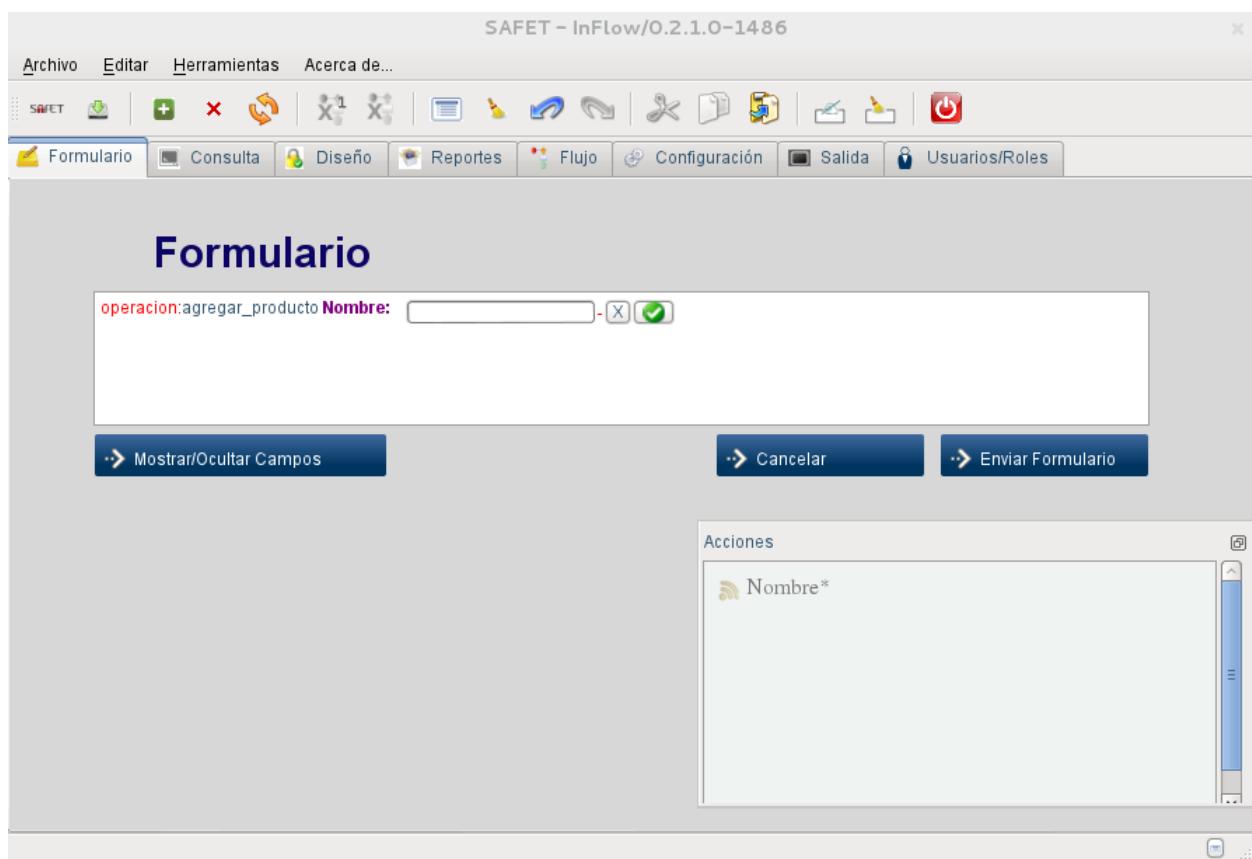


Figura 7.34: Figura 104: Campo en el formulario

### 3° TERCER PASO

- Llenamos el campo (**Nombre**), es decir colocamos el nombre del producto a registrar, como se muestra en la siguiente *Figura 105: Nombre del producto*

The screenshot shows the SAFET - InFlow application interface. At the top, there's a menu bar with Archivo, Editar, Herramientas, and Acerca de... followed by a toolbar with various icons. Below that is a navigation bar with links for Formulario, Consulta, Diseño, Reportes, Flujo, Configuración, Salida\*, and Usuarios/Roles. The main window title is "Formulario". Inside, there's a text input field with the placeholder "operacion:agregar\_producto Nombre:" and the value "Acetaminofén". To the right of the input field is a green checkmark icon with a plus sign. Below the input field are three buttons: "Mostrar/Ocultar Campos", "Cancelar", and "Enviar Formulario". On the right side of the screen, there's a sidebar titled "Acciones" which lists the "Nombre\*" field.

Figura 7.35: **Figura 105: Nombre del producto**

### 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo a registrar, como se muestra en la siguiente *Figura 106: Formulario lleno*

### 5° QUINTO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**La operación fue exitosa....ok!**), como se muestra en la siguiente *Figura 107: Resultado de la operación*

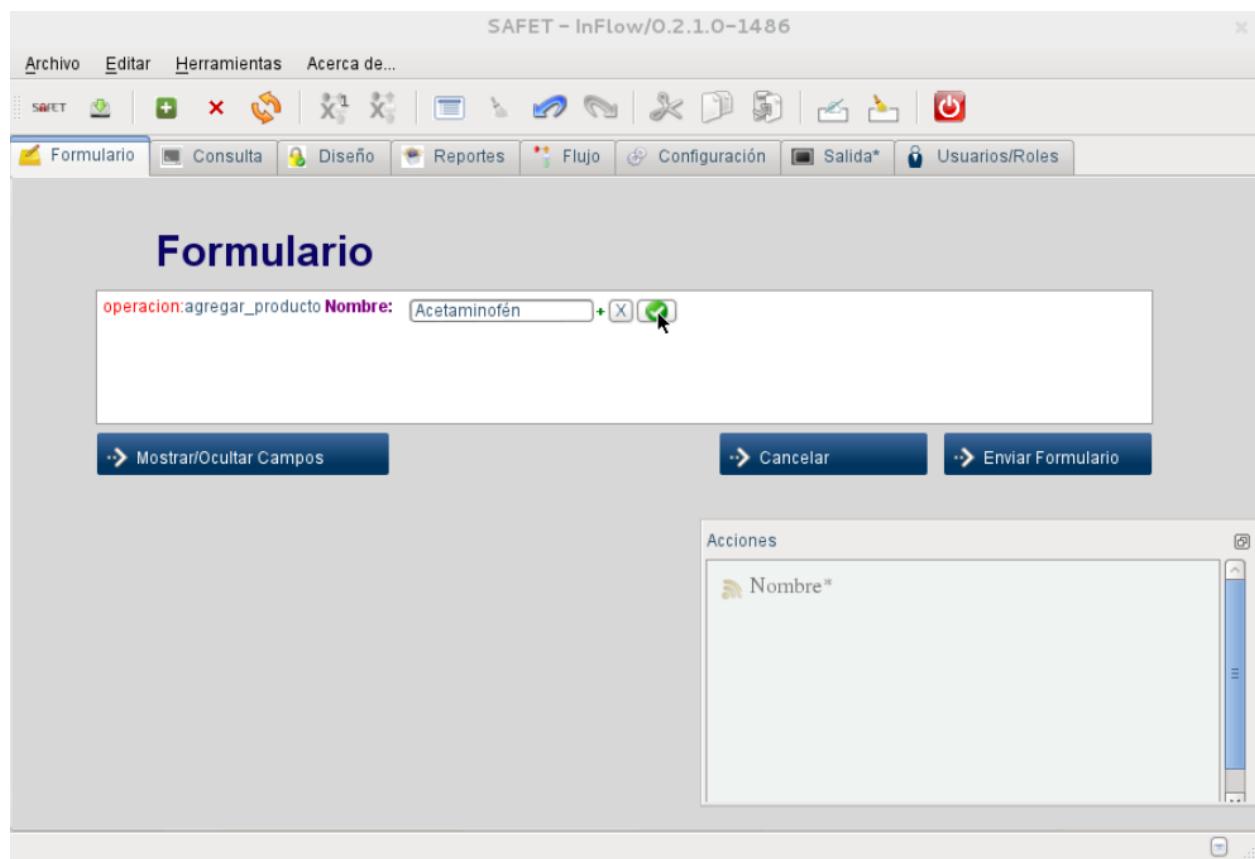


Figura 7.36: **Figura 106: Formulario lleno**

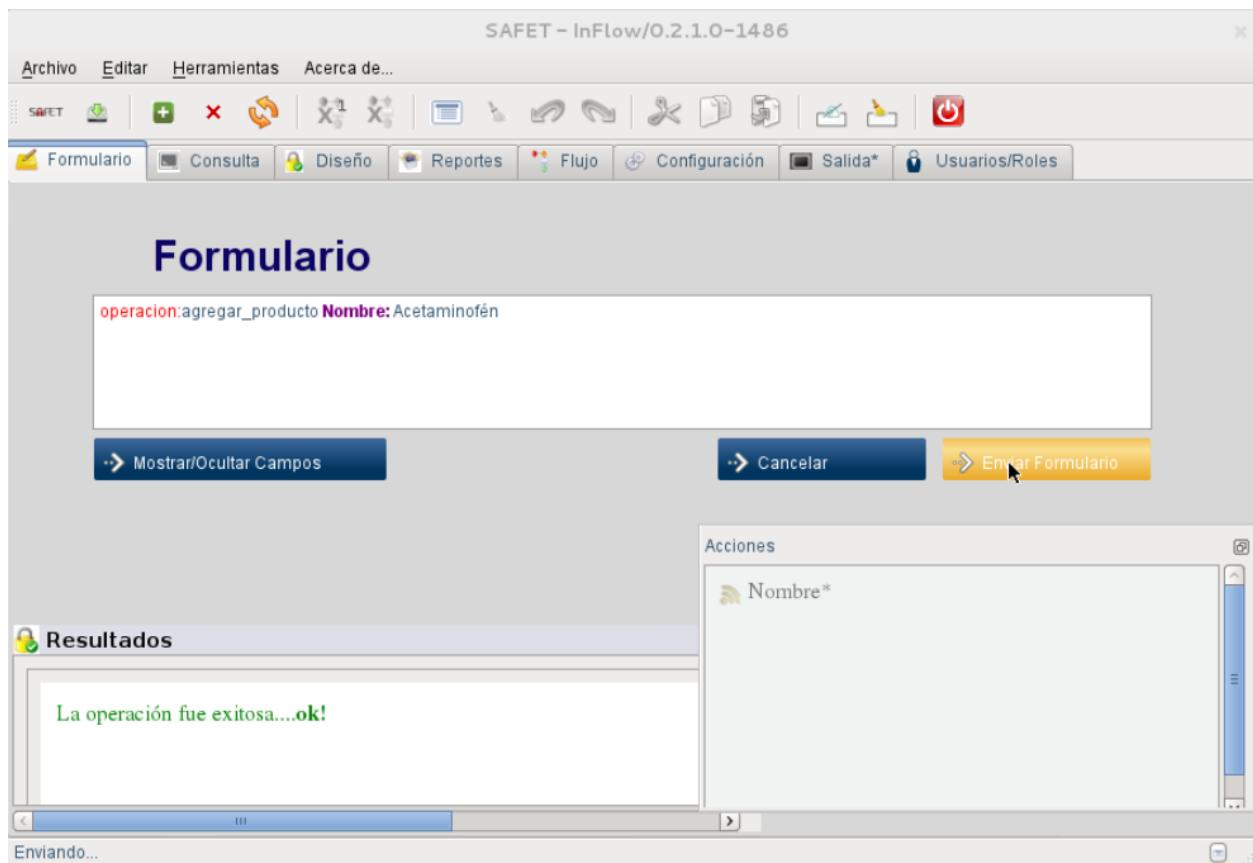


Figura 7.37: Figura 107: Resultado de la operación

## 6° SEXTO PASO

- Para pasar a la siguiente operación damos un click al botón cancelar como se muestra en la siguiente *Figura 108: Siguiete operación*

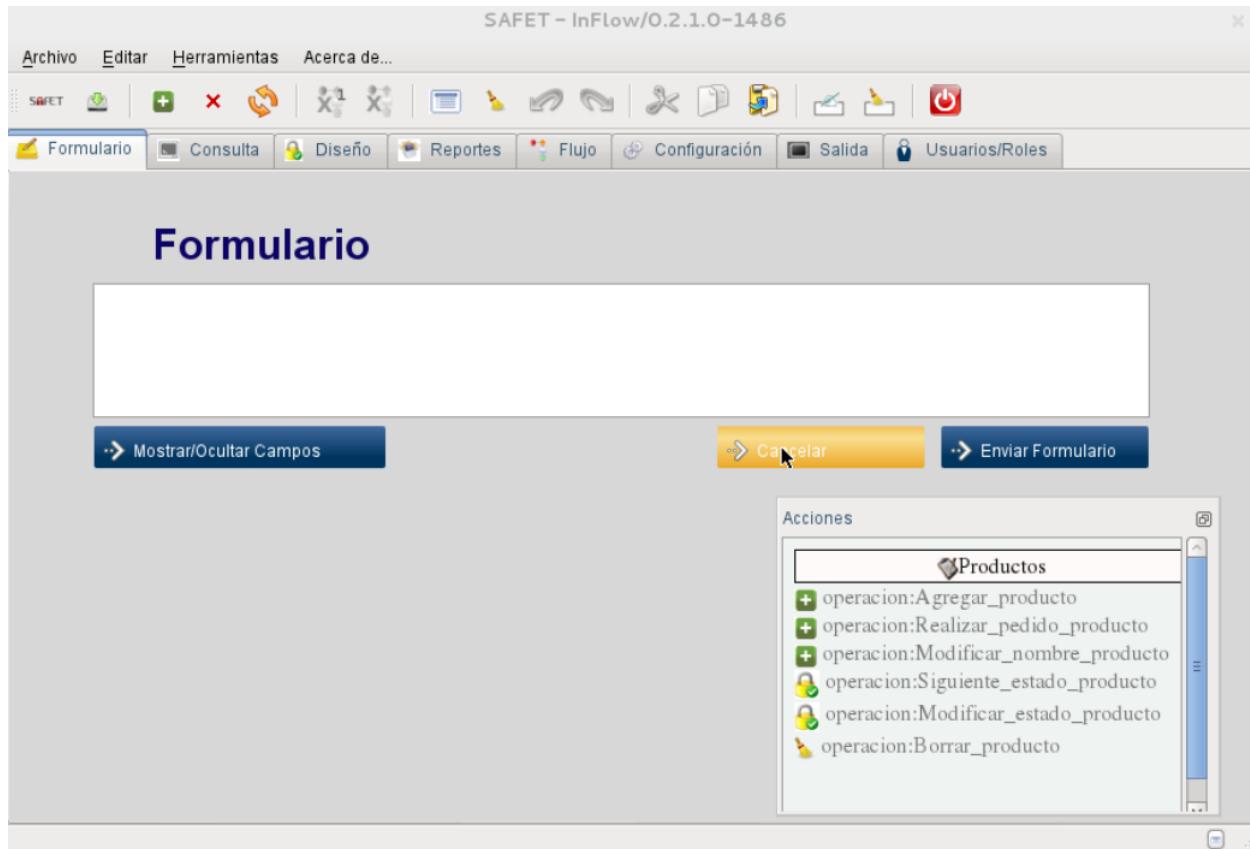


Figura 7.38: **Figura 108: Siguiete operación**

### 7.2.3 2° SEGUNDA OPERACIÓN CRUD+ (Realizar pedido de un producto)

#### 1° PRIMER PASO

- Damos click a la operación (**operacion:Realizar\_pedido\_producto**), como se muestra en la siguiente *Figura 109: realizar\_pedido*

#### 2° SEGUNDO PASO

- En esta (**operacion:agregar\_producto**) tenemos dos campo llamado (**id**) y (**Cuantos**), la cual damos un click al primer campo (**id**) para seleccionar el productos a pedir, como se muestra en la siguiente *Figura 110: Campo (id)*

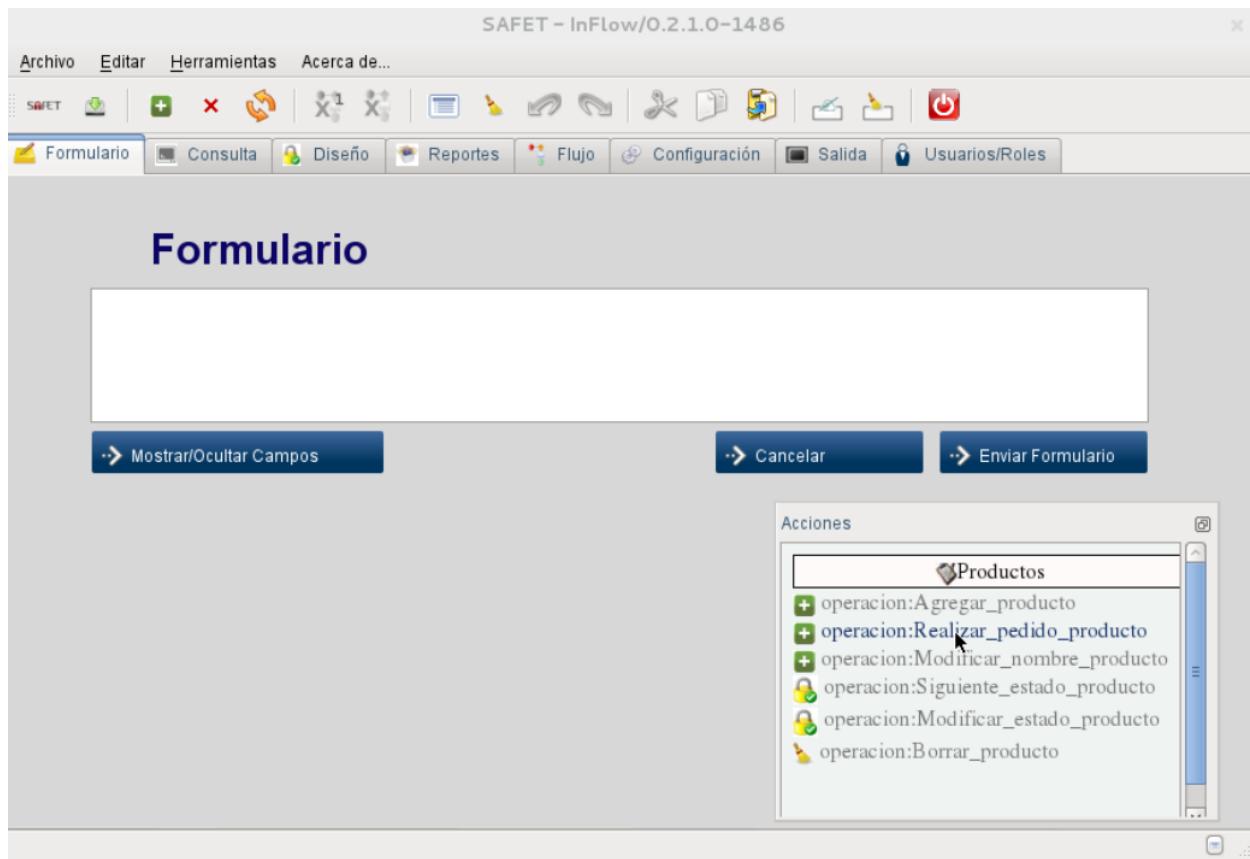


Figura 7.39: **Figura 109: realizar\_pedido**

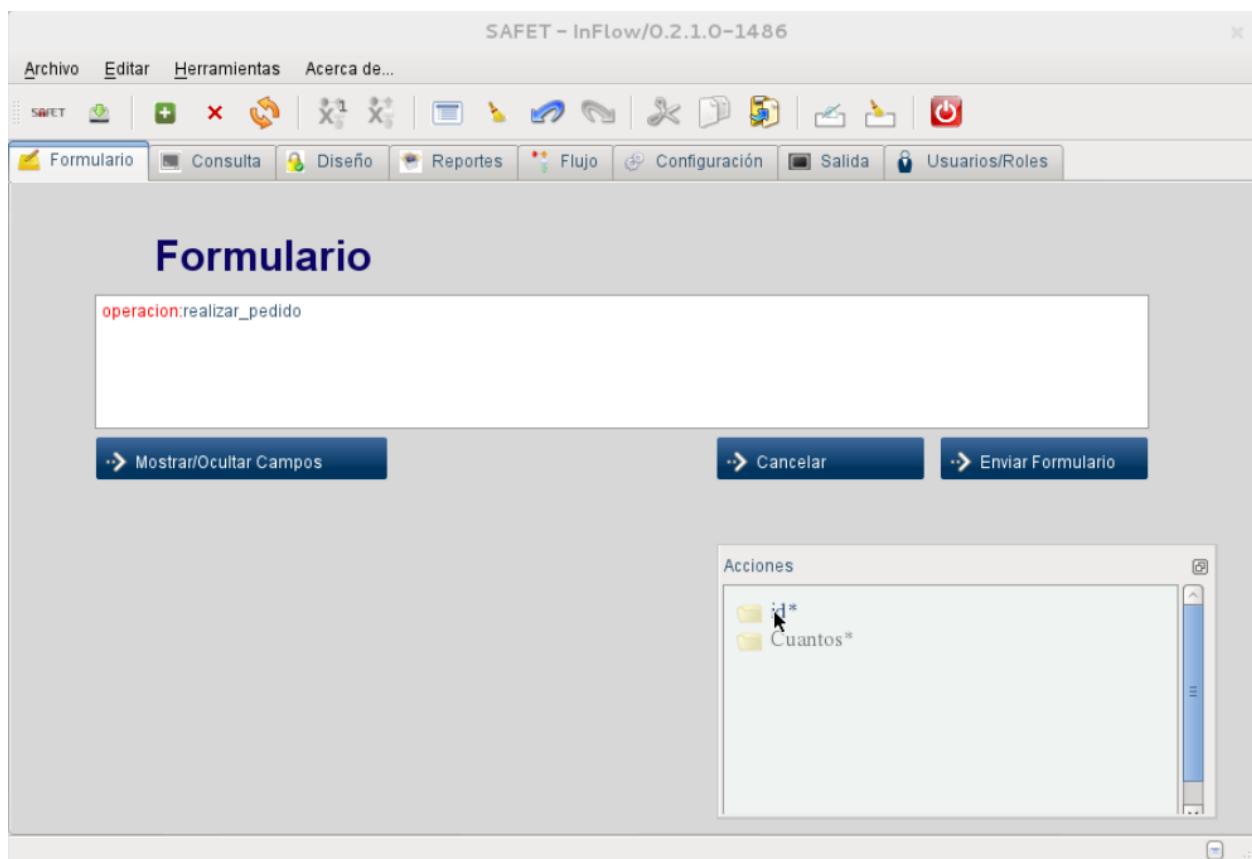


Figura 7.40: **Figura 110: Campo (id)**

### 3° TERCER PASO

- Seleccionamos el producto a pedir como se muestra en la siguiente *Figura 111: Producto a pedir*

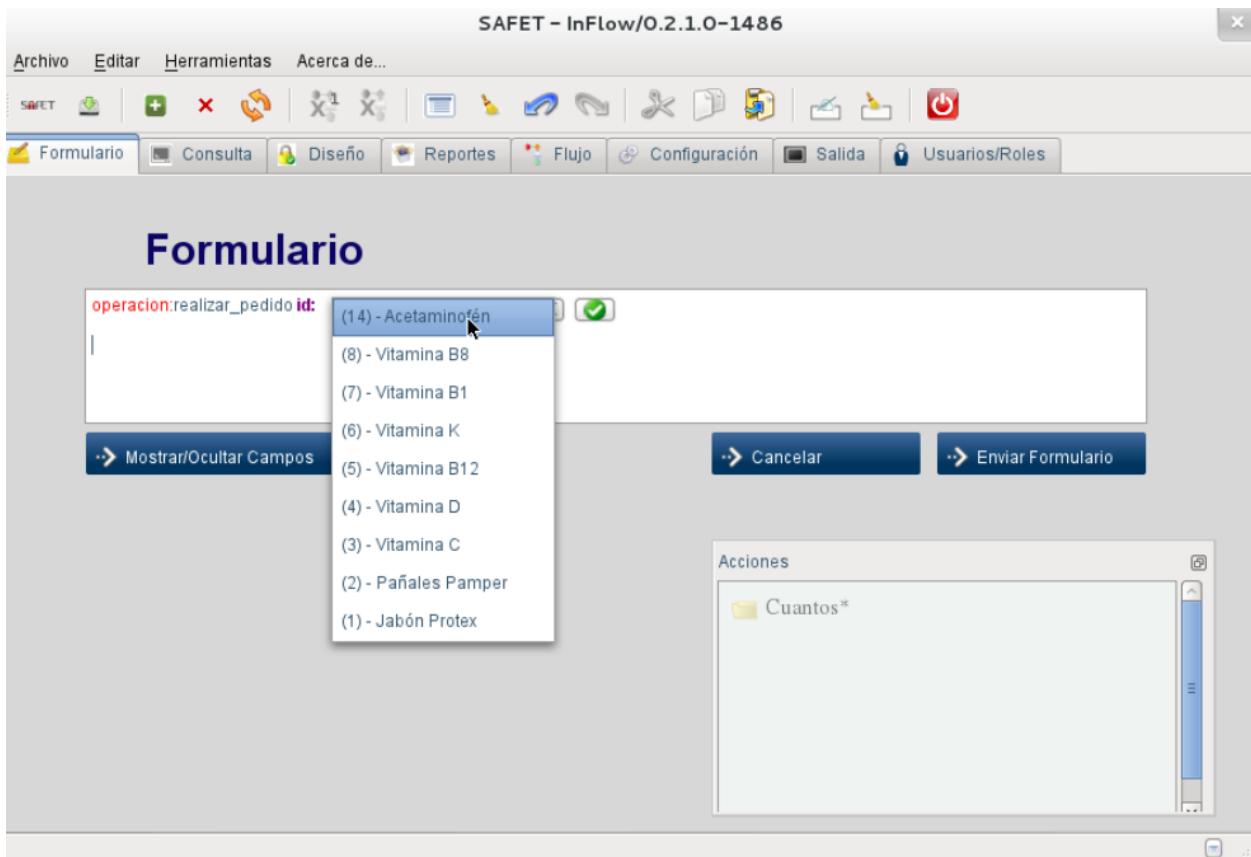


Figura 7.41: **Figura 111: Producto a pedir**

### 4° CUARTO PASO

- Damos un click al botón con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 112: Botón*

### 5° QUINTO PASO

- Damos un click al segundo campo (**Cuantos**) para indicarle el numero a pedir, como se muestra en la siguiente *Figura 113: Campo (Cuantos)*

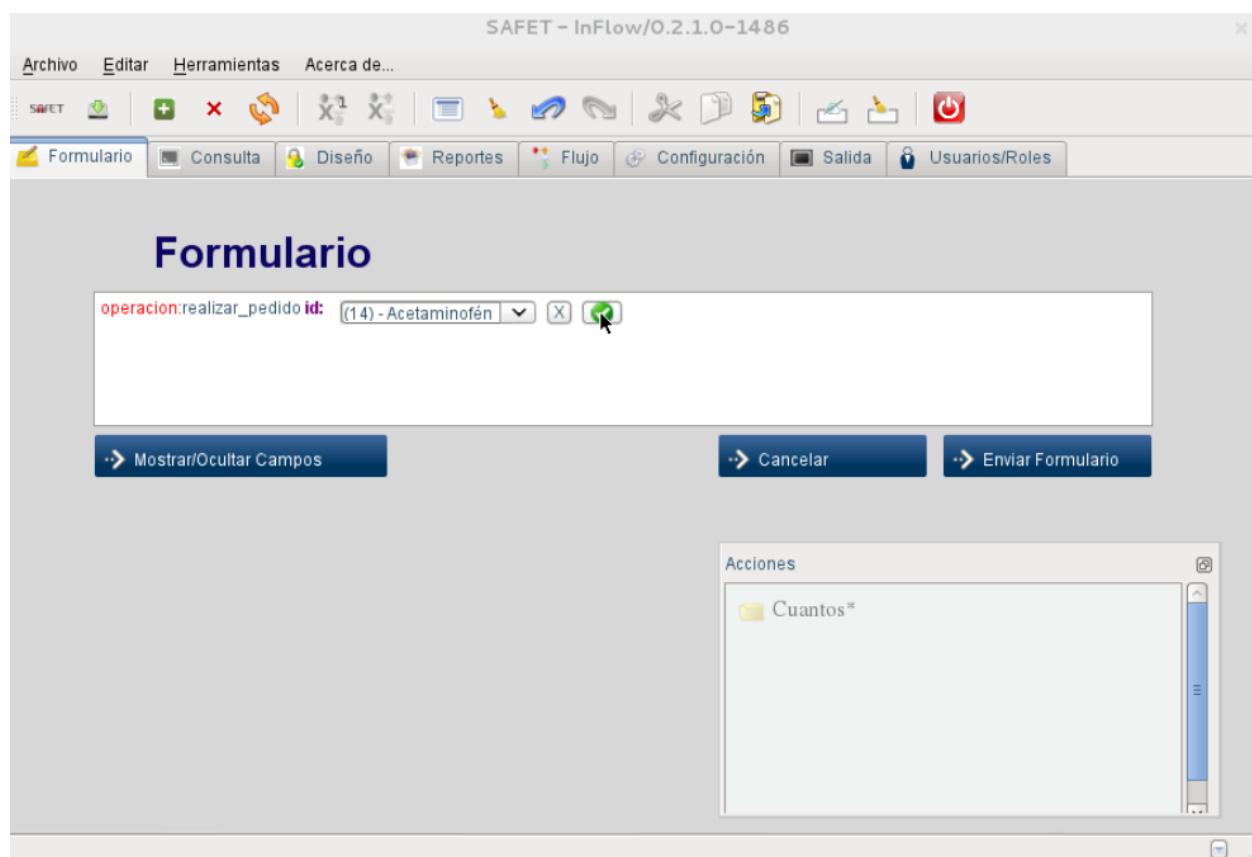


Figura 7.42: Figura 112: Botón

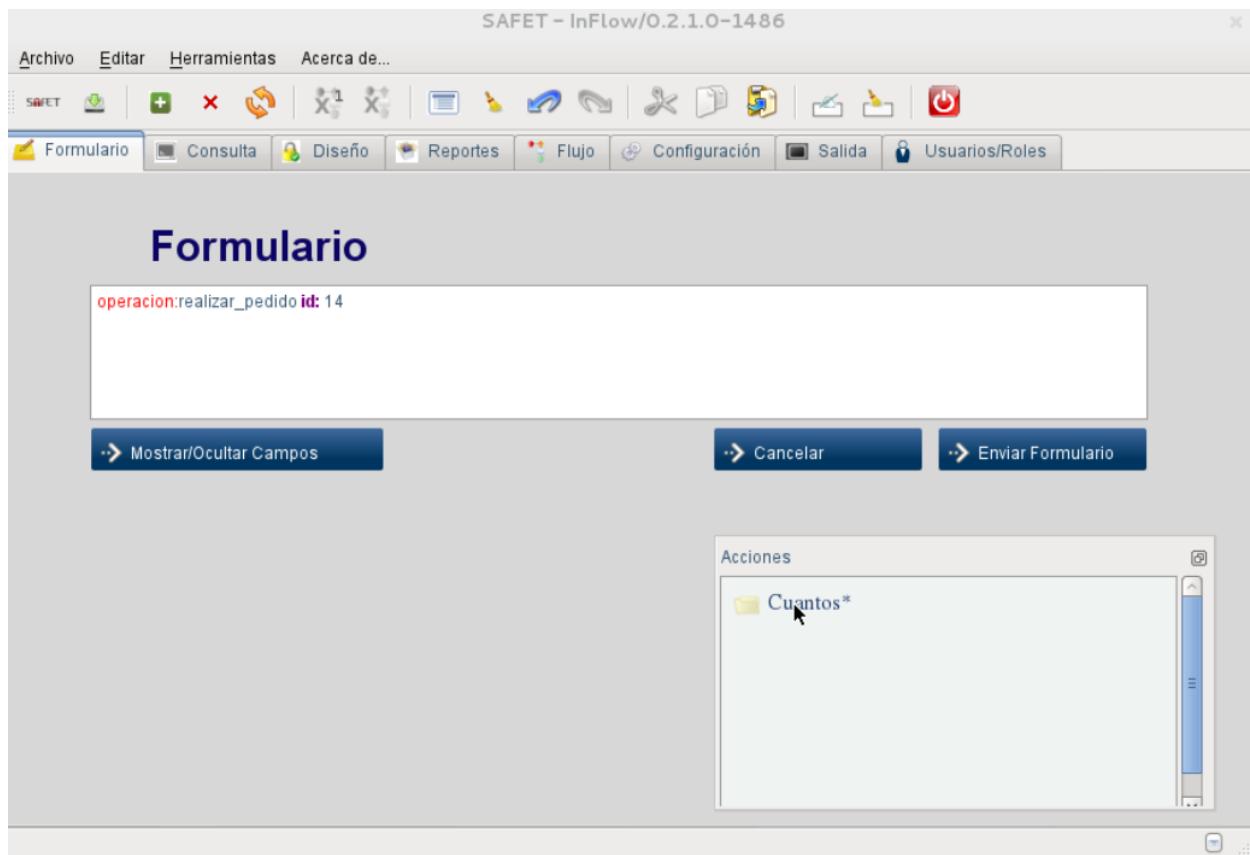


Figura 7.43: **Figura 113: Campo (Cuantos)**

## 6° SEXTO PASO

- Ingresamos el número a pedir, como se muestra en la siguiente *Figura 114: Número a pedir*

The screenshot shows the SAFET - InFlow application window. The title bar reads "SAFET - InFlow/0.2.1.0-1486". The menu bar includes "Archivo", "Editar", "Herramientas", and "Acerca de...". The toolbar contains various icons for file operations. The main menu bar has tabs: "Formulario", "Consulta", "Diseño", "Reportes", "Flujo", "Configuración", "Salida", and "Usuarios/Roles". The "Formulario" tab is selected. A sub-form titled "Formulario" is displayed. It contains a numeric input field with the value "220" and a green checkmark button. A tooltip below the field says "Campo Numérico. Escriba un número entero o decimal". At the bottom of the sub-form are three buttons: "Mostrar/Ocultar Campos", "Cancelar", and "Enviar Formulario". To the right of the sub-form, there is a sidebar titled "Acciones" with a scrollable list.

Figura 7.44: **Figura 114: Número a pedir**

## 7° SEPTIMO PASO

- Damos un click al botón con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 115: Botón*

## 8° OCTAVO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**La operación fue exitosa....ok!**), como se muestra en la siguiente *Figura 116: Resultado de la operación*

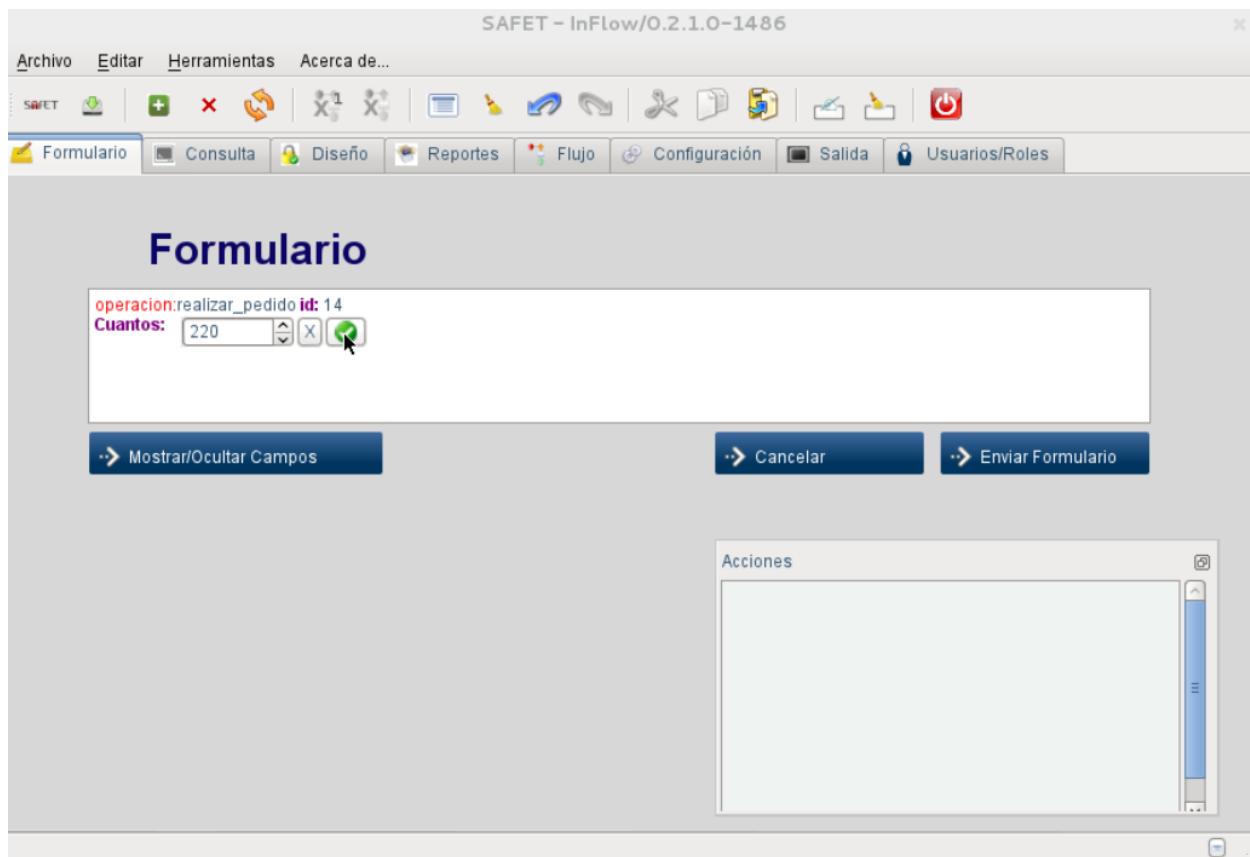


Figura 7.45: Figura 115: Botón

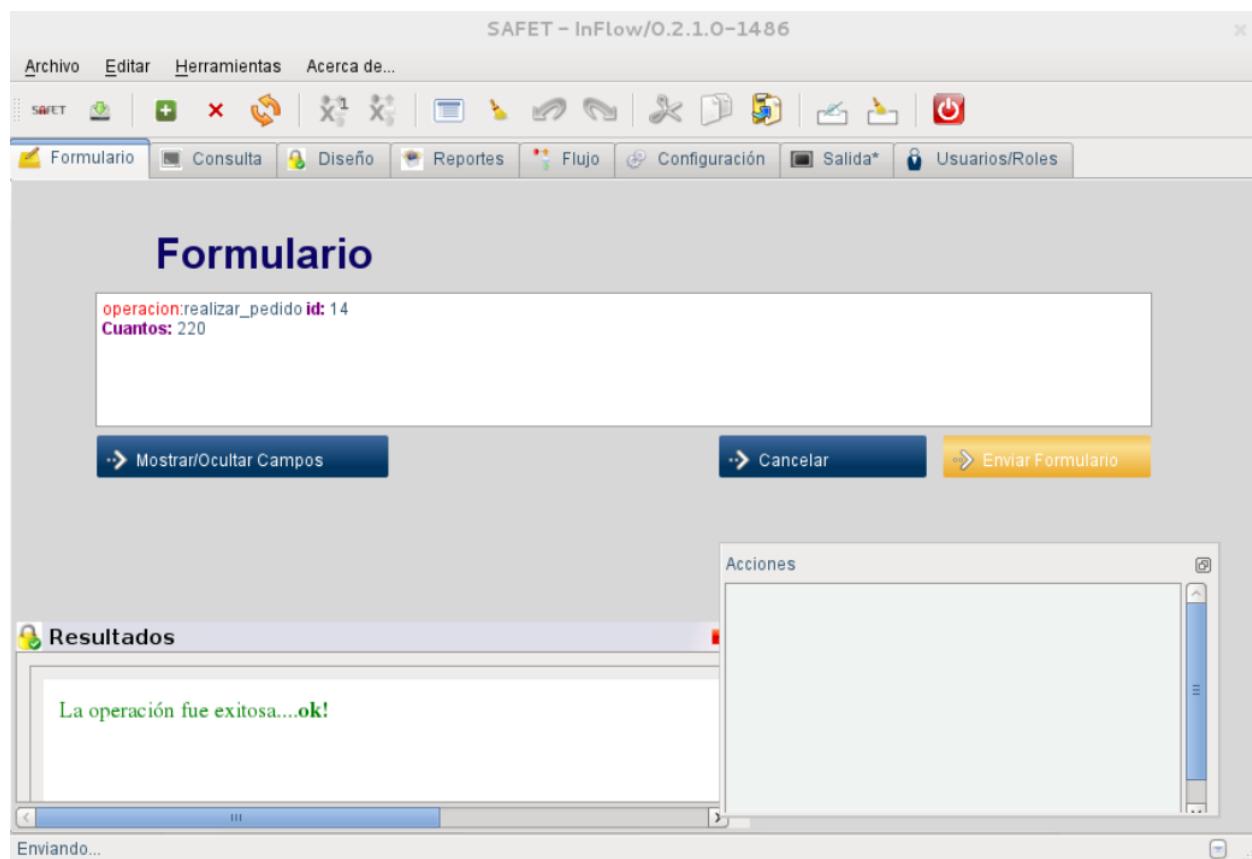


Figura 7.46: Figura 116: Resultado de la operación

### 9° NOVENO PASO

- Para pasar a la siguiente operación damos un click al botón **cancelar** como se muestra en la siguiente *Figura 117: Siguiente operación*

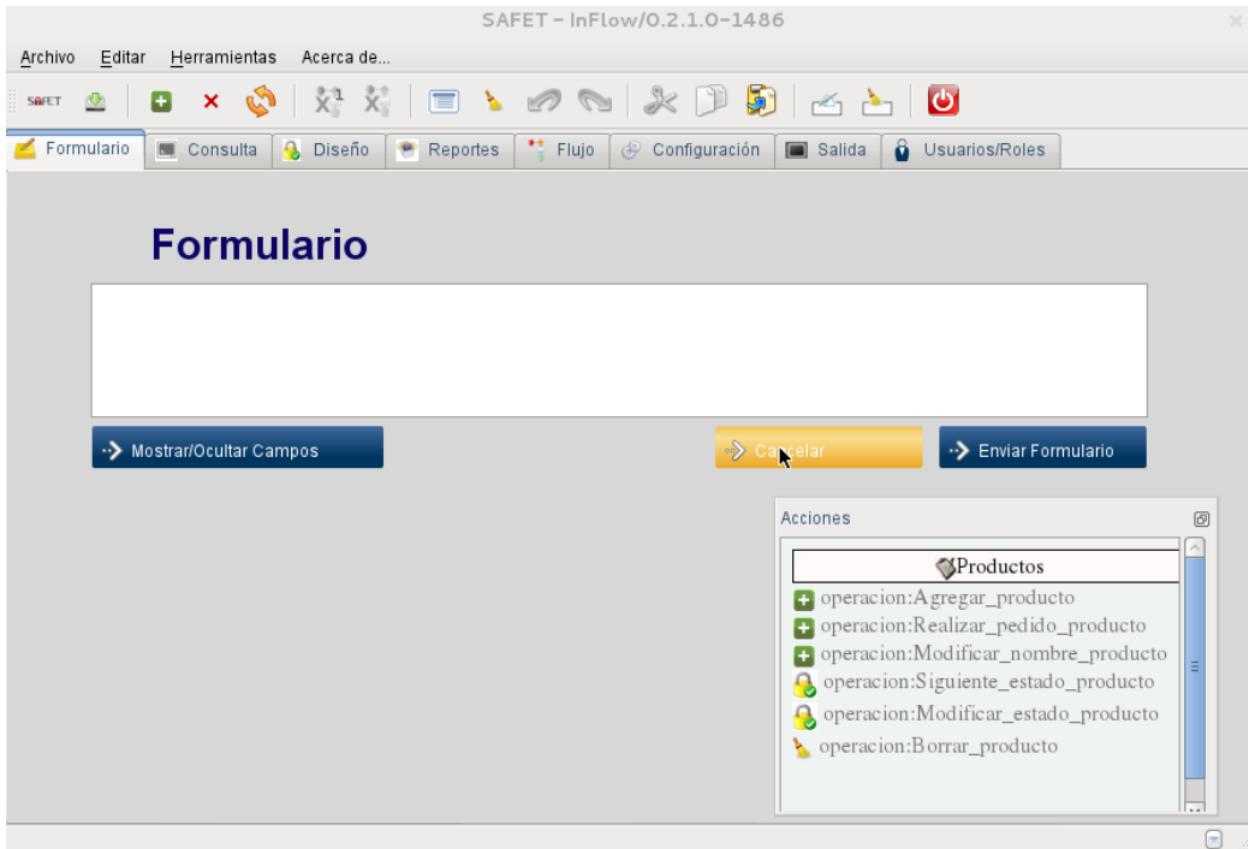


Figura 7.47: **Figura 117: Siguiente operación**

### 7.2.4 3° TERCERA OPERACIÓN CRUD+ (Modificar el nombre de un producto)

#### 1° PRIMER PASO

- Damos click a la operación (**operacion:Modificar\_nombre\_producto**), como se muestra en la siguiente *Figura 118: Modificar\_nombre\_producto*

#### 2° SEGUNDO PASO

- En esta (**operacion:Modificar\_nombre\_producto**) tenemos dos campo llamado (**id**) y (**Nombre**), la cual damos un click al primer campo (**id**) para seleccionar el productos a modificar, como se muestra en la siguiente *Figura 119: Campo (id)*

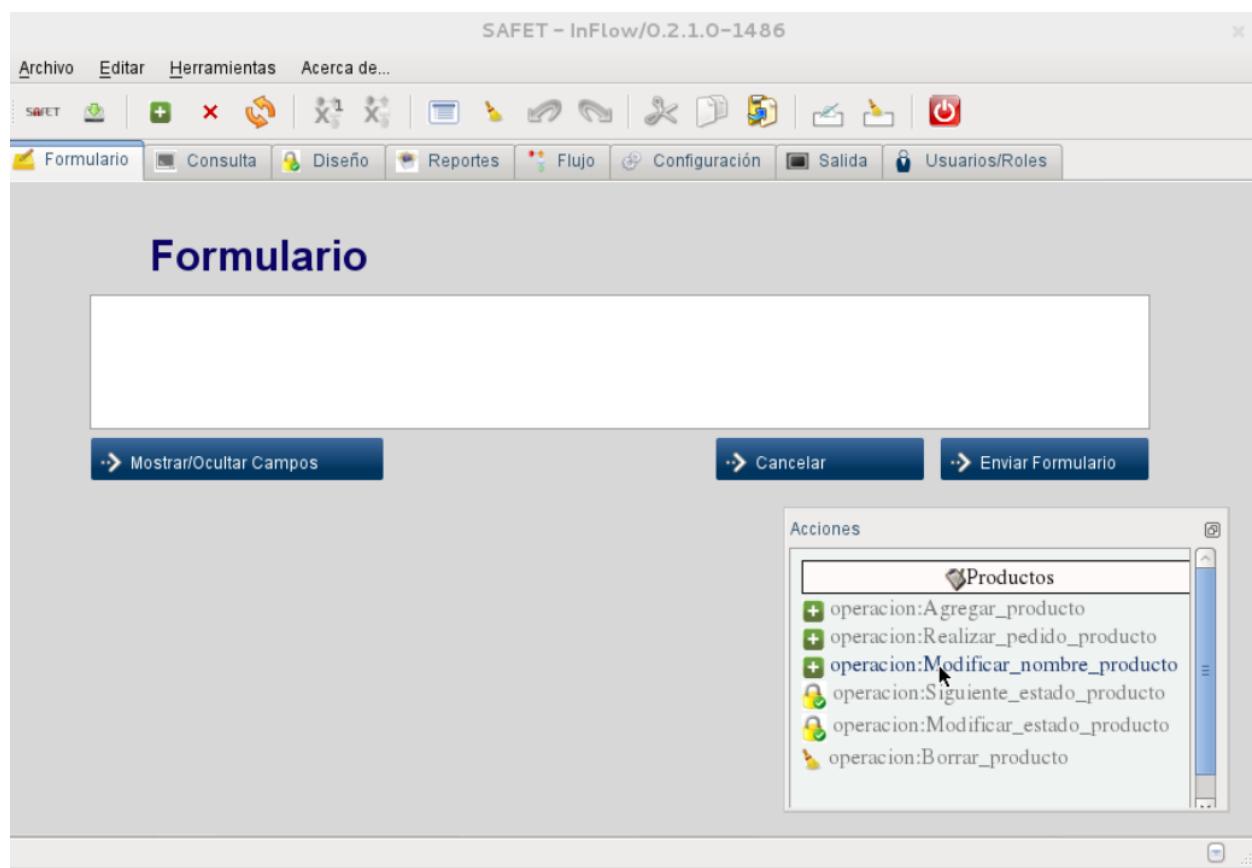


Figura 7.48: Figura 118: Modificar\_nombre\_producto

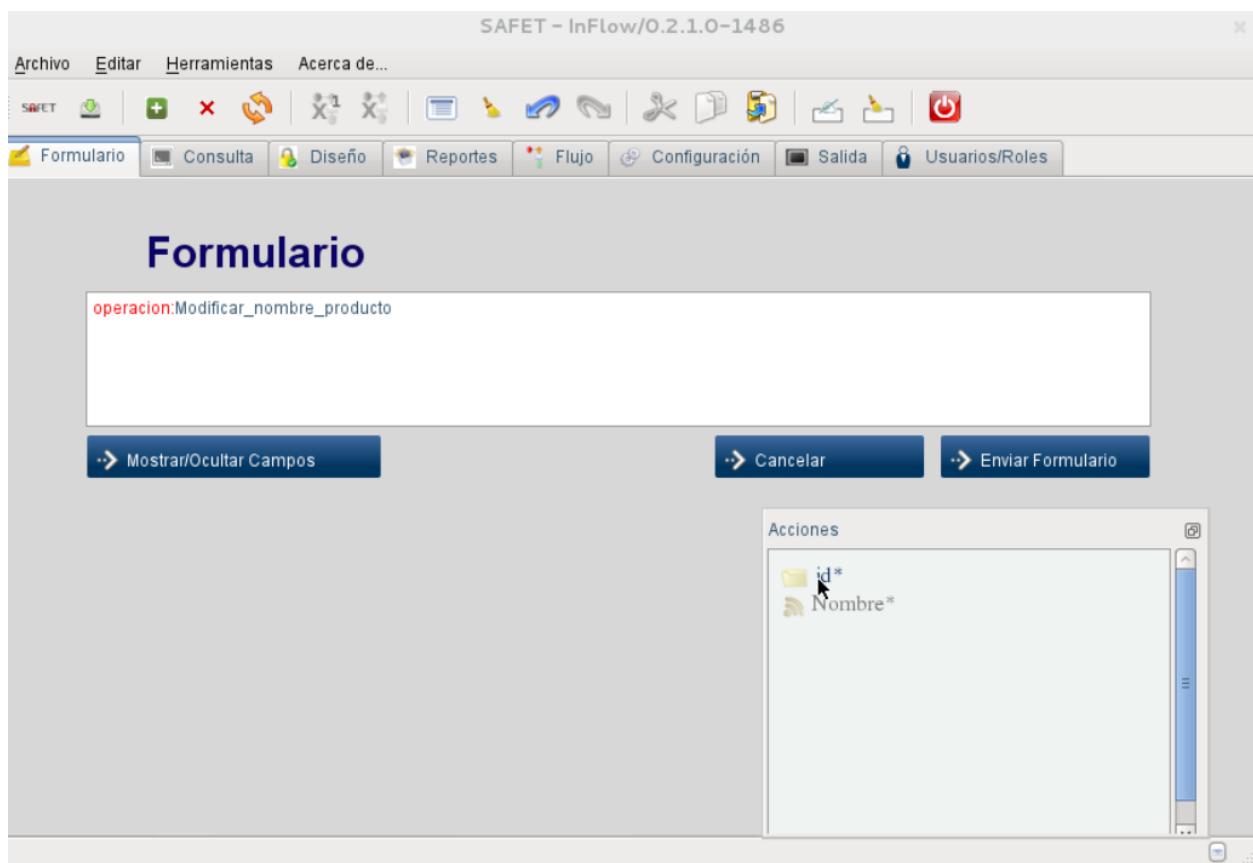


Figura 7.49: Figura 119: Campo (id)

### 3° TERCER PASO

- Seleccionamos el producto a modificar como se muestra en la siguiente *Figura 120: Producto a modificar*

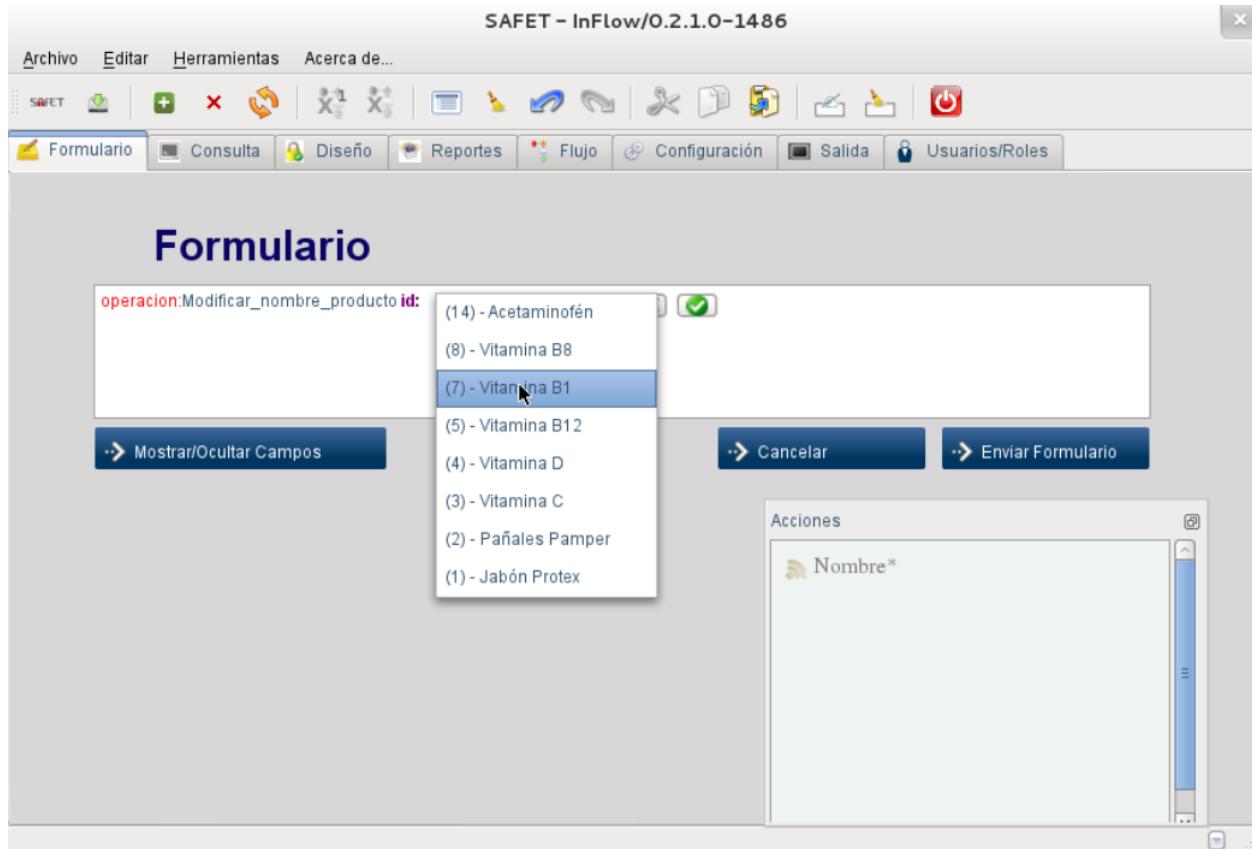


Figura 7.50: **Figura 120: Producto a modificar**

### 4° CUARTO PASO

- Damos un click al botón con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 121: Botón*

### 5° QUINTO PASO

- Damos un click al segundo campo (**Nombre**), como se muestra en la siguiente *Figura 122: Campo (Nombre)*

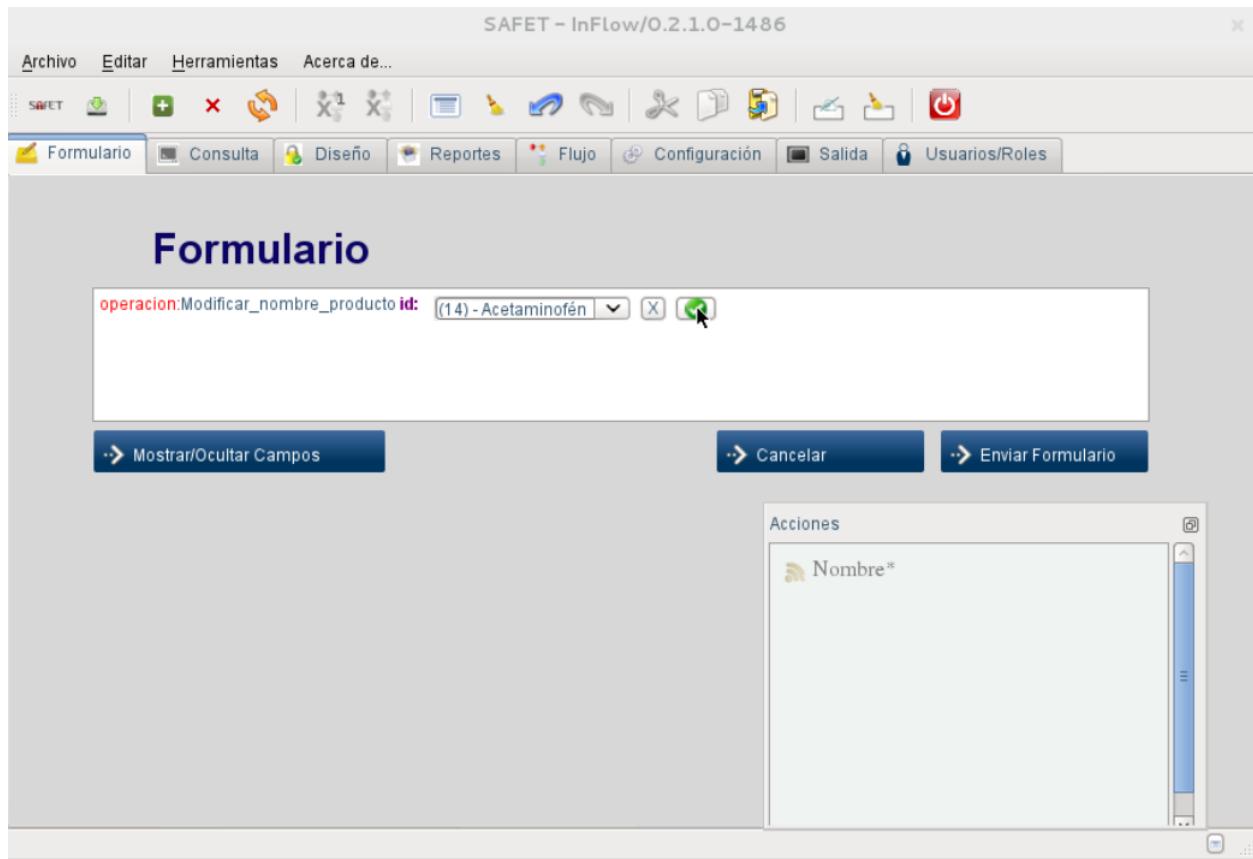


Figura 7.51: **Figura 121: Botón**

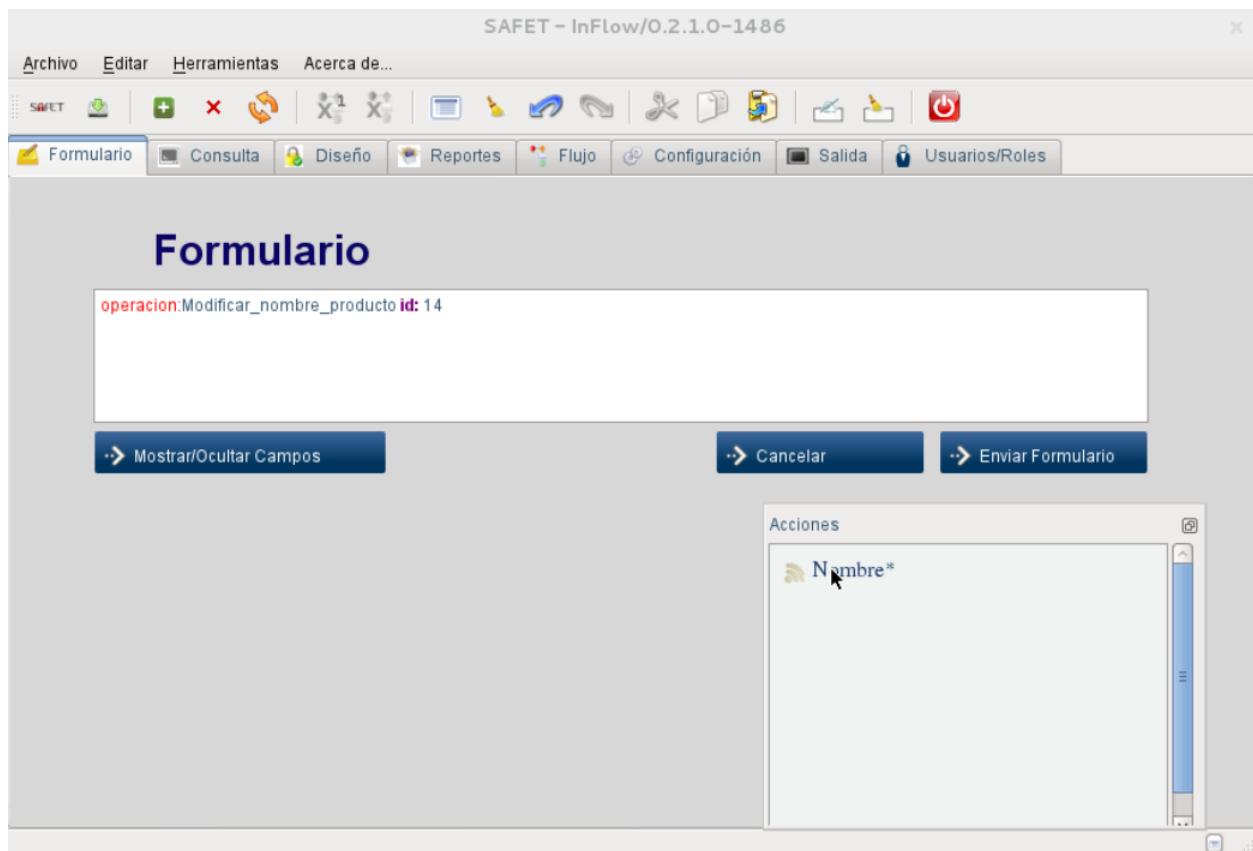


Figura 7.52: **Figura 122: Campo (Nombre)**

### 6° SEXTO PASO

- Ingresamos el nombre a modificar, como se muestra en la siguiente *Figura 123: Nombre a modificar*

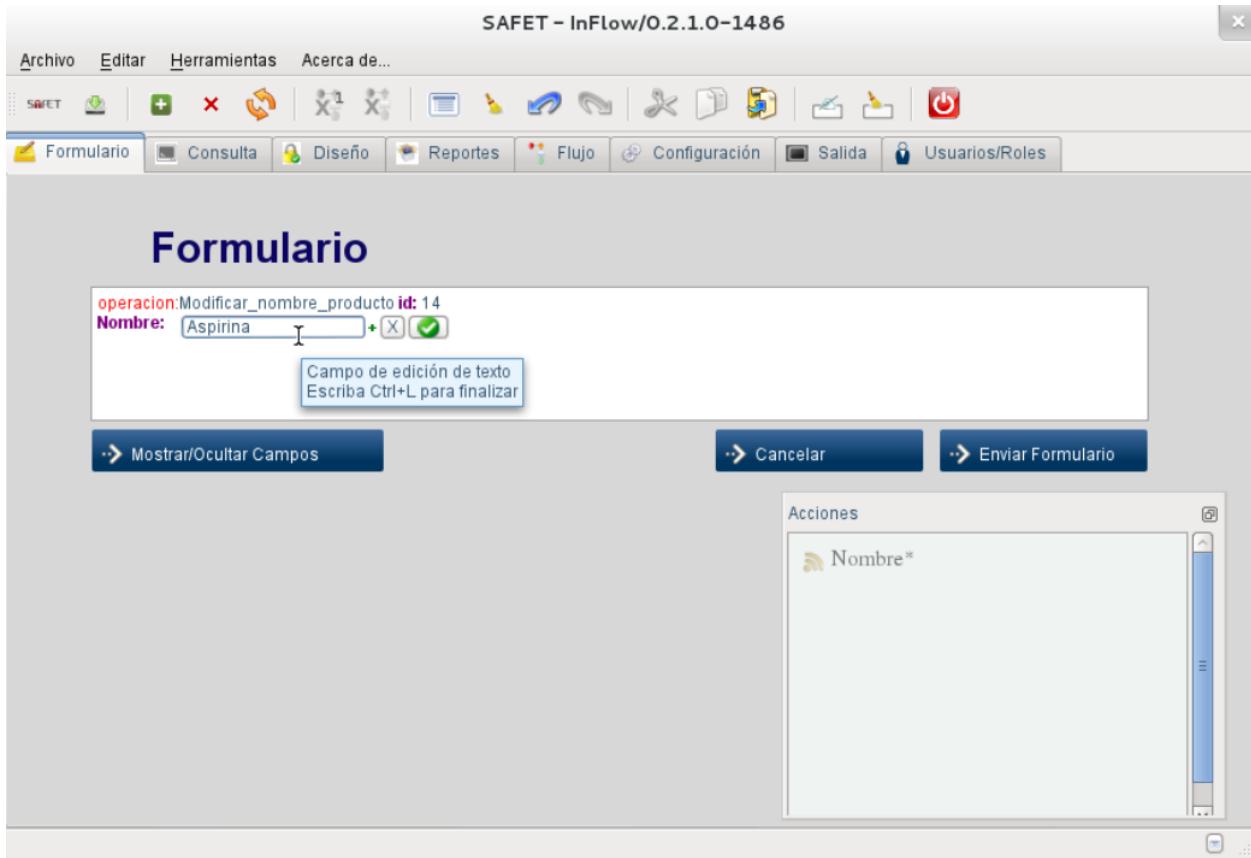


Figura 7.53: **Figura 123: Nombre a modificar**

### 7° SEPTIMO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 124: Botón*

### 8° OCTAVO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**La operación fue exitosa....ok!**), como se muestra en la siguiente *Figura 125: Resultado de la operación*

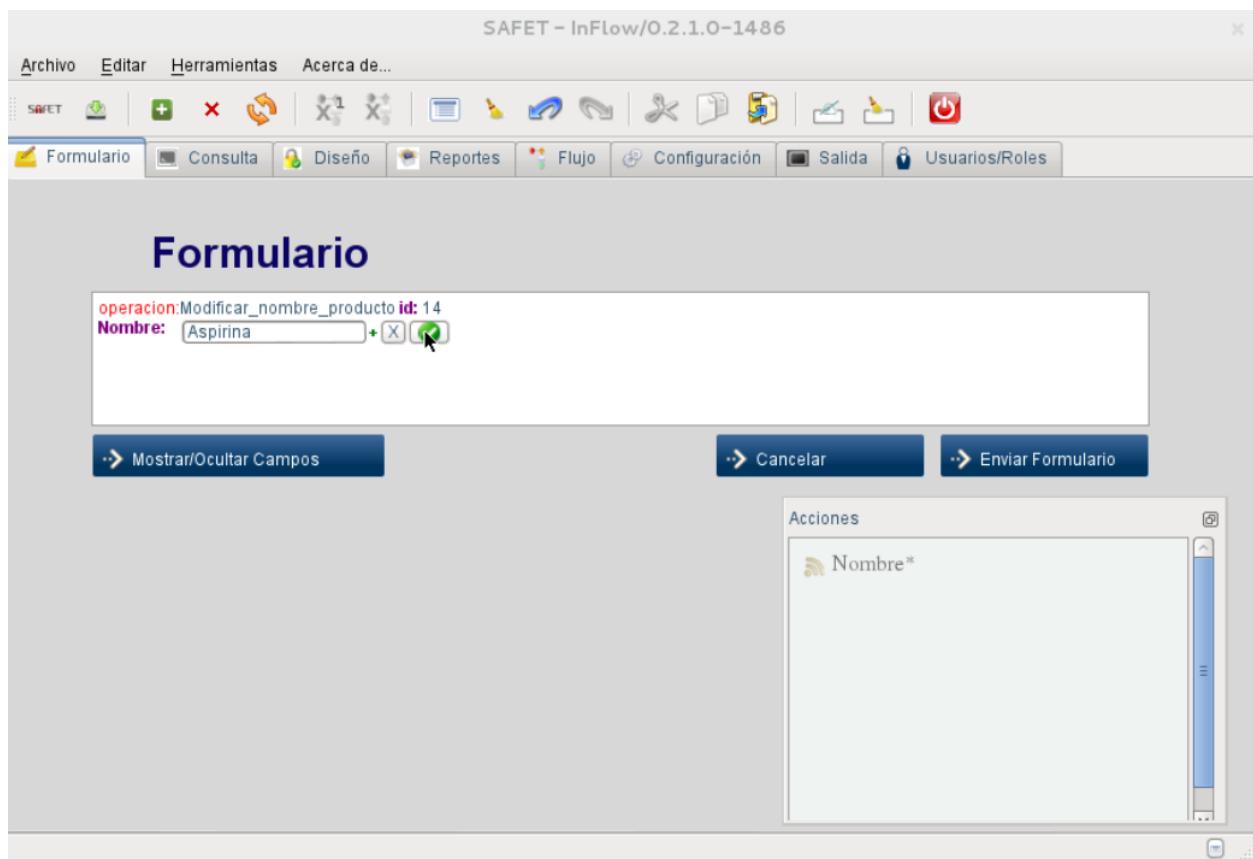


Figura 7.54: Figura 124: Botón

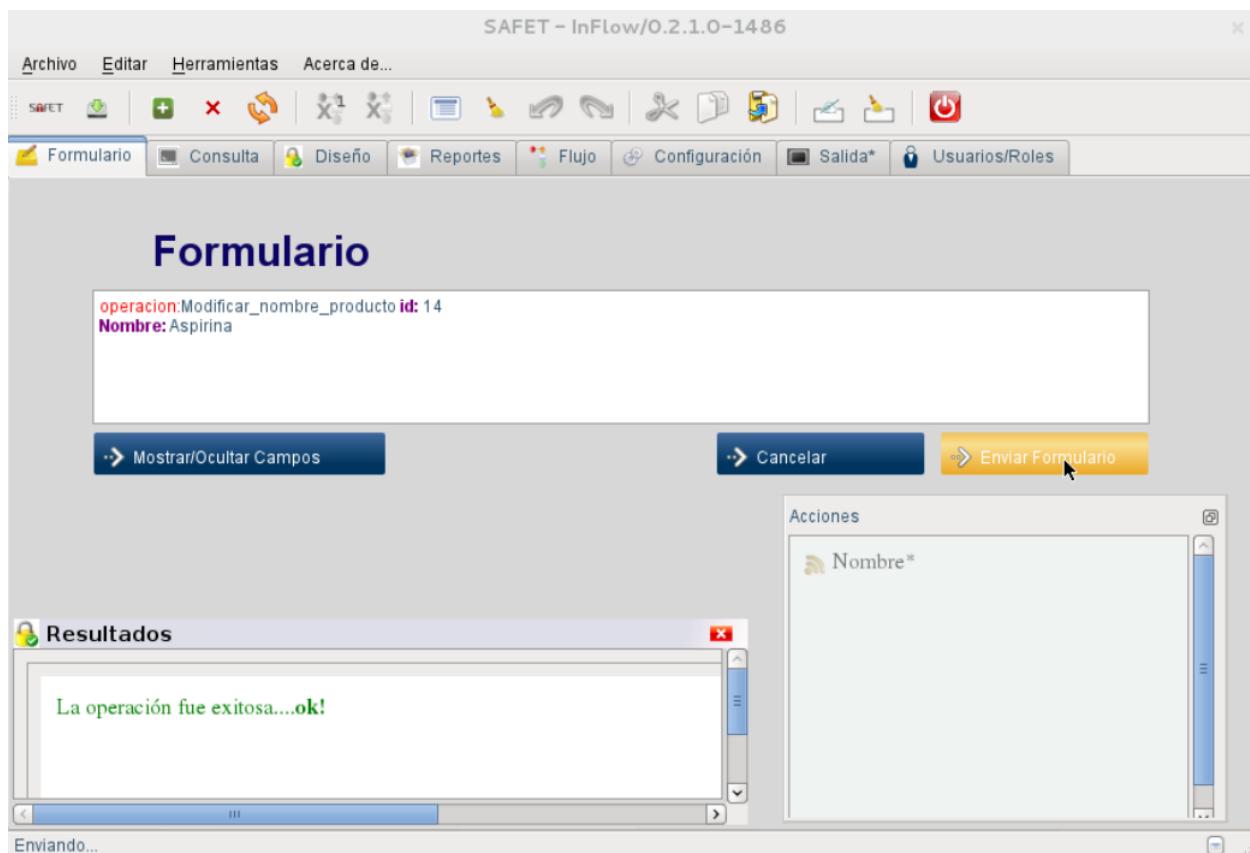


Figura 7.55: Figura 125: Resultado de la operación

## 9° NOVENO PASO

- Para pasar a la siguiente operación damos un click al botón **cancelar** como se muestra en la siguiente *Figura 126: Siguiente operación*

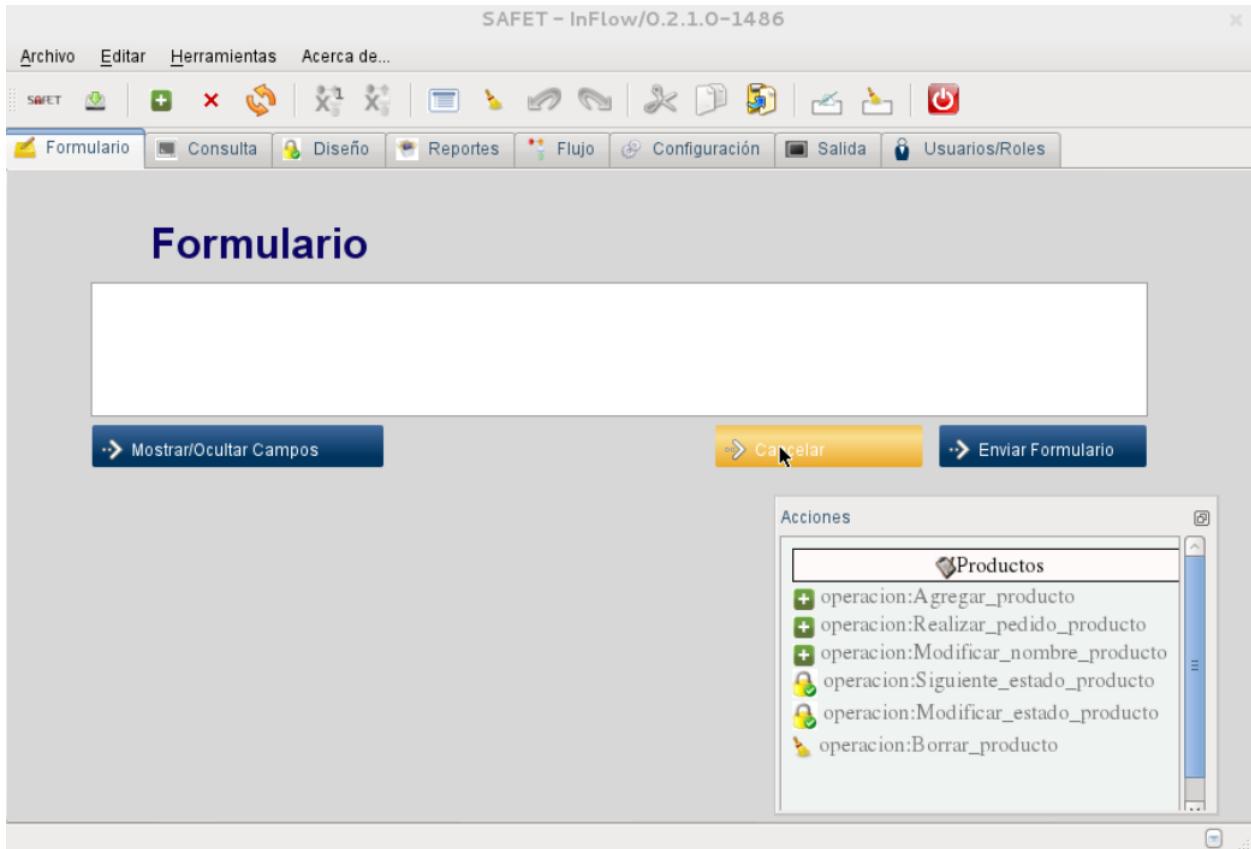


Figura 7.56: **Figura 126: Siguiente operación**

### 7.2.5 4° CUARTA OPERACIÓN CRUD+ (Siguiente estado del producto)

#### 1° PRIMER PASO

- Damos click a la operación (**operacion:Siguiente\_estado\_producto**), como se muestra en la siguiente *Figura 127: Modificar\_nombre\_producto*

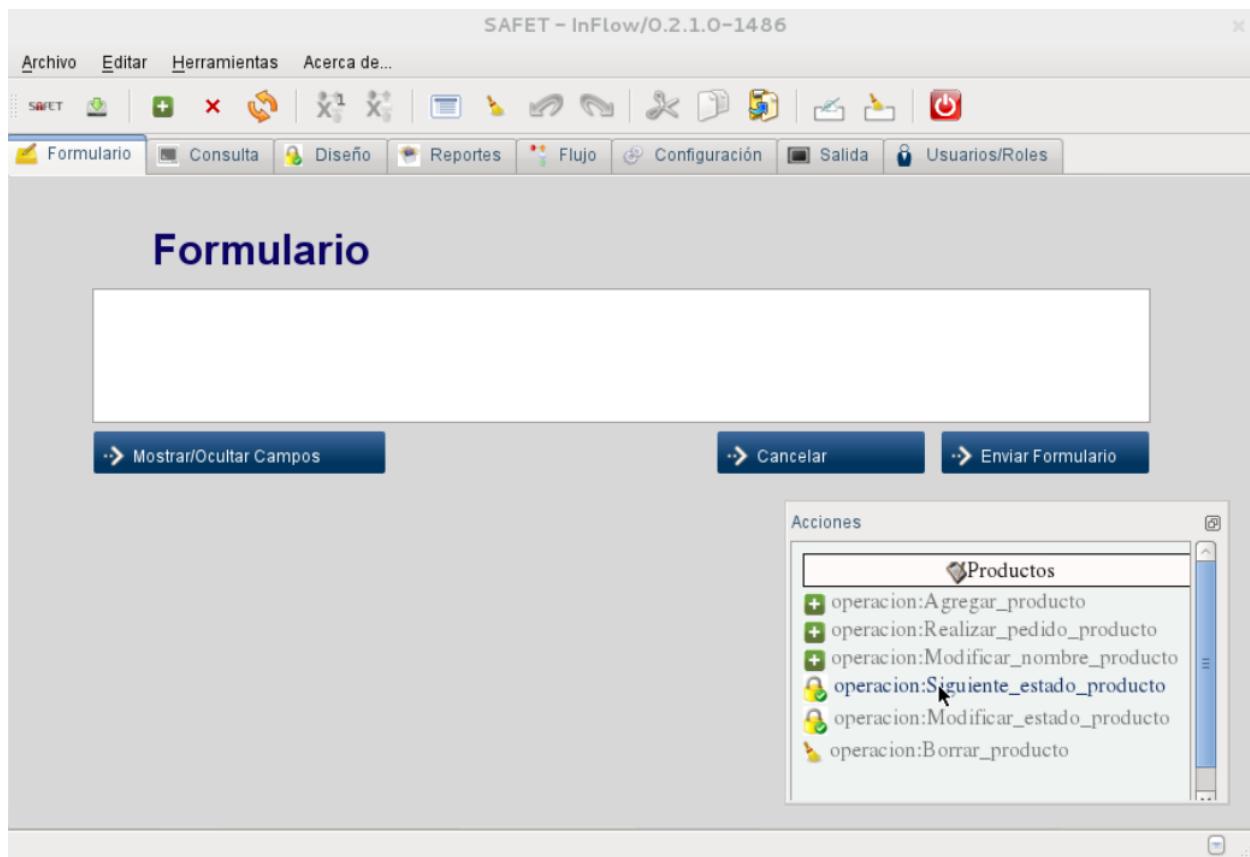


Figura 7.57: Figura 127: Modificar\_nombre\_producto

## 2° SEGUNDO PASO

- En esta (**operacion:Siguiente\_estado\_producto**) tenemos dos campo llamado (**id**) y (**Estado\_producto**), la cual damos un click al primer campo (**id**) para seleccionar el productos a modificar, como se muestra en la siguiente *Figura 128: Campo (id)*

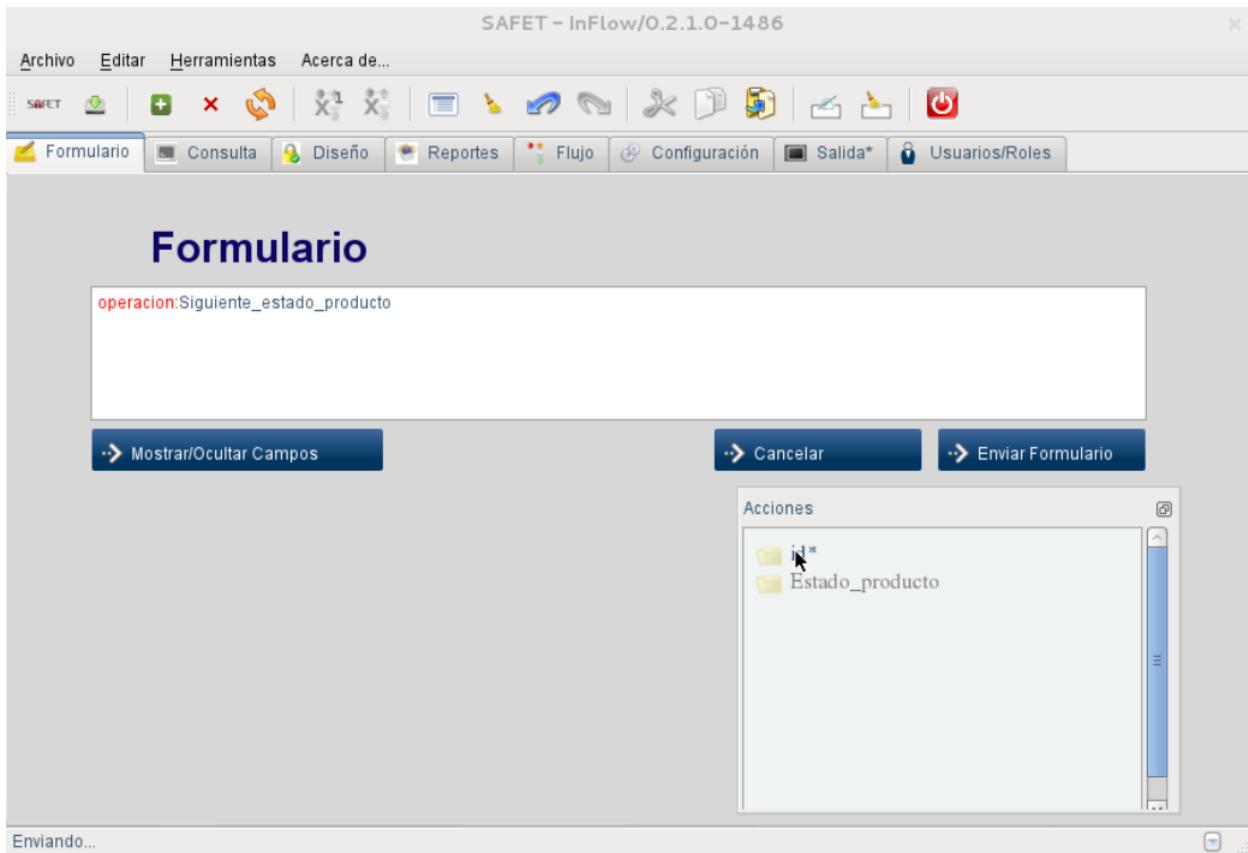


Figura 7.58: **Figura 128: Campo (id)**

## 3° TERCER PASO

- Seleccionamos el producto a pasar al siguientes estado, como se muestra en la siguiente *Figura 129: Pasar al siguientes estado*

## 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 130: Botón*

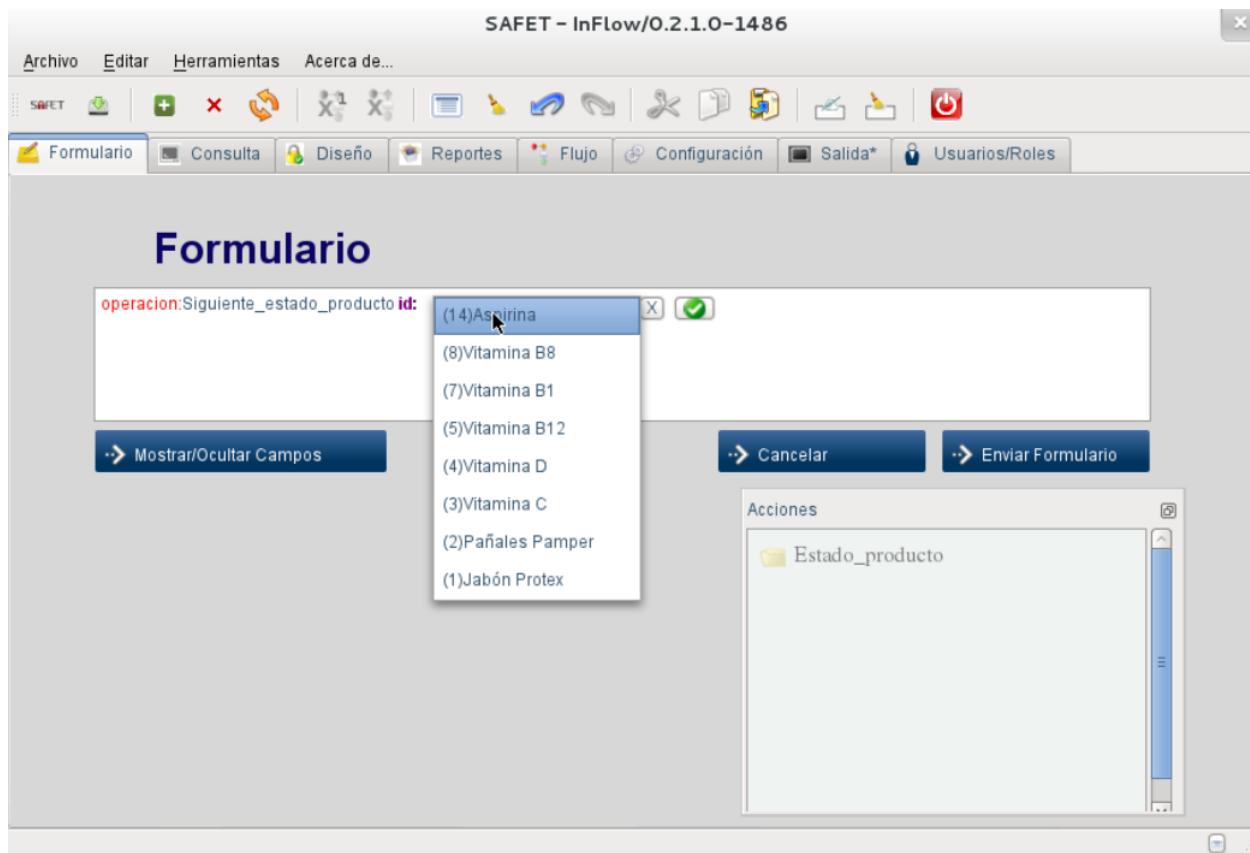


Figura 7.59: Figura 129: Pasar al siguientes estado

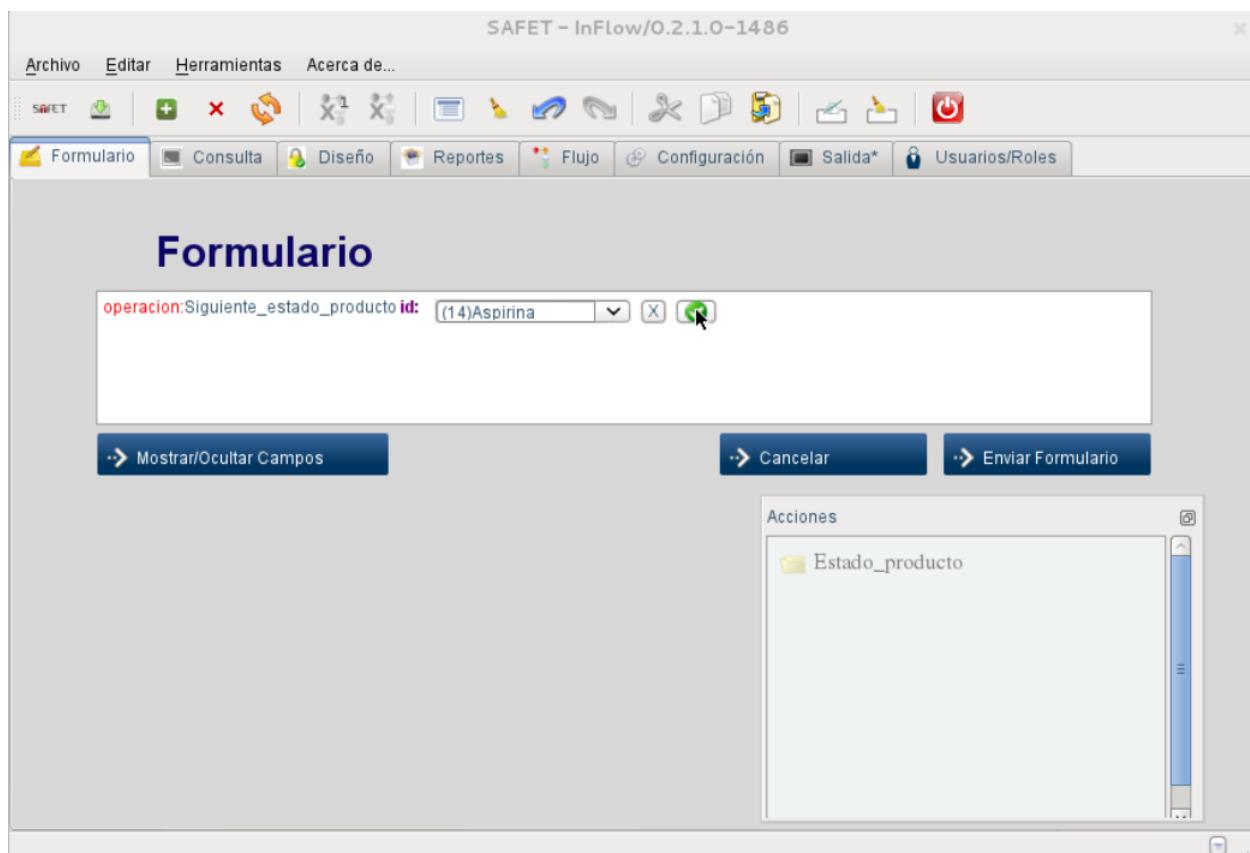


Figura 7.60: Figura 130: Botón

### 5° QUINTO PASO

- Damos un click al segundo campo (**Estado\_producto**), como se muestra en la siguiente *Figura 131: Campo (Estado\_producto)*

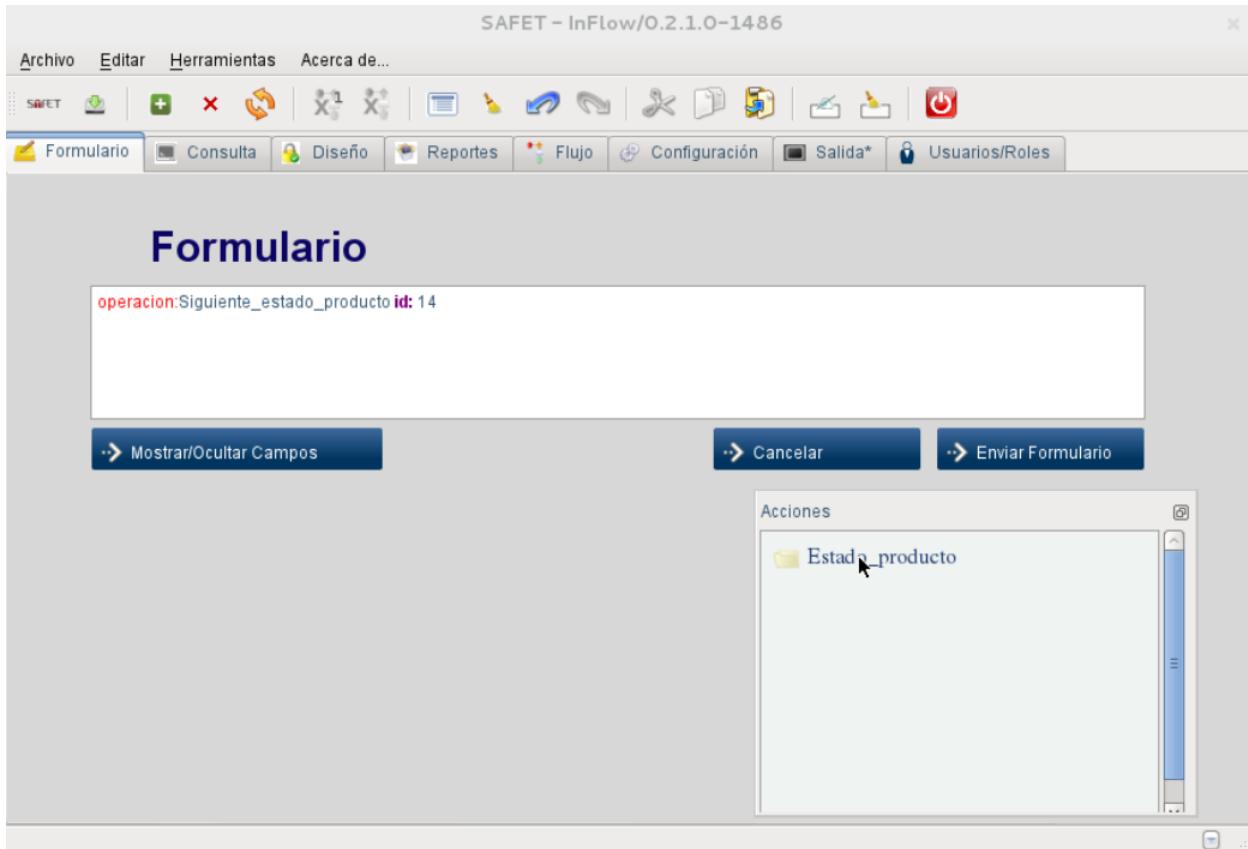


Figura 7.61: **Figura 131: Campo (Estado\_producto)**

### 6° SEXTO PASO

- Seleccionamos el estado que va a pasar el producto (**Disponible, Por\_llegar**), como se muestra en la siguiente *Figura 132: Nombre a modificar*

### 7° SEPTIMO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 133: Botón*

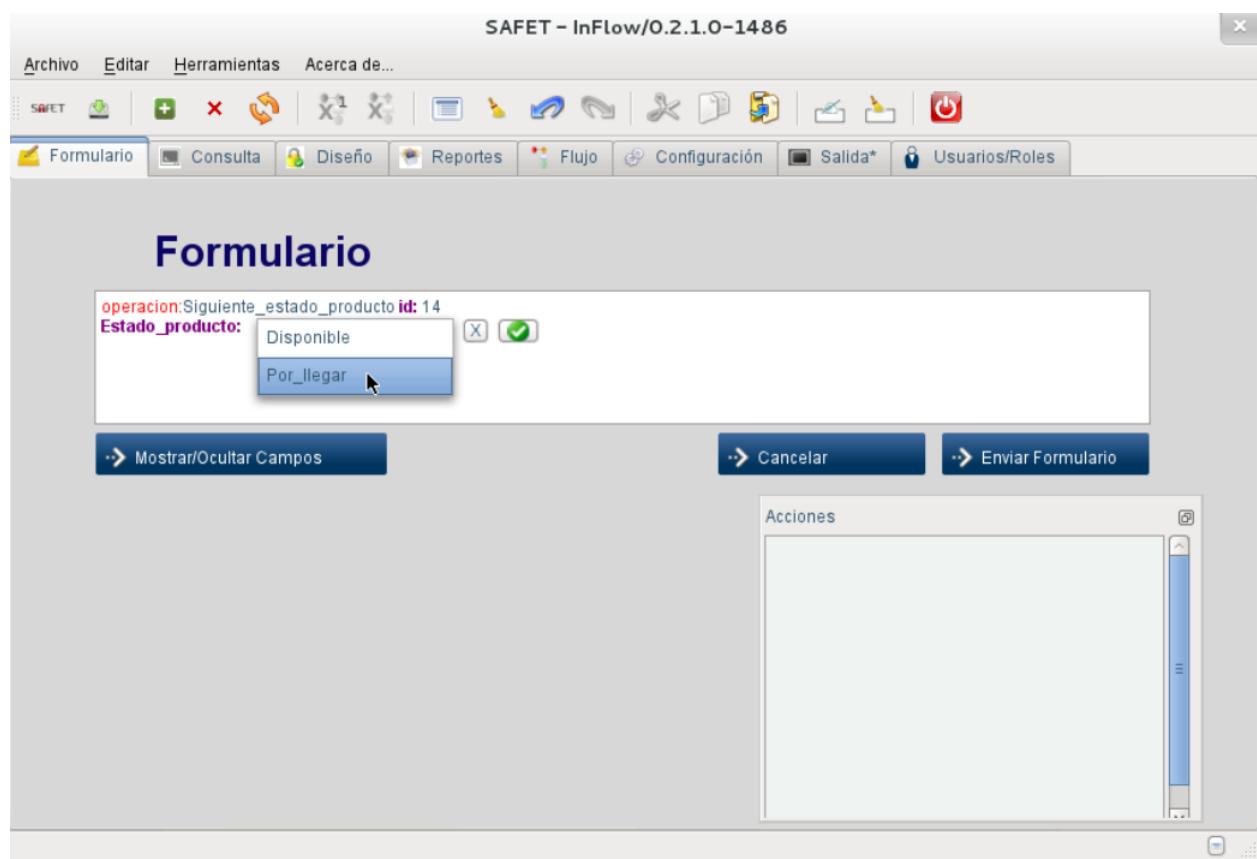


Figura 7.62: Figura 132: Nombre a modificar

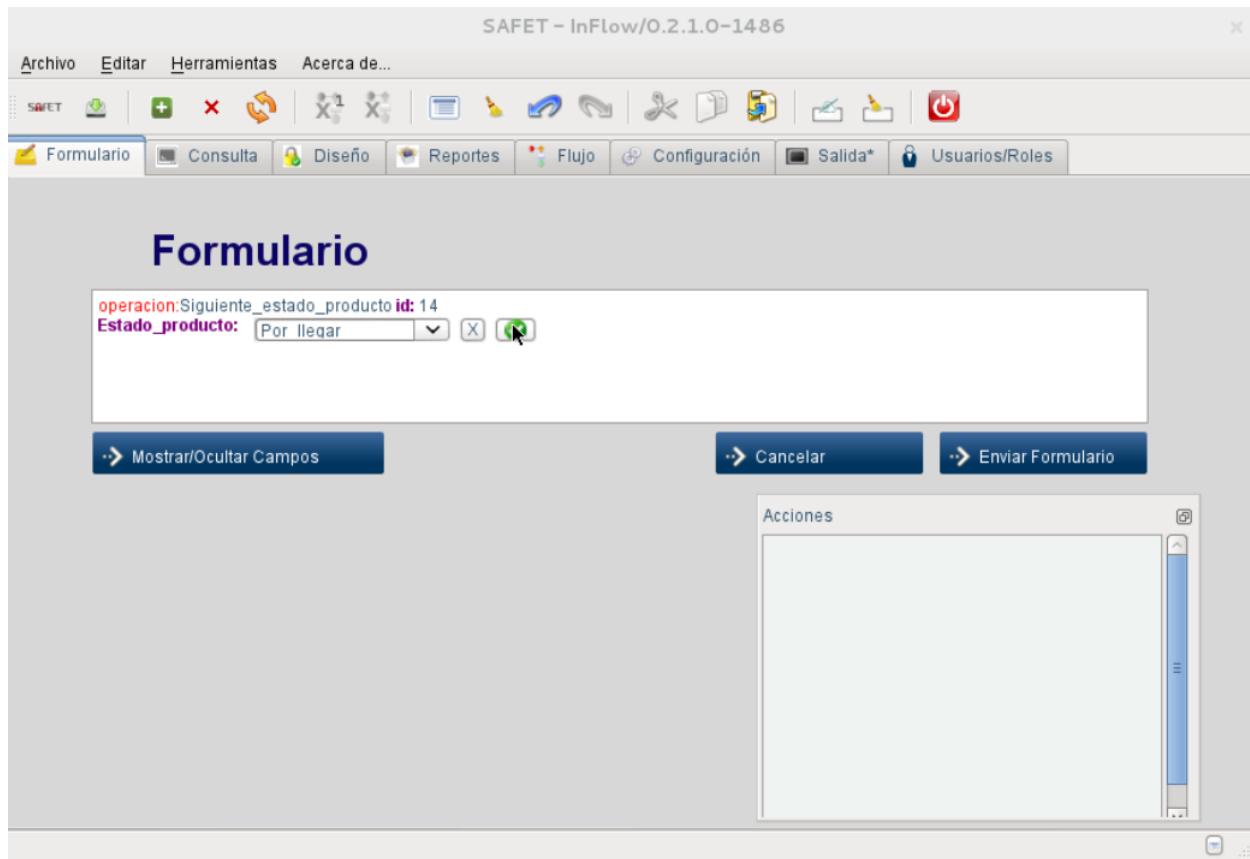


Figura 7.63: **Figura 133: Botón**

## 8° OCTAVO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**La operación fue exitosa....ok!**), como se muestra en la siguiente *Figura 134: Resultado de la operación*

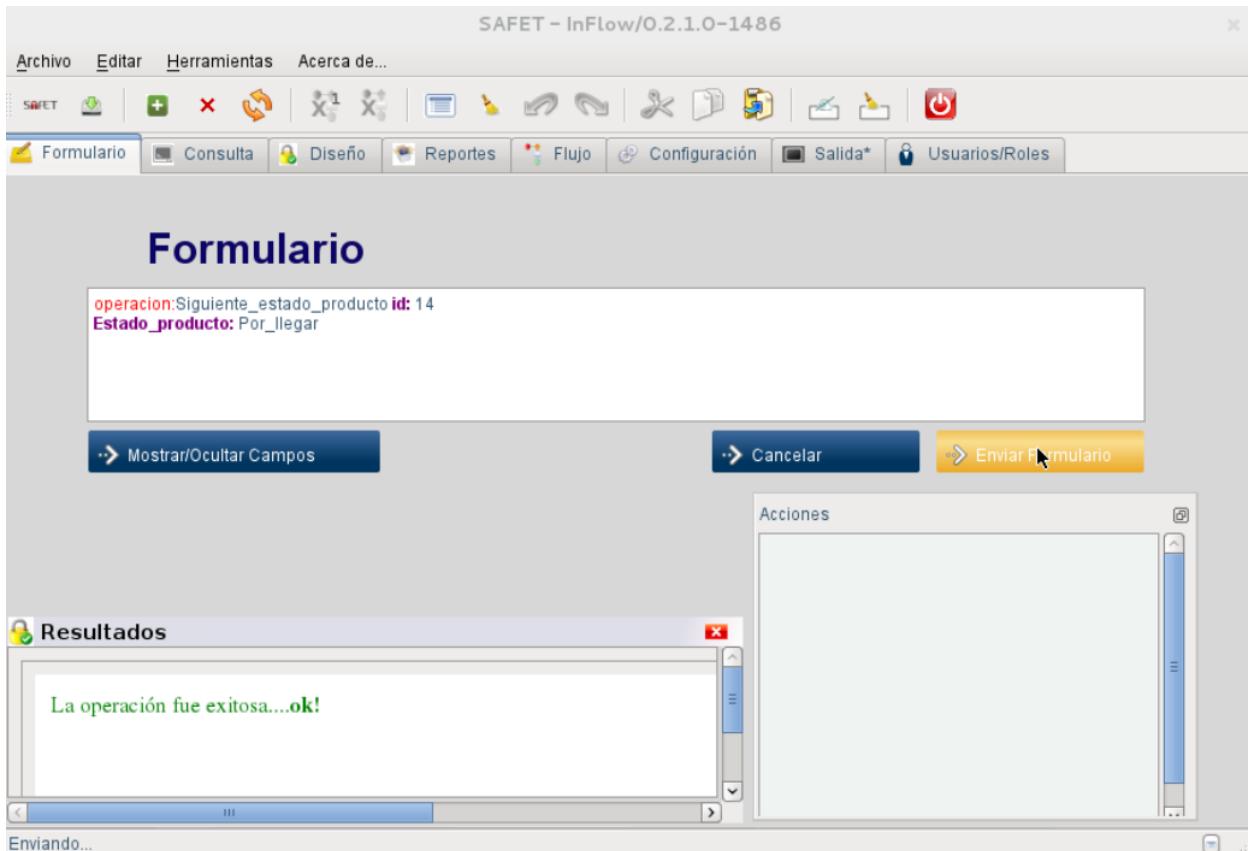


Figura 7.64: **Figura 134: Resultado de la operación**

## 9° NOVENO PASO

- Para pasar a la siguiente operación damos un click al botón **cancelar** como se muestra en la siguiente *Figura 135: Siguiente operación*

### 7.2.6 5° QUINTA OPERACIÓN CRUD+ (Modificar estado del producto)

#### 1° PRIMER PASO

- Damos click a la operación (**operacion:Modificar\_estado\_producto**), como se muestra en la siguiente *Figura 136: Modificar\_estado\_producto*

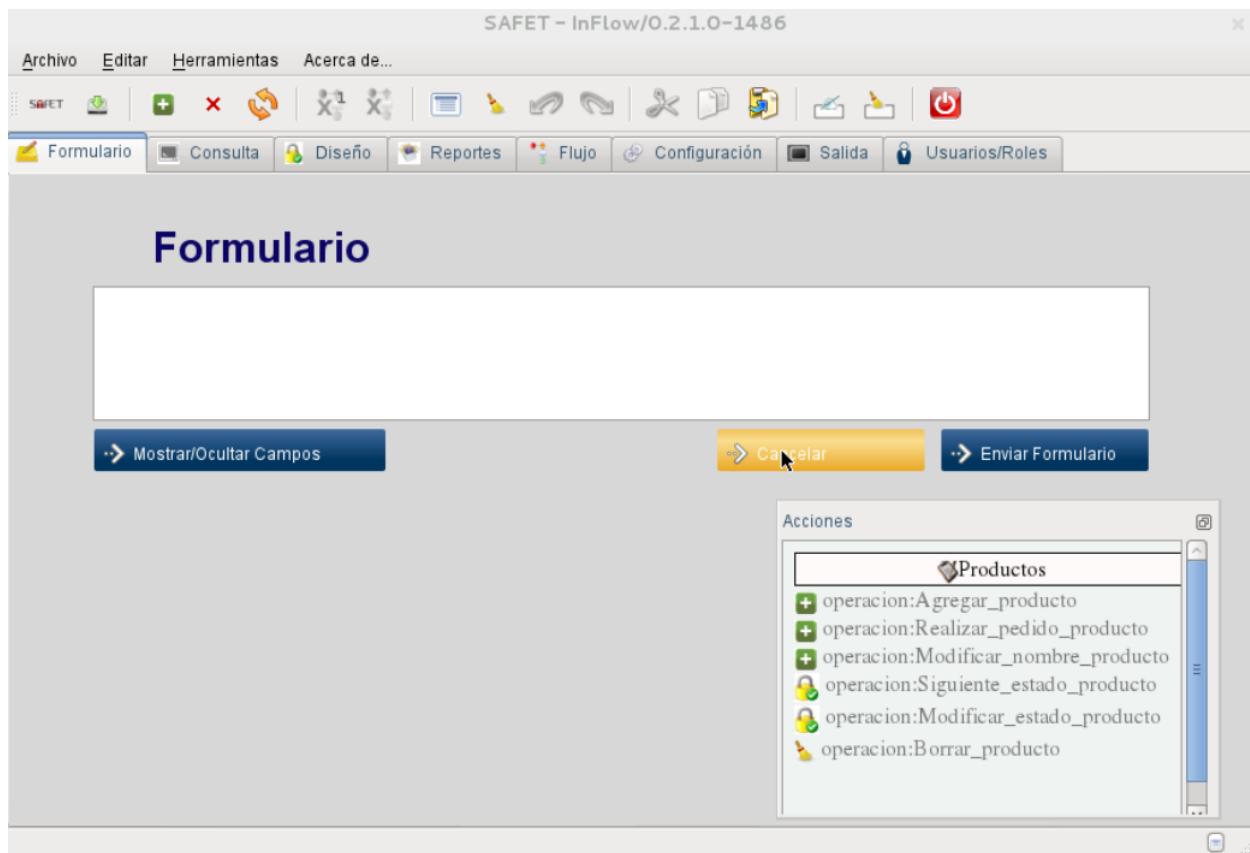


Figura 7.65: Figura 135: Siguiente operación

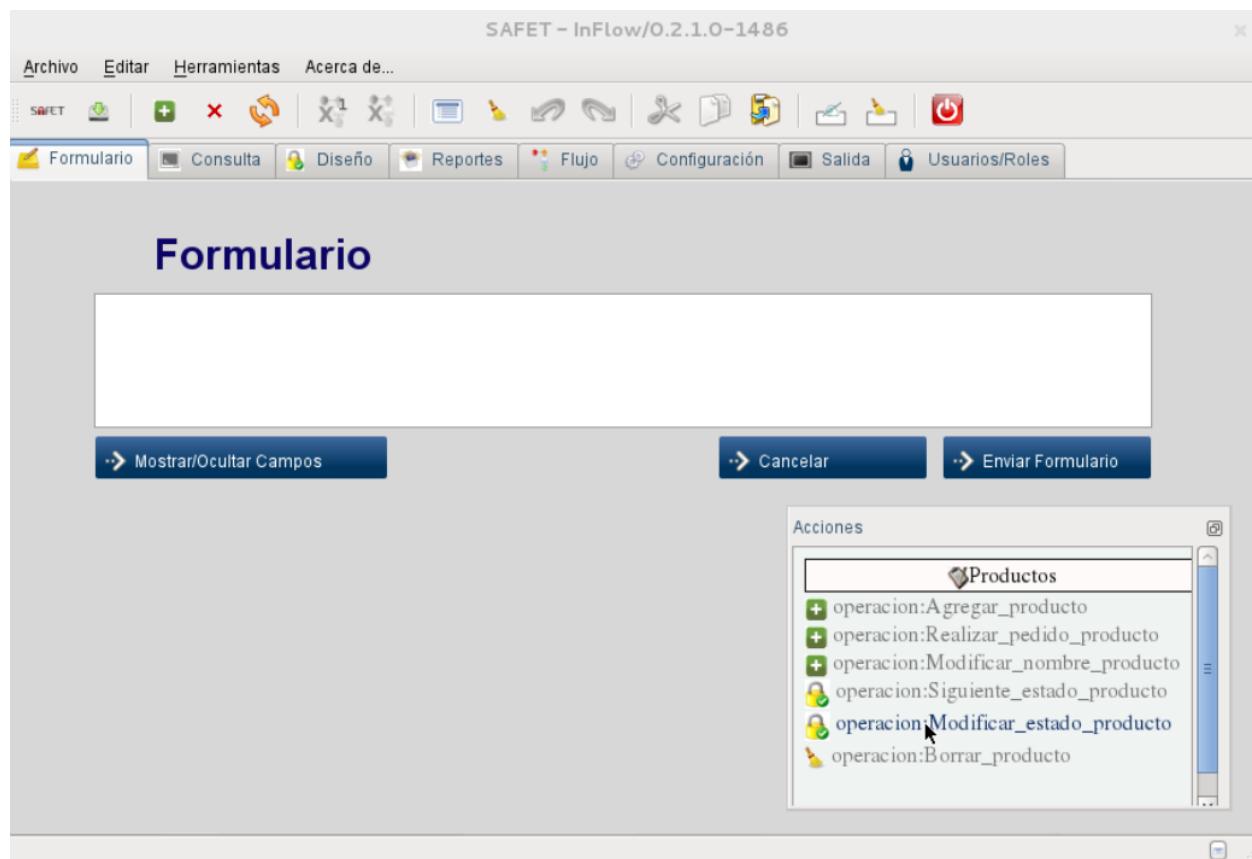


Figura 7.66: Figura 136: Modificar\_estado\_producto

## 2° SEGUNDO PASO

- En esta (**operacion:Modificar\_estado\_producto**) tenemos dos campo llamado (**id**) y (**Estados\_a\_modificar**), la cual damos un click al primer campo (**id**) para seleccionar el productos a modificar su estado, como se muestra en la siguiente *Figura 137: Campo (id)*

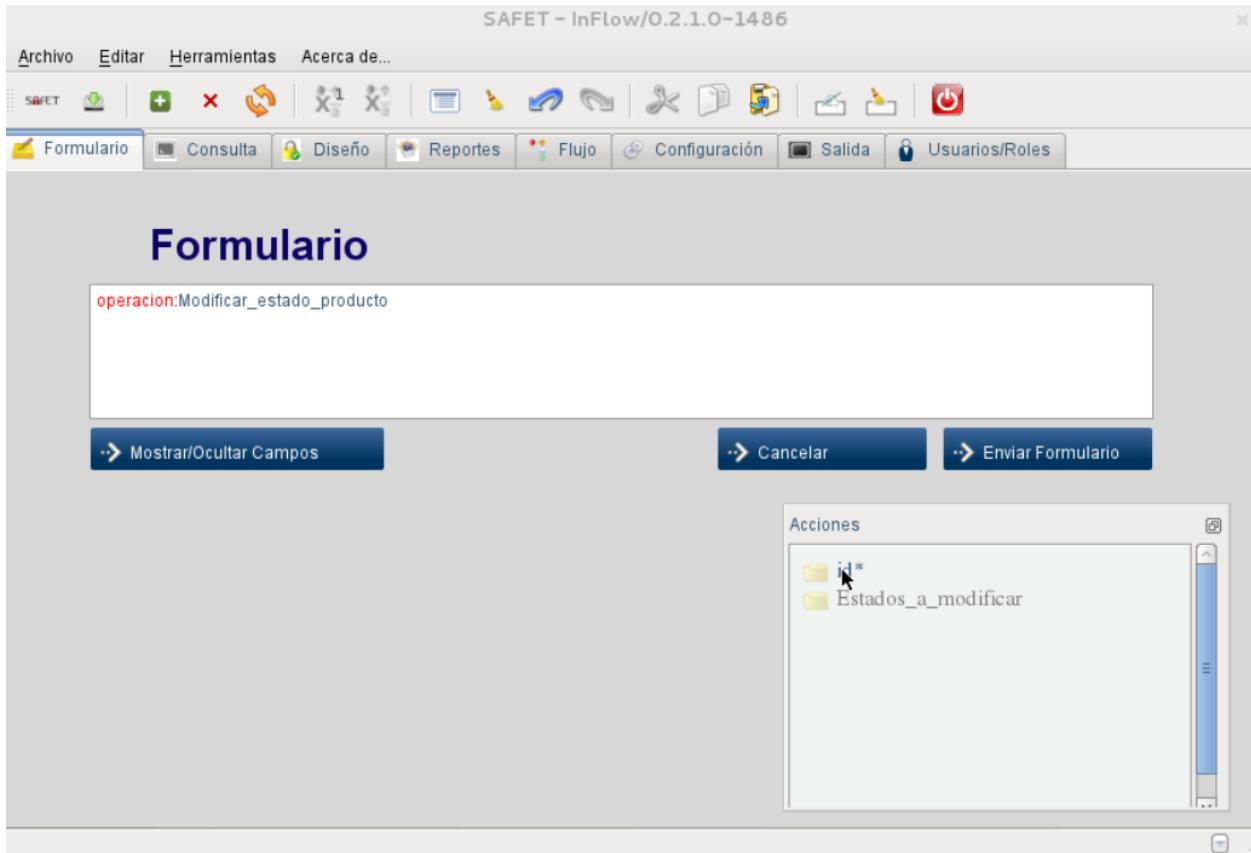


Figura 7.67: **Figura 137: Campo (id)**

## 3° TERCER PASO

- Seleccionamos el producto a modificar su estado, como se muestra en la siguiente *Figura 138: Pasar al siguientes estado*

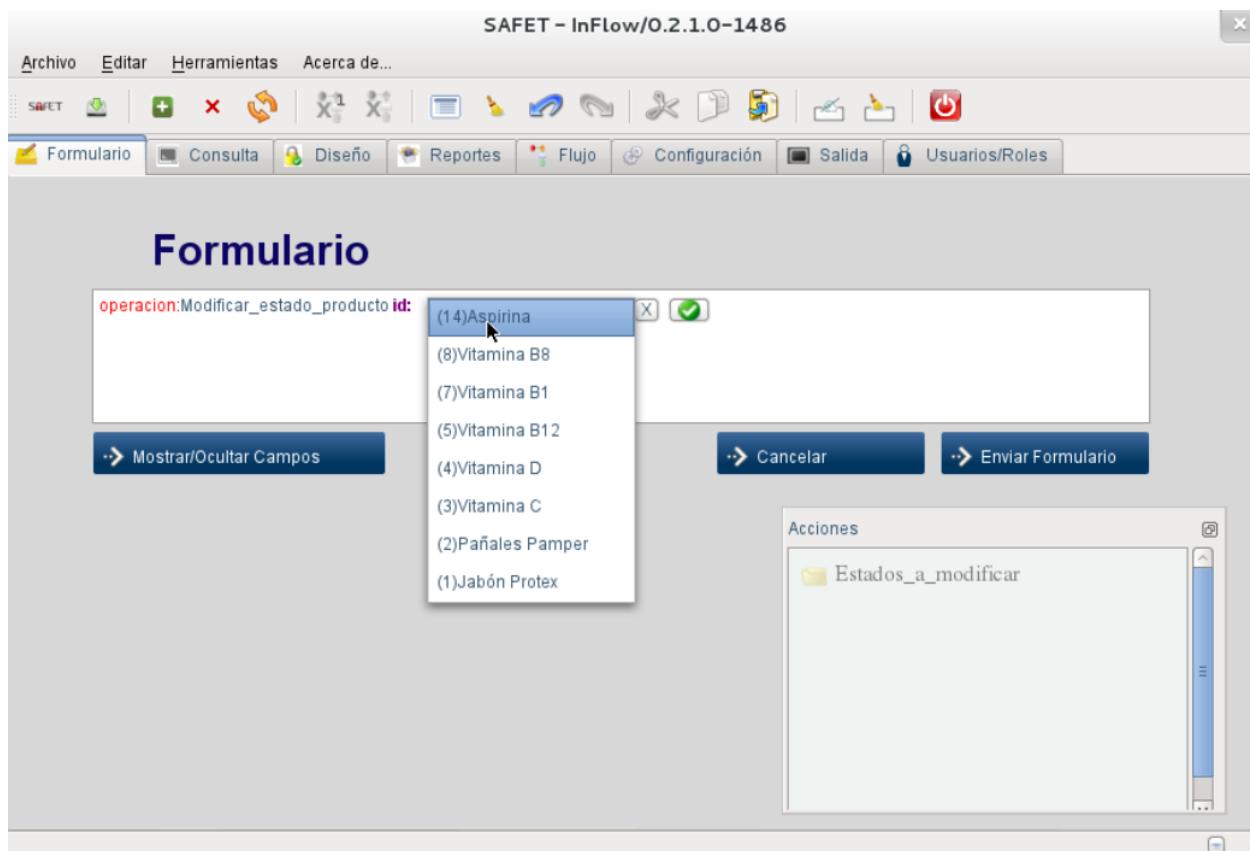


Figura 7.68: Figura 138: Pasar al siguientes estado

### 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 139: Botón*

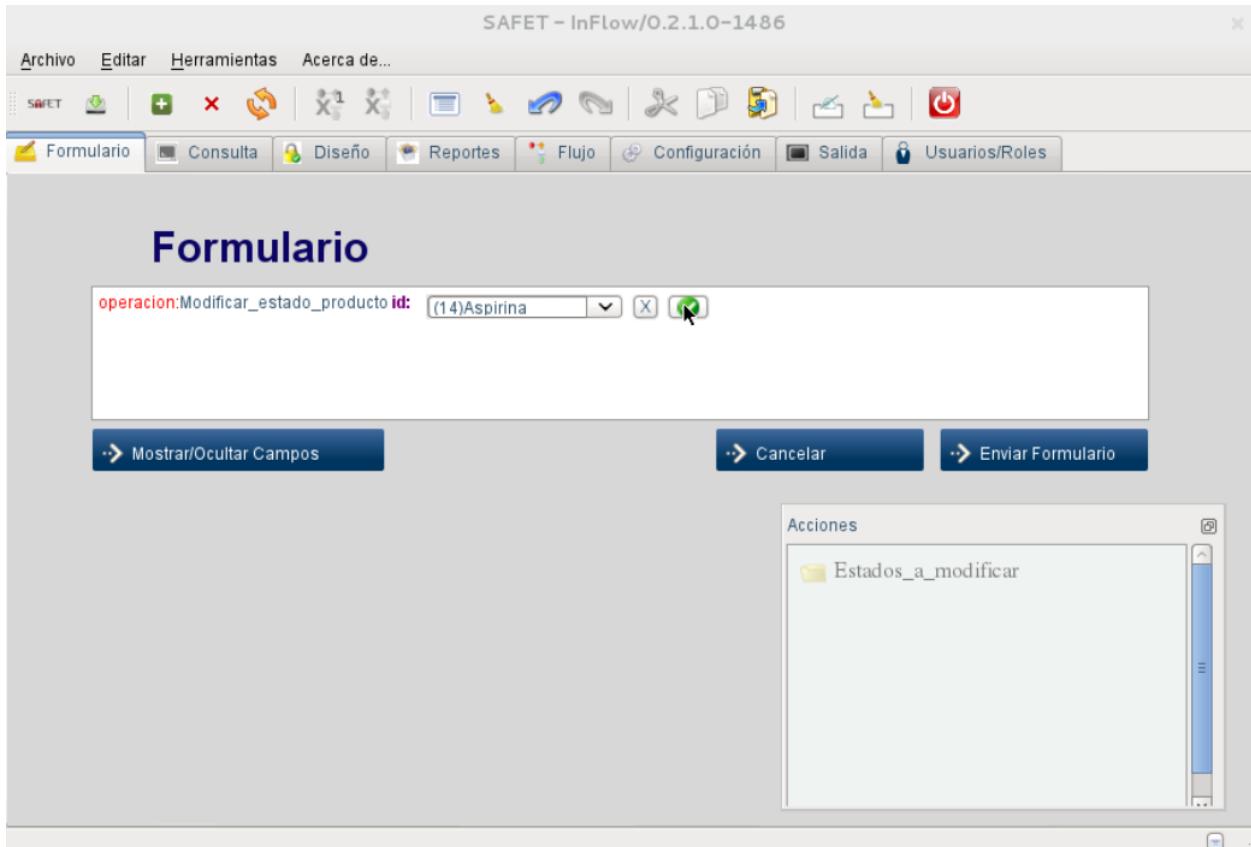


Figura 7.69: **Figura 139: Botón**

### 5° QUINTO PASO

- Damos un click al segundo campo (**Estados\_a\_modificar**), como se muestra en la siguiente *Figura 140: Campo (Estados\_a\_modificar)*

### 6° SEXTO PASO

- Seleccionamos el estado a modificar (**Disponible, Por\_agotarse, Pedido**), como se muestra en la siguiente *Figura 141: Modificar estado*

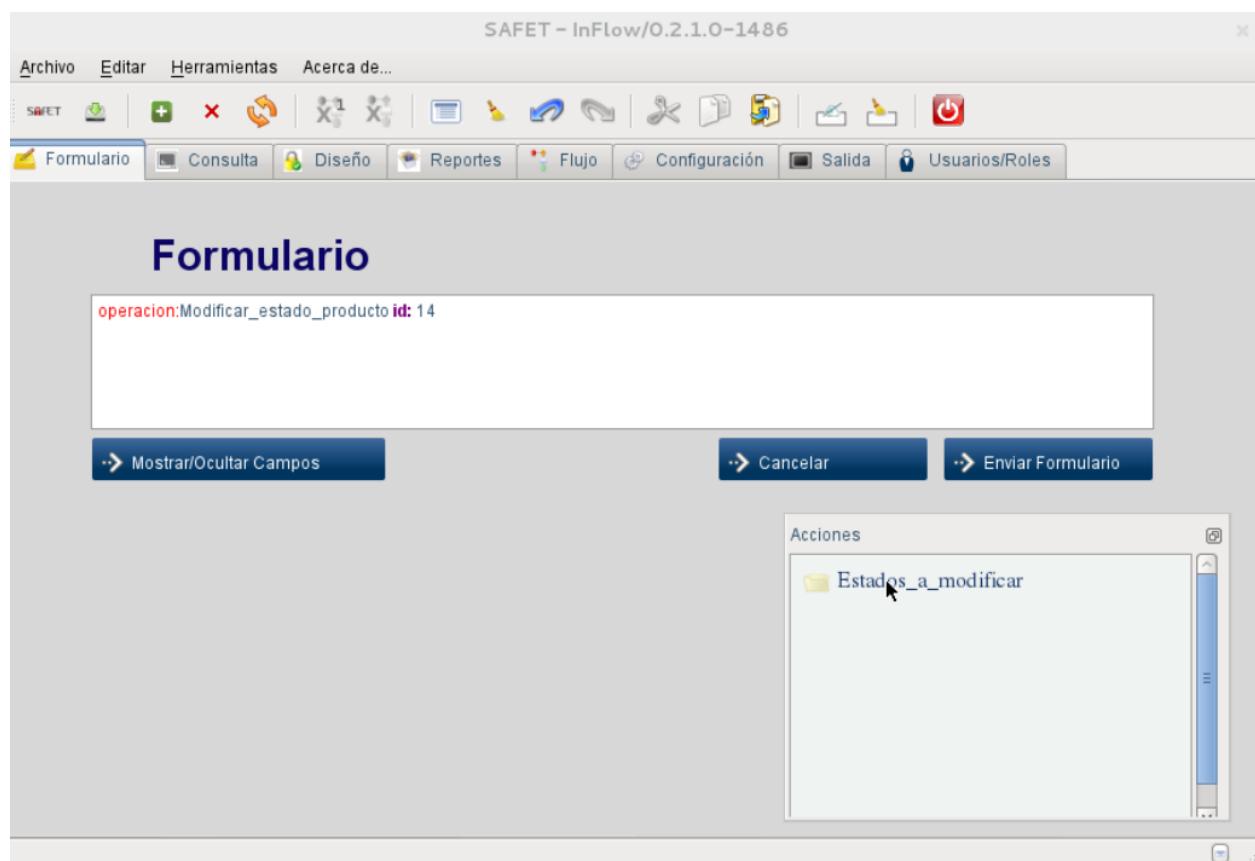


Figura 7.70: Figura 140: Campo (Estados\_a\_modificar)

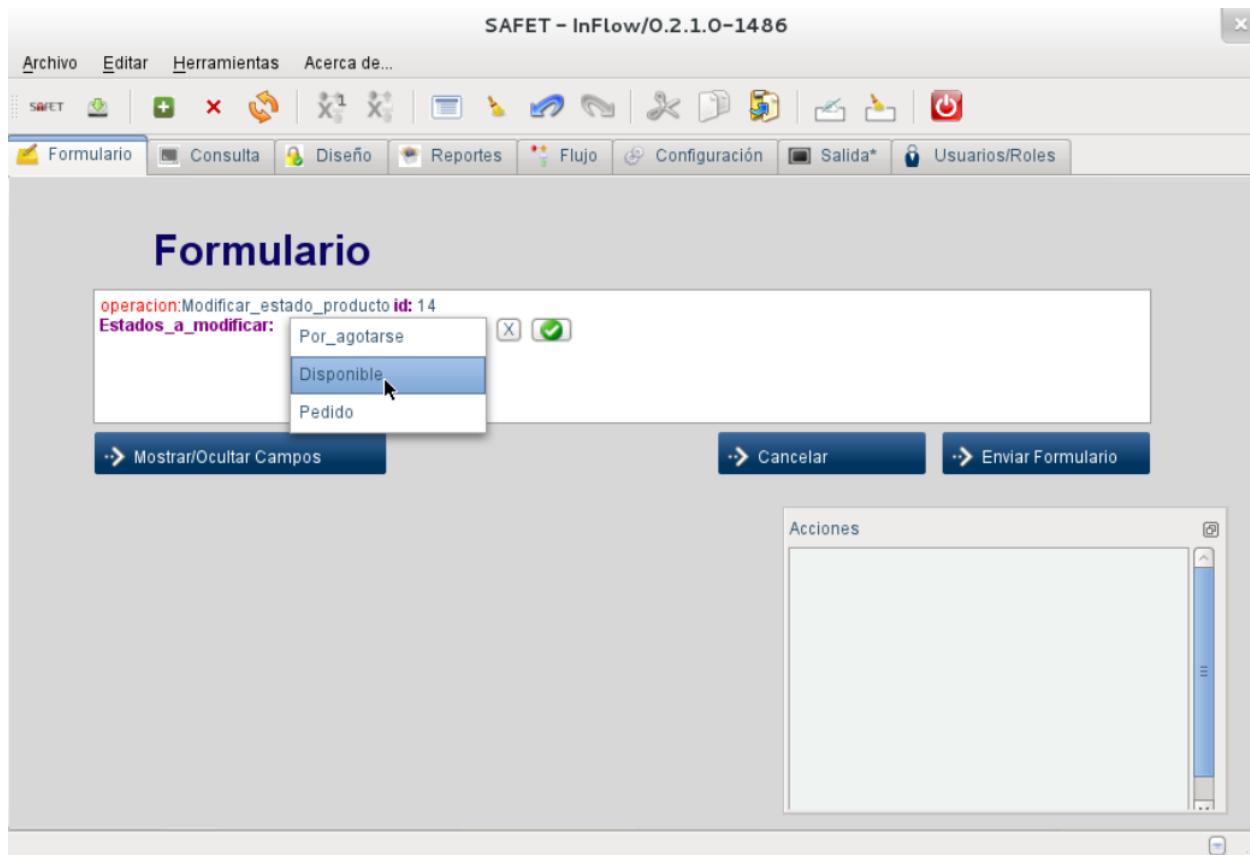


Figura 7.71: Figura 141: Modificar estado

## 7° SEPTIMO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 142: Botón*

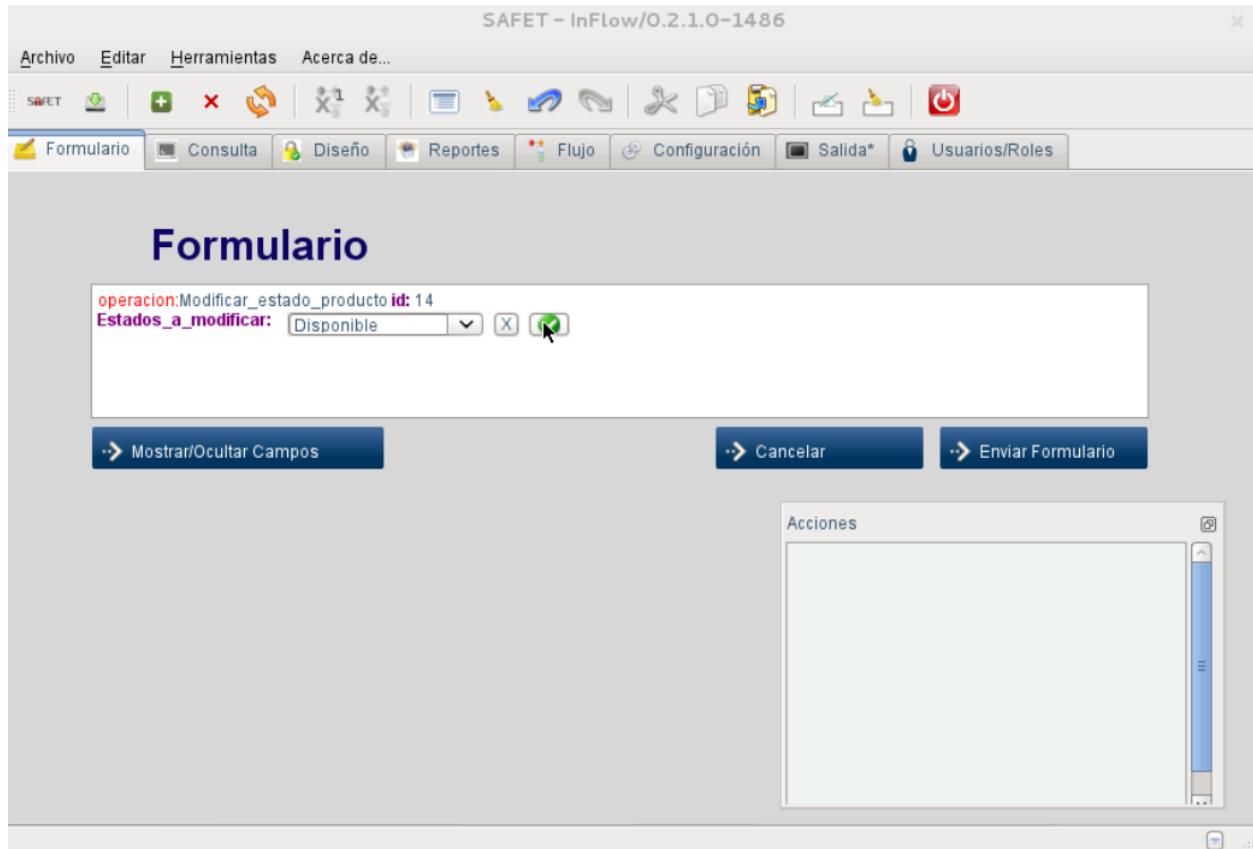


Figura 7.72: **Figura 142: Botón**

## 8° OCTAVO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**La operación fue exitosa....ok!**), como se muestra en la siguiente *Figura 143: Resultado de la operación*

## 9° NOVENO PASO

- Para pasar a la siguiente operación damos un click al botón **cancelar** como se muestra en la siguiente *Figura 144: Siguiente operación*

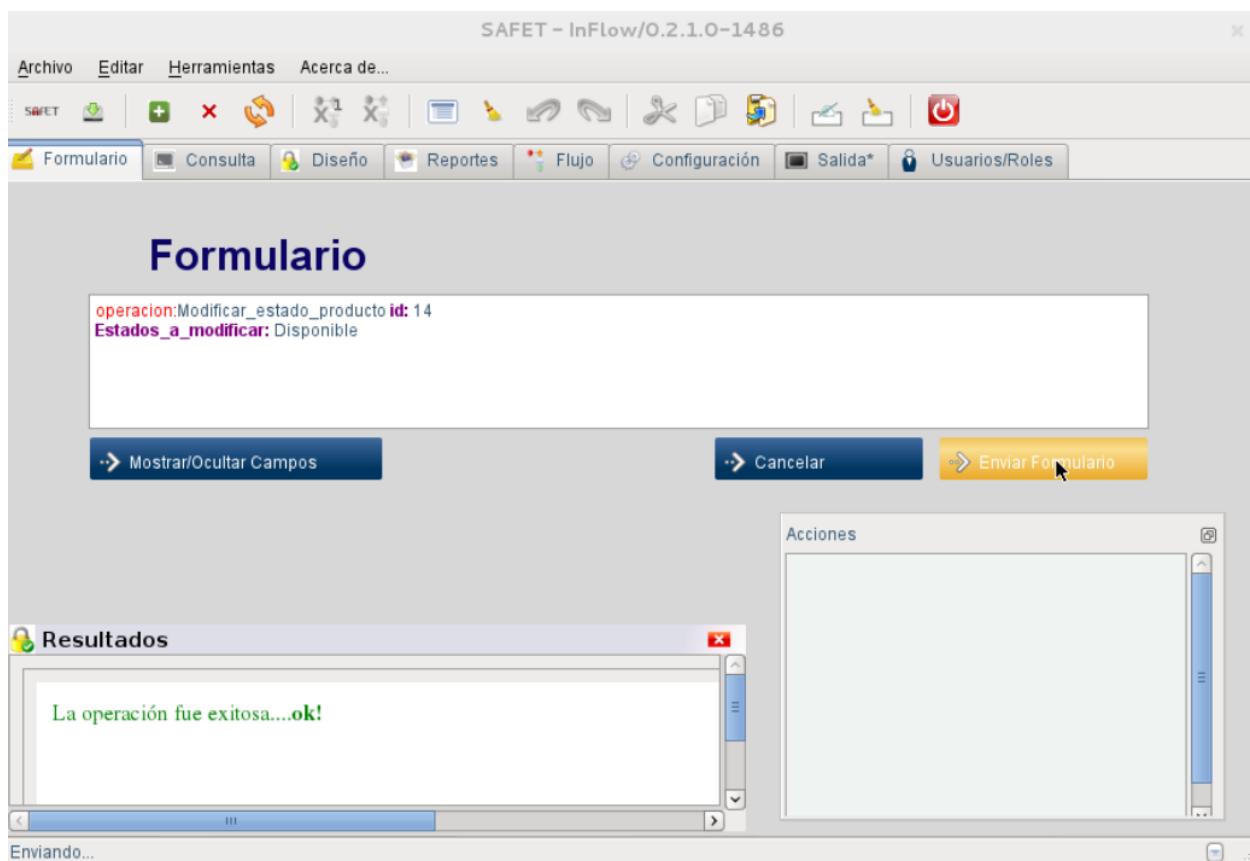


Figura 7.73: Figura 143: Resultado de la operación

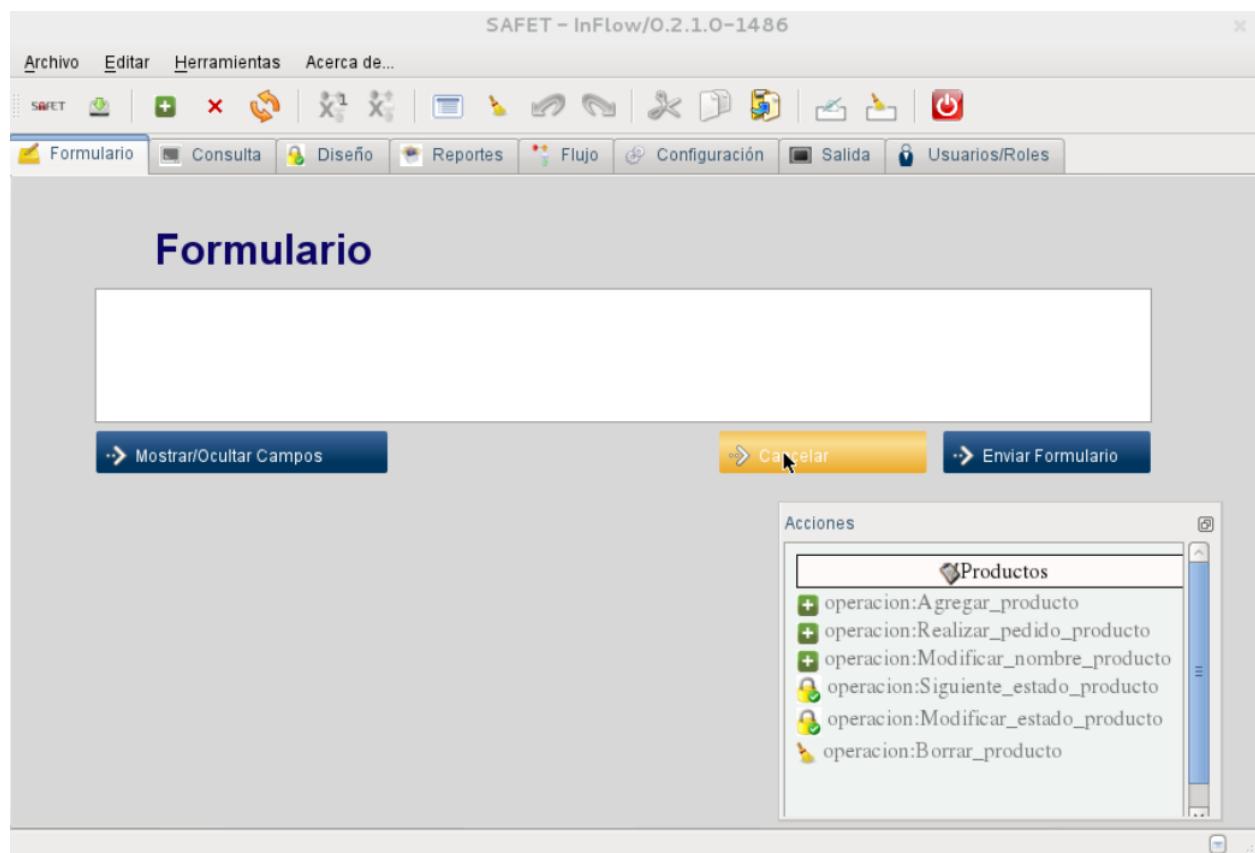


Figura 7.74: Figura 144: Siguiente operación

## 7.2.7 6° SEXTO OPERACIÓN CRUD+ (Borrar un producto)

### 1° PRIMER PASO

- Damos click a la operación (**operacion:Borrar\_producto**), como se muestra en la siguiente *Figura 145: Borrar\_producto*

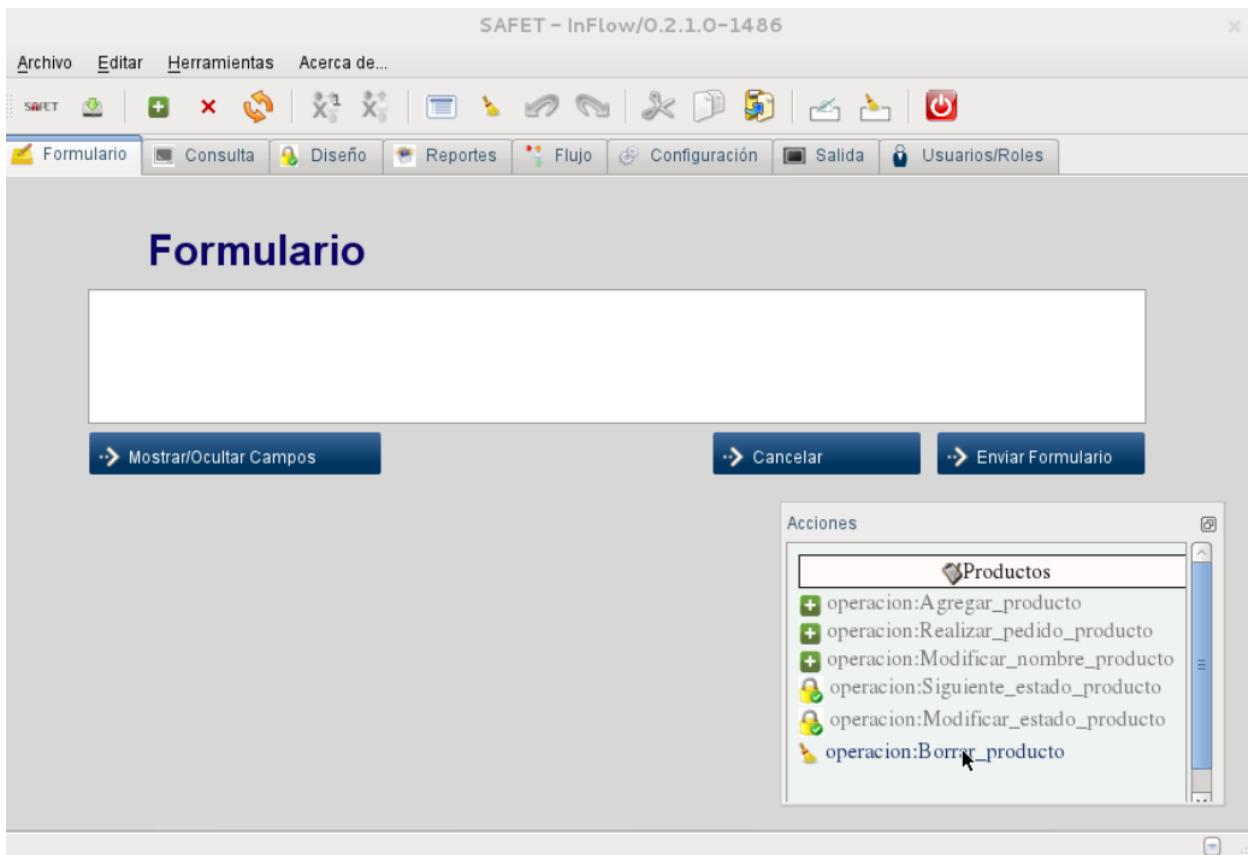


Figura 7.75: **Figura 145: Borrar\_producto**

### 2° SEGUNDO PASO

- En esta (**operacion:Borrar\_producto**) tenemos un sola campo llamado (**id**), la cual damos un click al ese campo (**id**) para seleccionar el productos a eliminar, como se muestra en la siguiente *Figura 146: Campo (id)*

### 3° TERCER PASO

- Seleccionamos el producto a eliminar, como se muestra en la siguiente *Figura 147: Producto a eliminar*

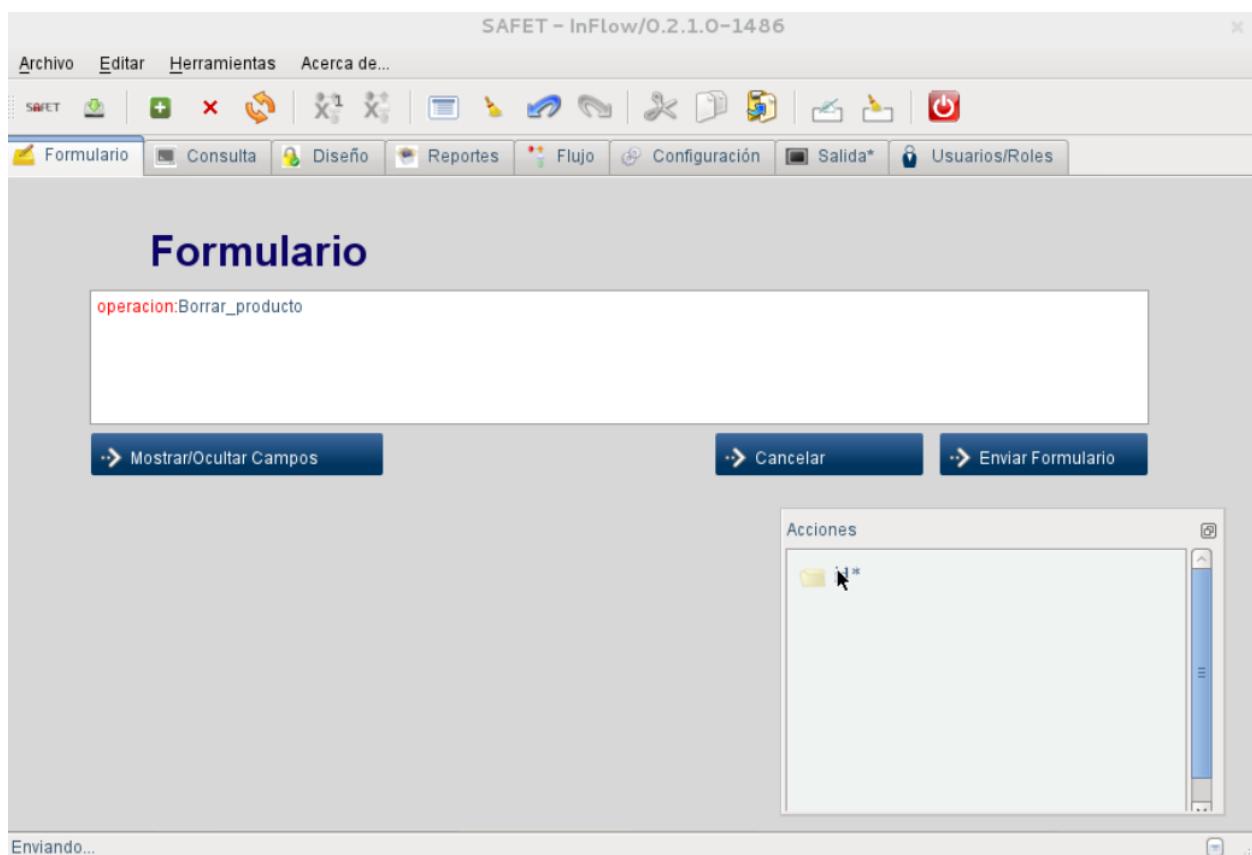


Figura 7.76: **Figura 146: Campo (id)**

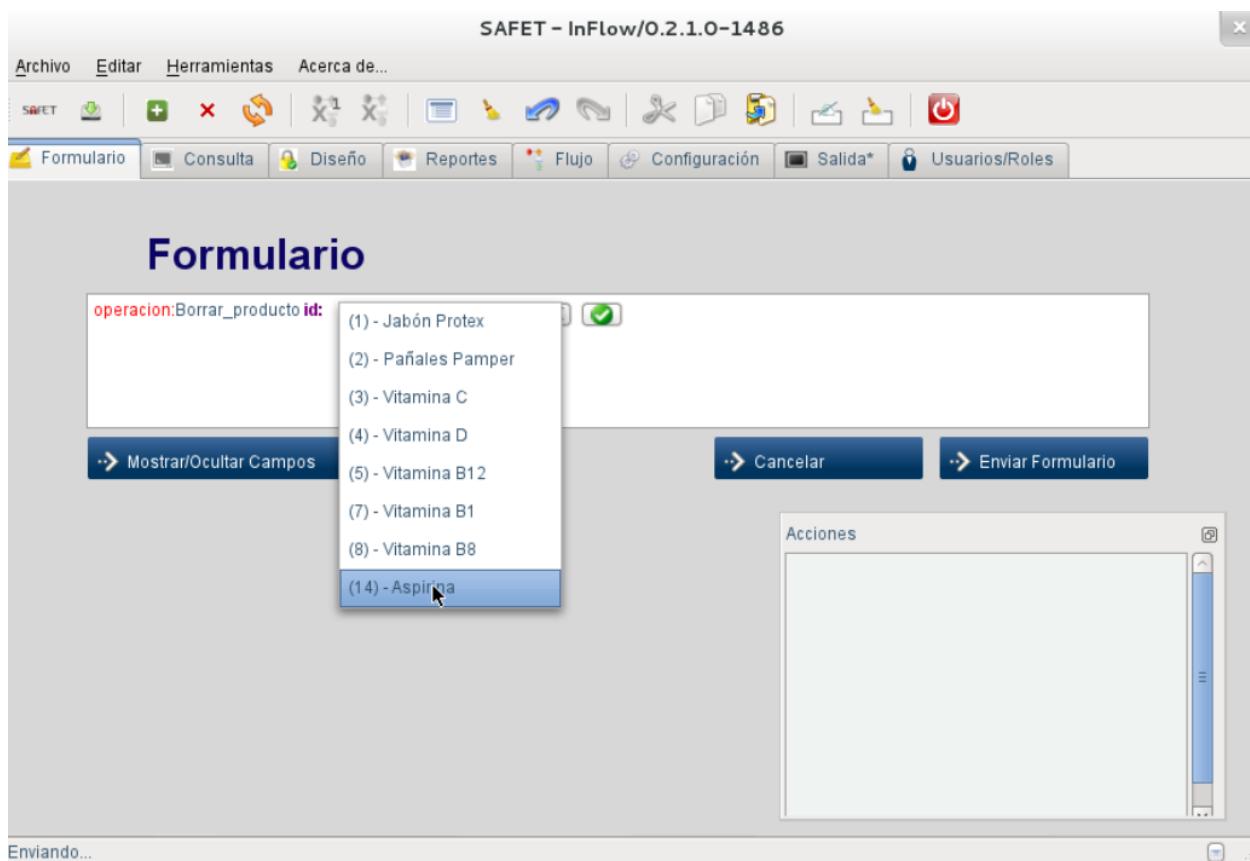


Figura 7.77: Figura 147: Producto a eliminar

## 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 148: Botón*

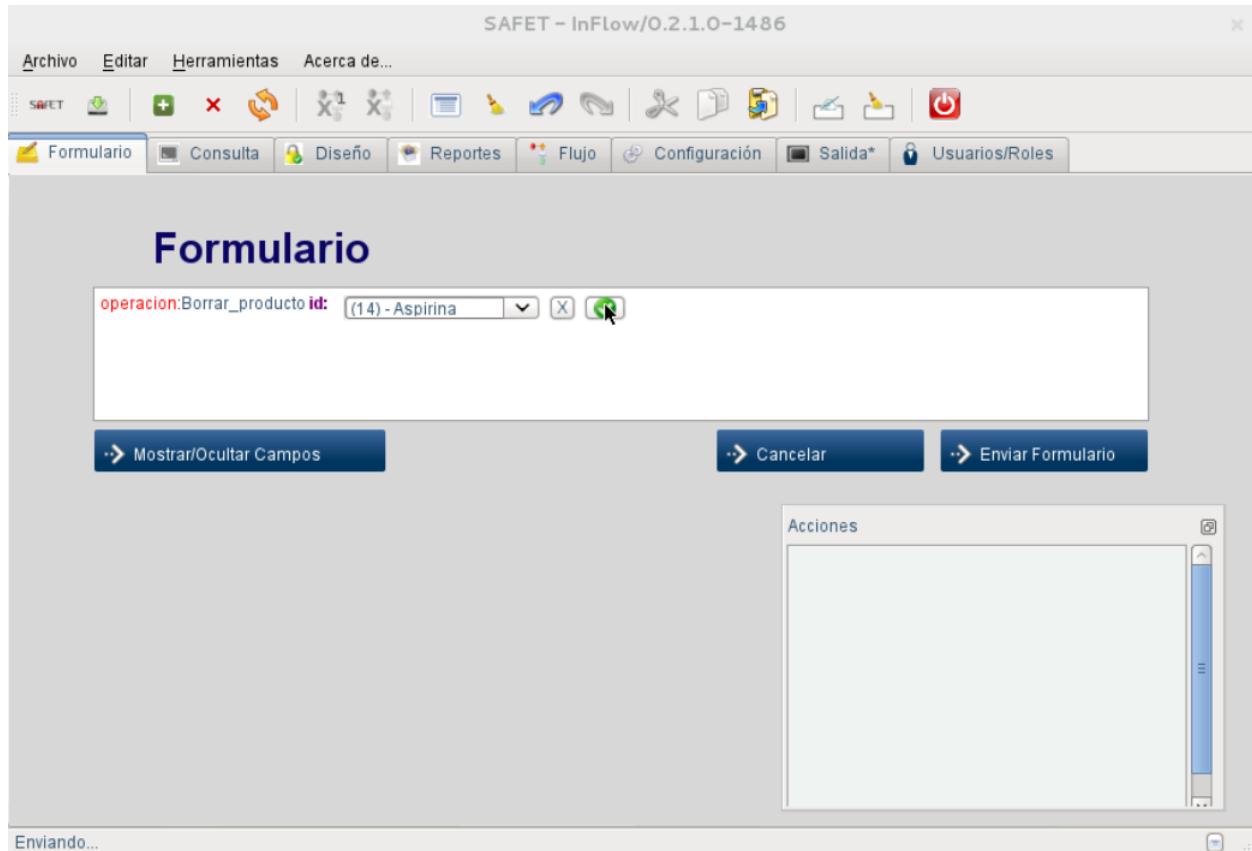


Figura 7.78: **Figura 148: Botón**

## 5° QUINTO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**La operación fue exitosa....ok!**), como se muestra en la siguiente *Figura 149: Resultado de la operación*

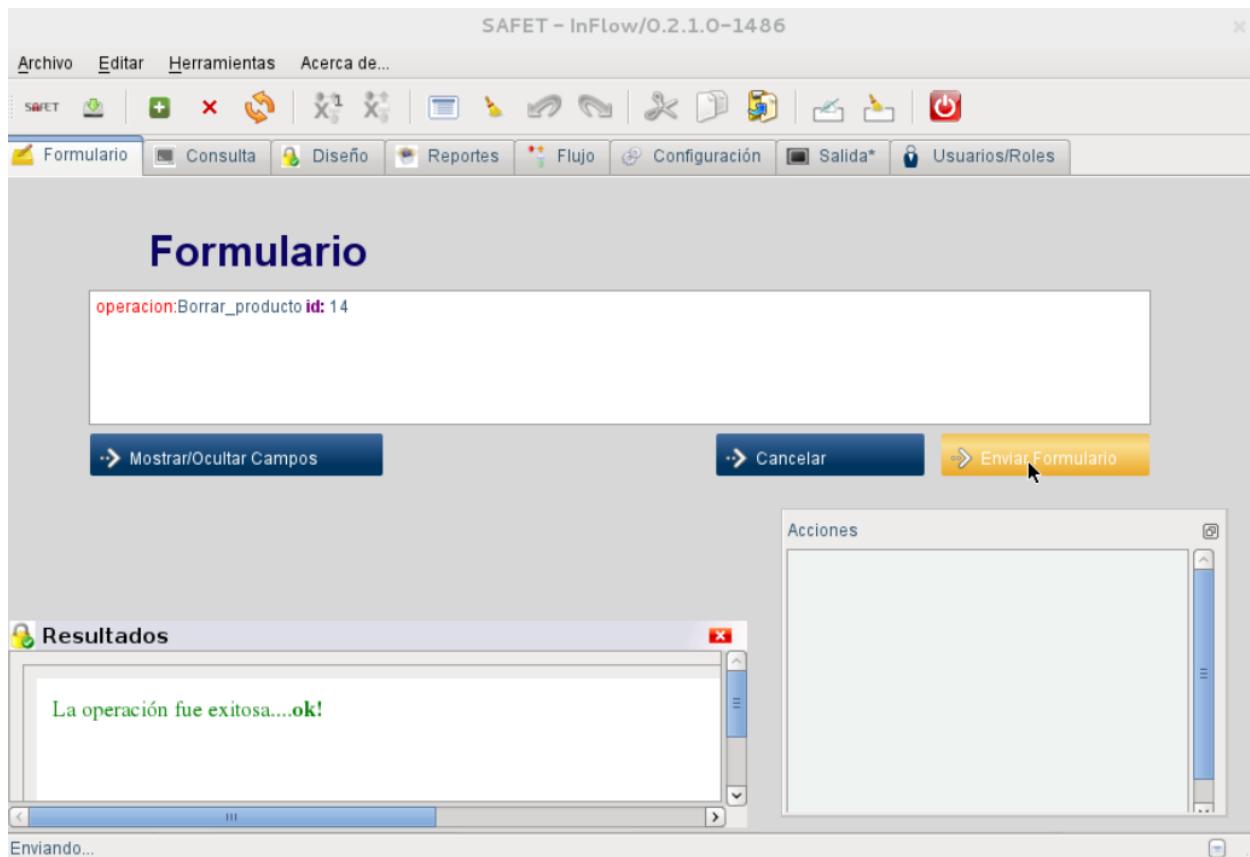


Figura 7.79: Figura 149: Resultado de la operación

## 6° SEXTO PASO

- Para pasar a la siguiente operación damos un click al botón **cancelar** como se muestra en la siguiente *Figura 150: Siguiete operación*

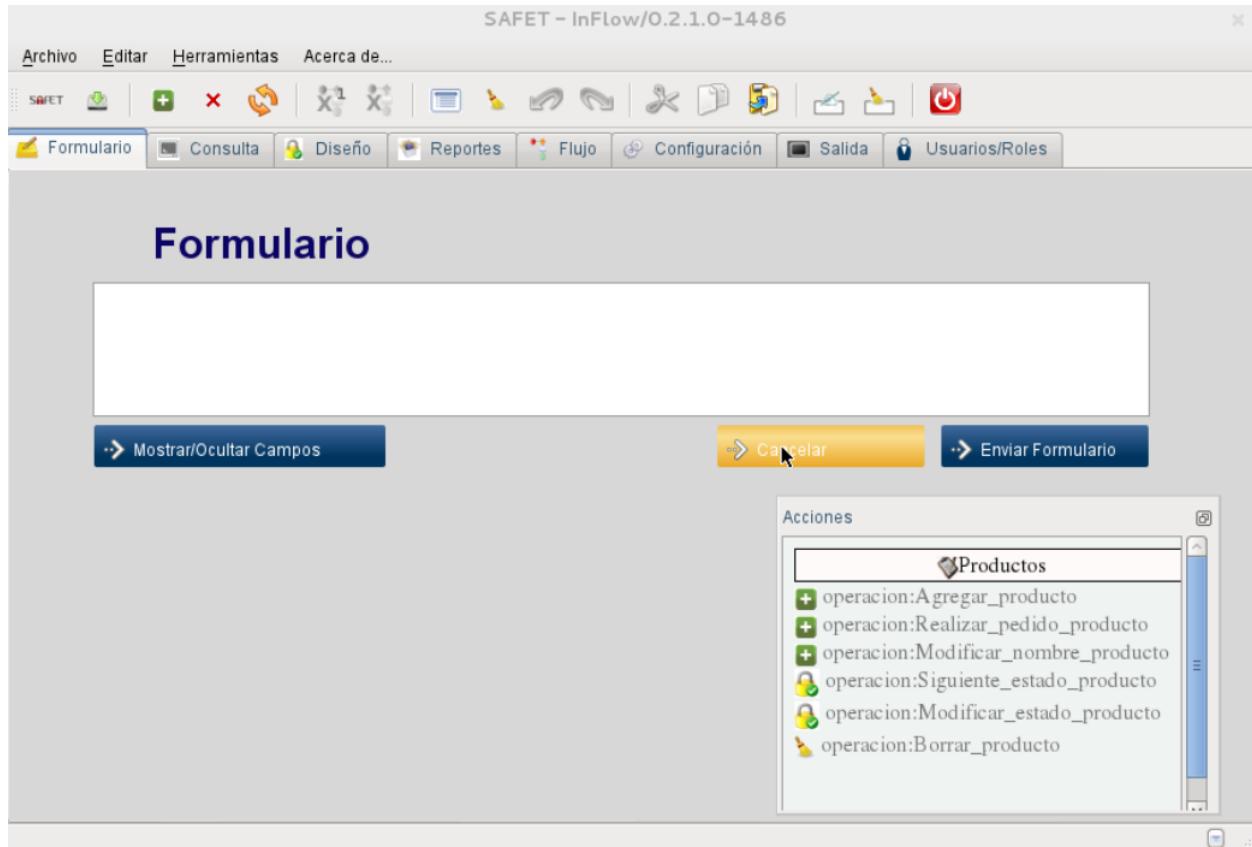


Figura 7.80: **Figura 150: Siguiete operación**

## 7.3 Ejecución de listados de reportes normal

En este tutorial explicaremos el **ejemplo de inventario** utilizando el listado de reportes de datos, para ello debemos tener instalado inflow, sino damos un click al siguiente enlace. Instalación de la interfaz gráfica (inflow).

A continuación seguimos los siguientes pasos para abrir inflow y obtener el listado de reportes:

### 7.3.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 151: Autenticación de Inflow*.



Figura 7.81: **Figura 151: Autenticación de Inflow**.

#### 3° TERCER PASO

- Damos click a la segunda opción (**Realizar consultas**), como se muestra en la siguiente *Figura 152: Realizar consultas*

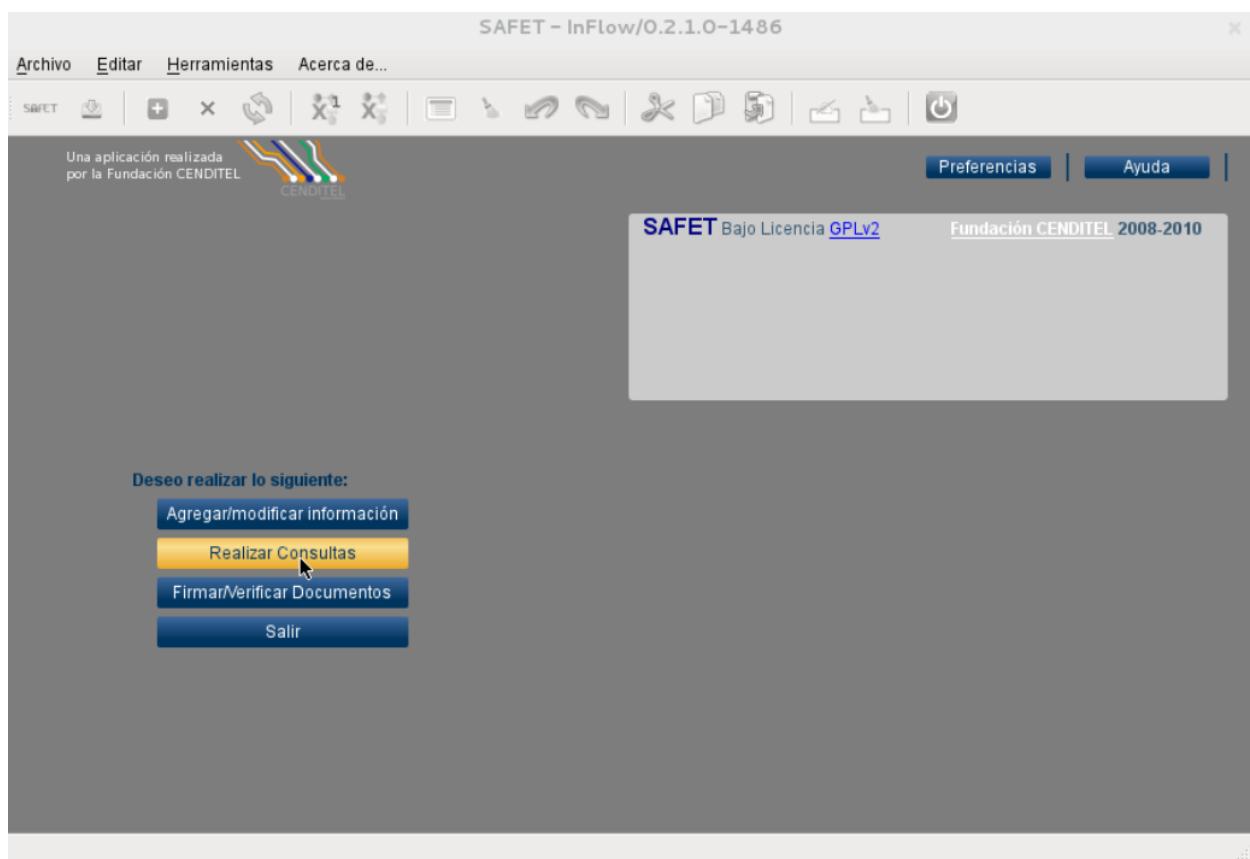


Figura 7.82: **Figura 152: Realizar consultas**

### 4° CUARTO PASO

- Damos click a la opción (**Mostrar/Ocultar Campos**), la cual se nos mostrara las acciones la **Opciones de listados**, como se muestra en la siguiente *Figura 153: Acciones de reportes*

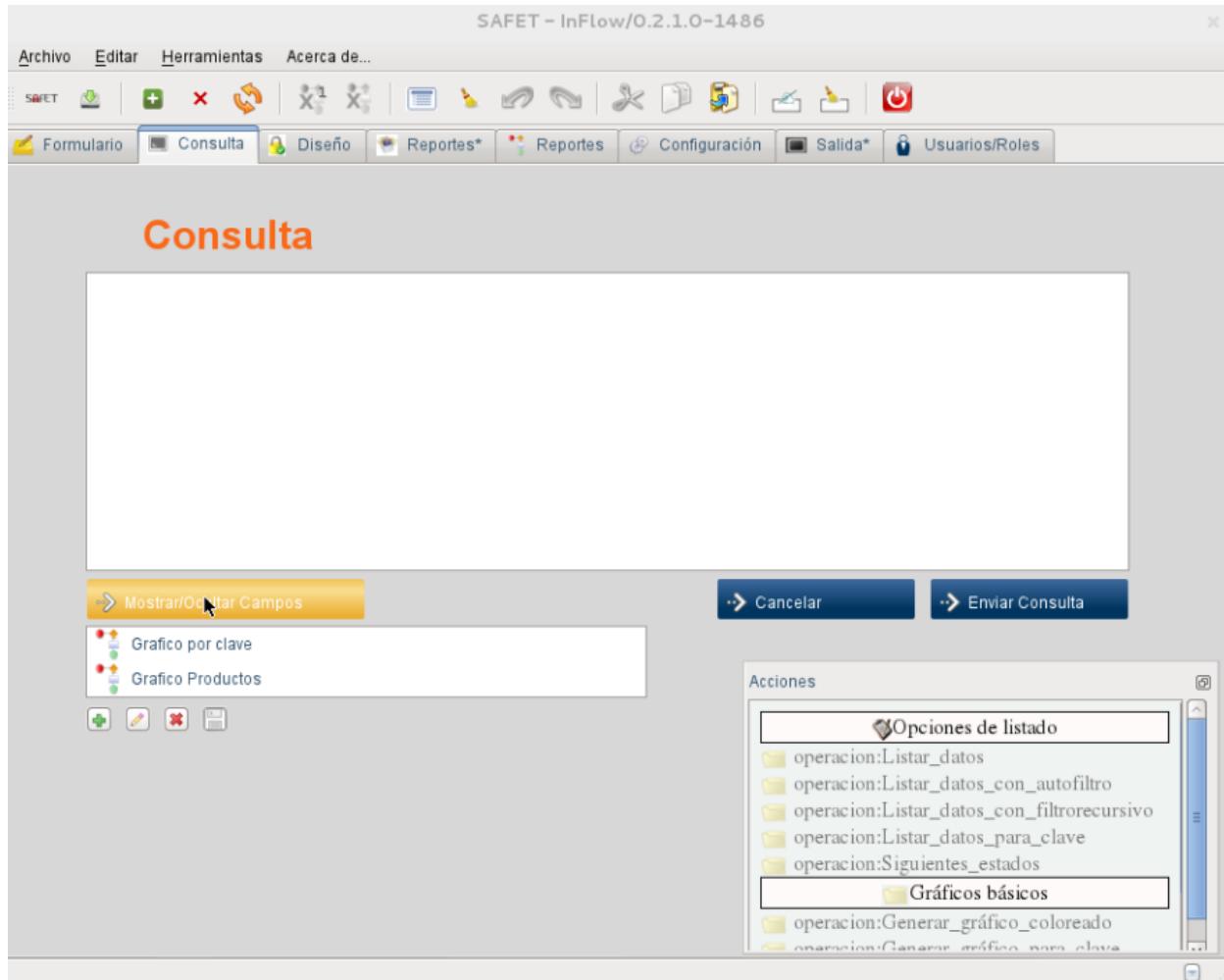


Figura 7.83: **Figura 153: Acciones de reportes**

### 7.3.2 B.- Listar datos normal

#### 1° PRIMER PASO

- Damos click a la operación (**operacion>Listar\_datos**), como se muestra en la siguiente *Figura 154: Listar datos*

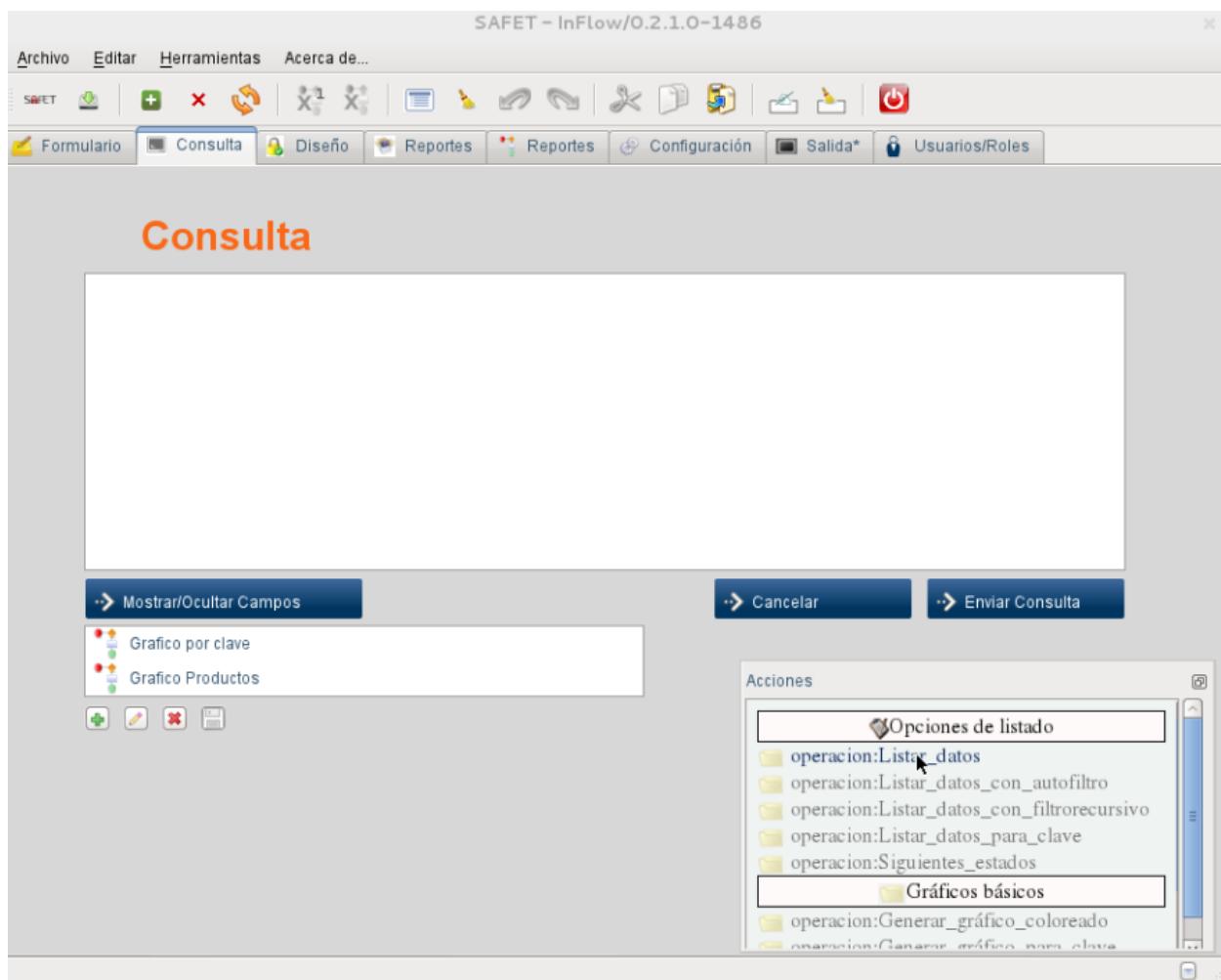


Figura 7.84: **Figura 154: Listar datos**

### 2° SEGUNDO PASO

- Damos click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 155: Campo (Cargar\_archivo\_flujo\*)*

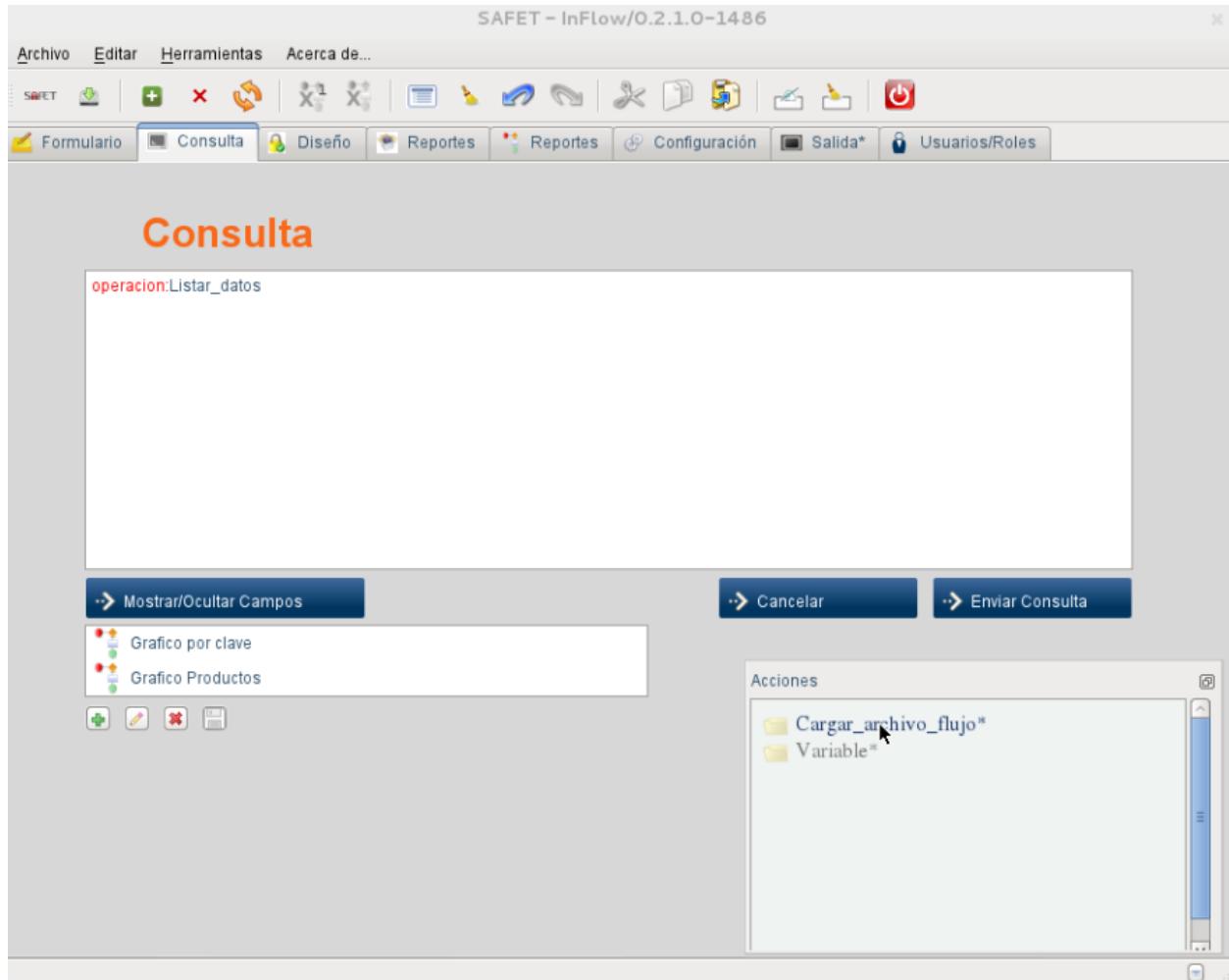


Figura 7.85: **Figura 155: Campo (Cargar\_archivo\_flujo\*)**

### 3° TERCER PASO

- Seleccionamos el archivo **productos.xml** que se encuentra en el directorio <HOME>.safet/flowfiles/, como se muestra en la siguiente *Figura 156: Archivo (xml)*

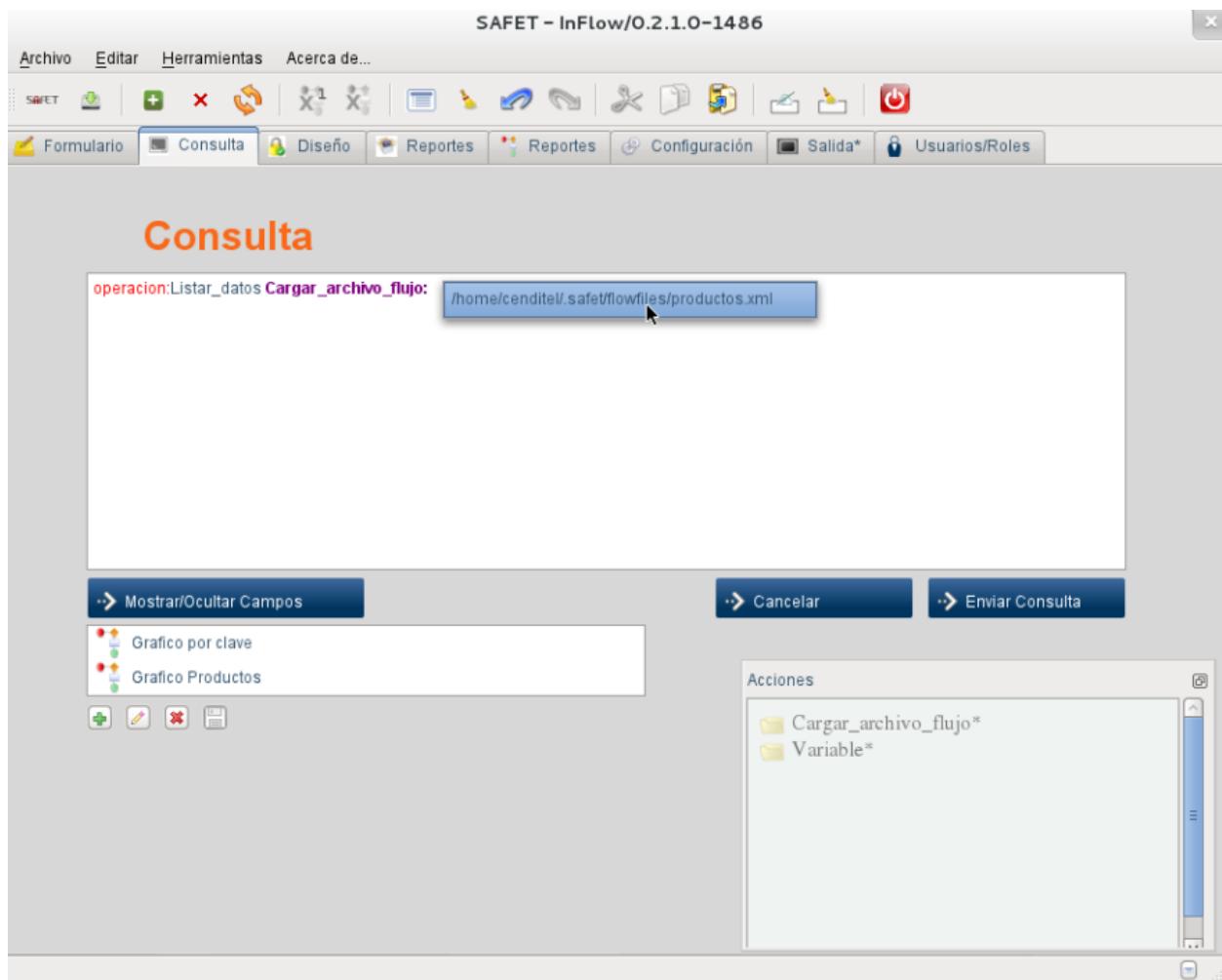


Figura 7.86: **Figura 156: Archivo (xml)**

### 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 157: Botón*

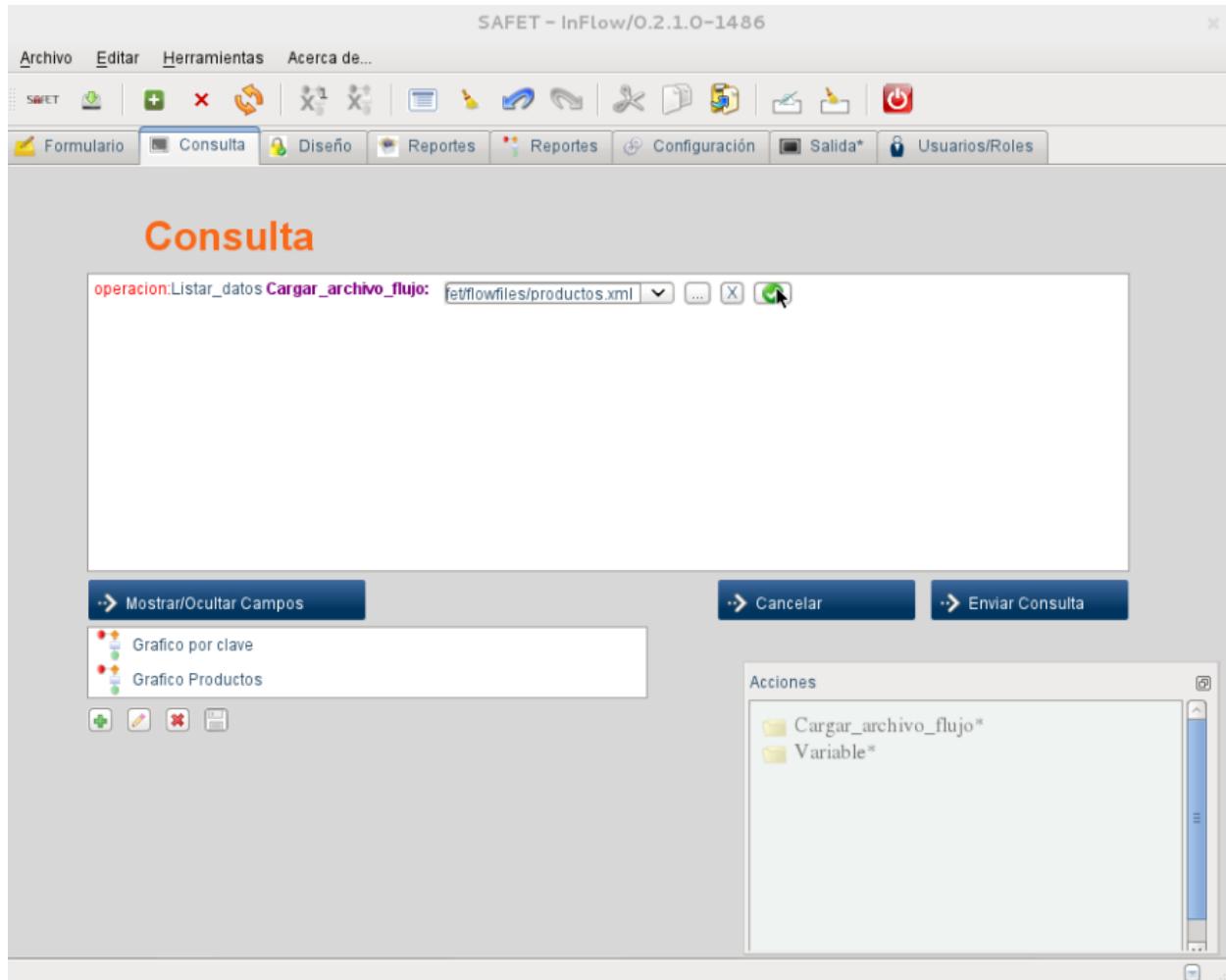


Figura 7.87: **Figura 157: Botón**

### 5° QUINTO PASO

- Damos un click al campo obligatorio (**Variable\***) para indicarle la variable a consultar, como se muestra en la siguiente *Figura 158: Campo (Variable\*)*

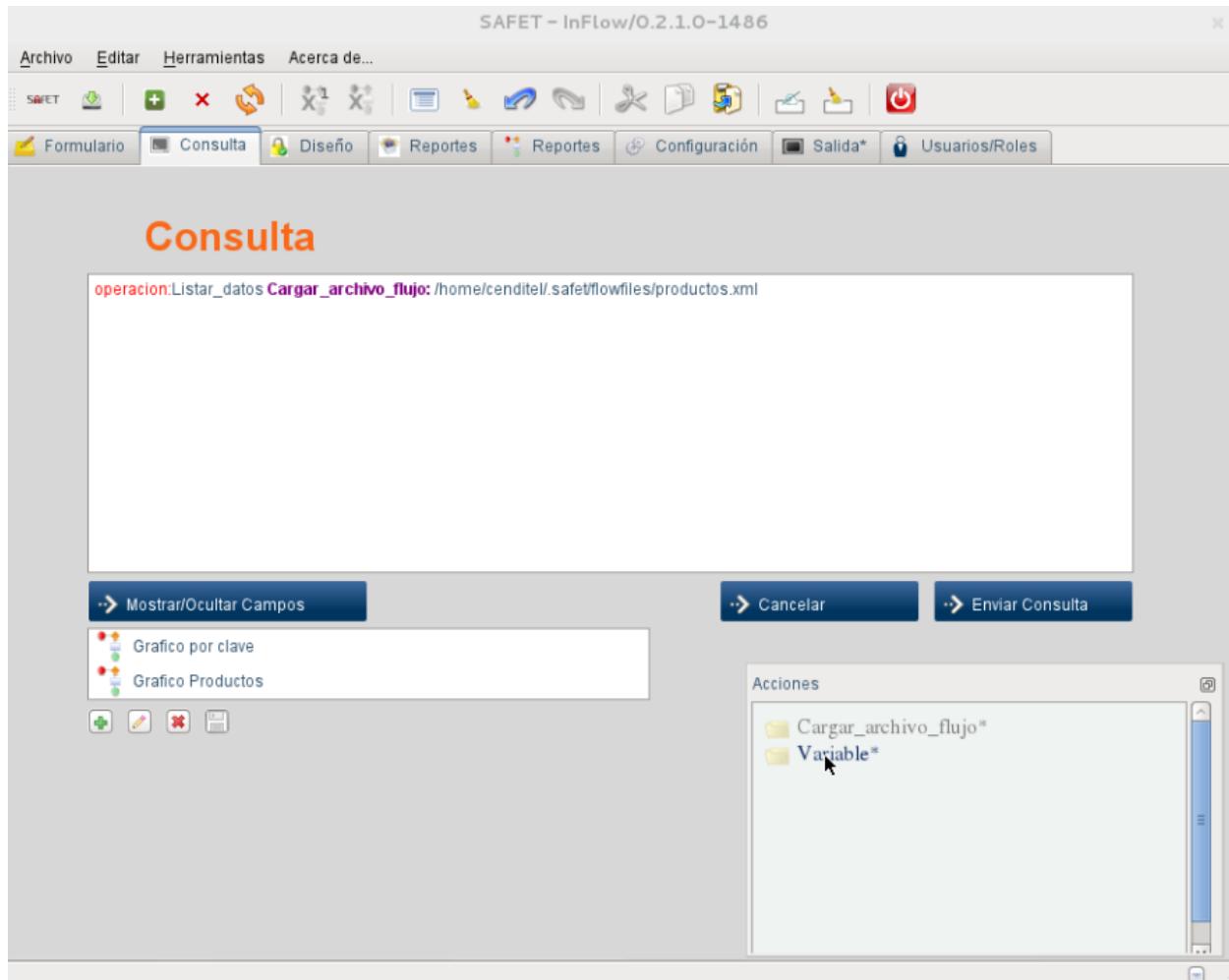


Figura 7.88: **Figura 158: Campo (Variable\*)**

### 6° SEXTO PASO

- Seleccionamos la variable a consultar por ejemplo **vDisponible**, significa que va a mostrar los productos que están disponibles, como se muestra en la siguiente *Figura 159: Variable a consultar*

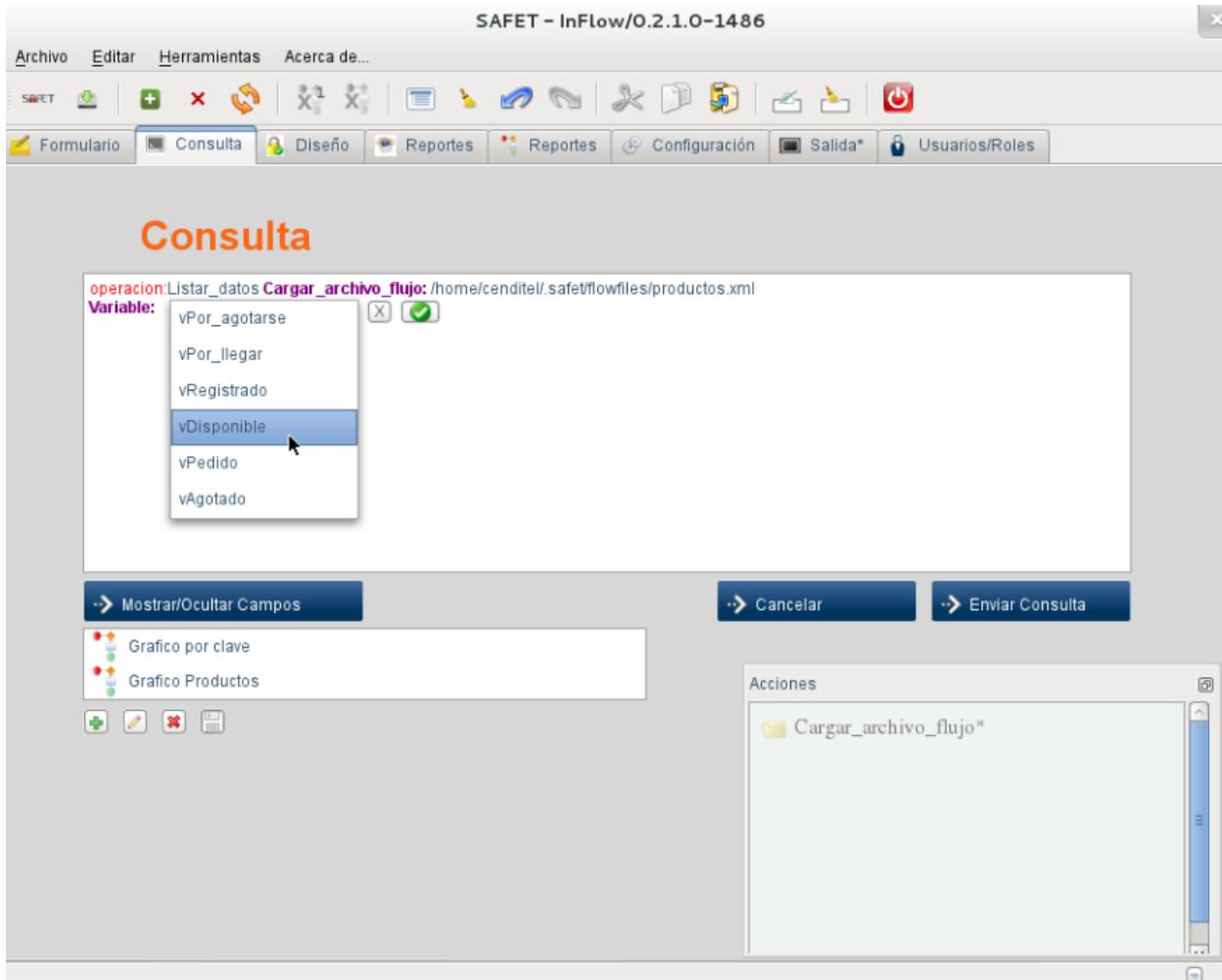


Figura 7.89: *Figura 159: Variable a consultar*

### 7° SEPTIMO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 160: Botón*

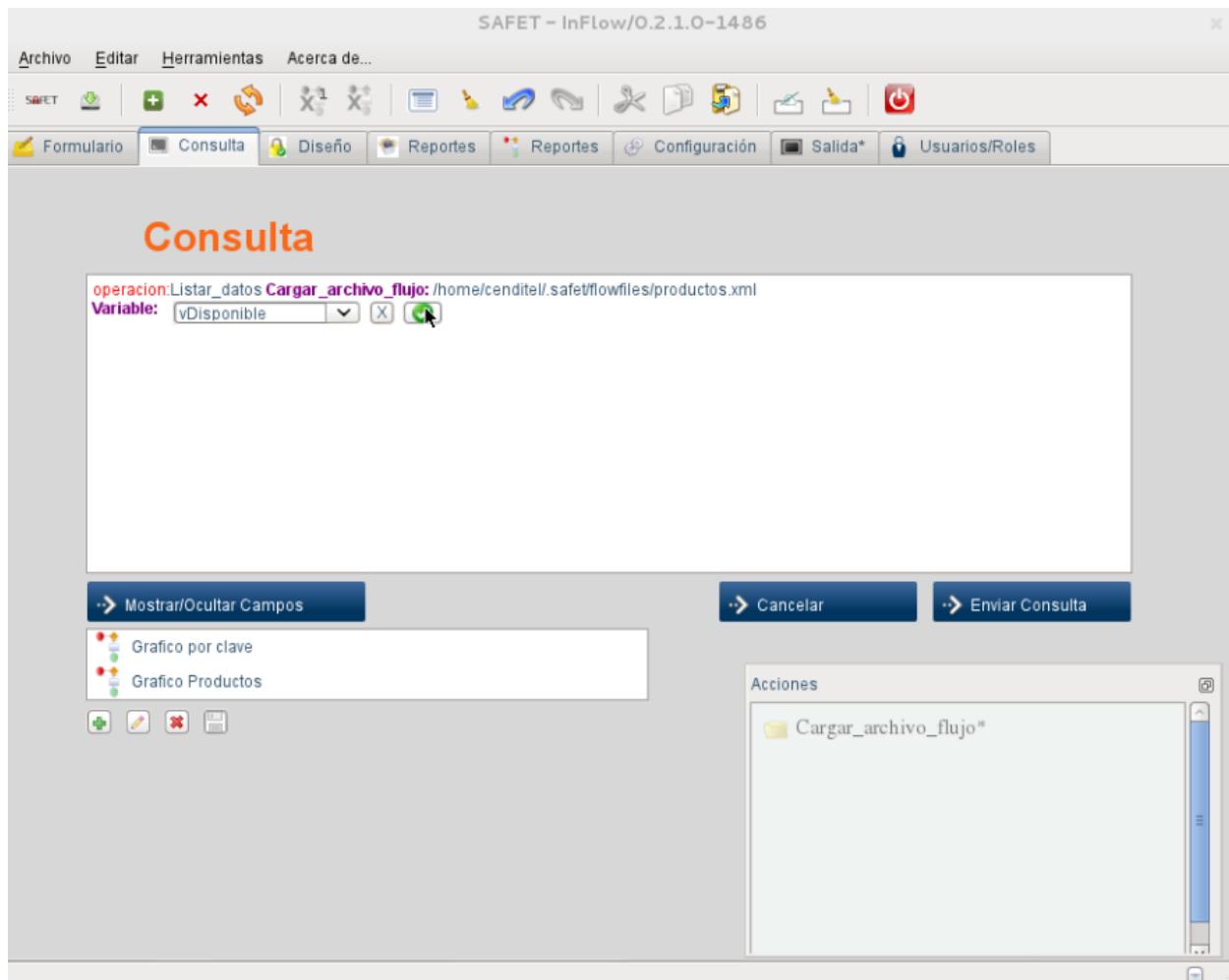


Figura 7.90: Figura 160: Botón

### 8° OCTAVO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**Consulta fue exitosa....ok!** (*Ver reporte*)), como se muestra en la siguiente *Figura 161: Fin de la operación*

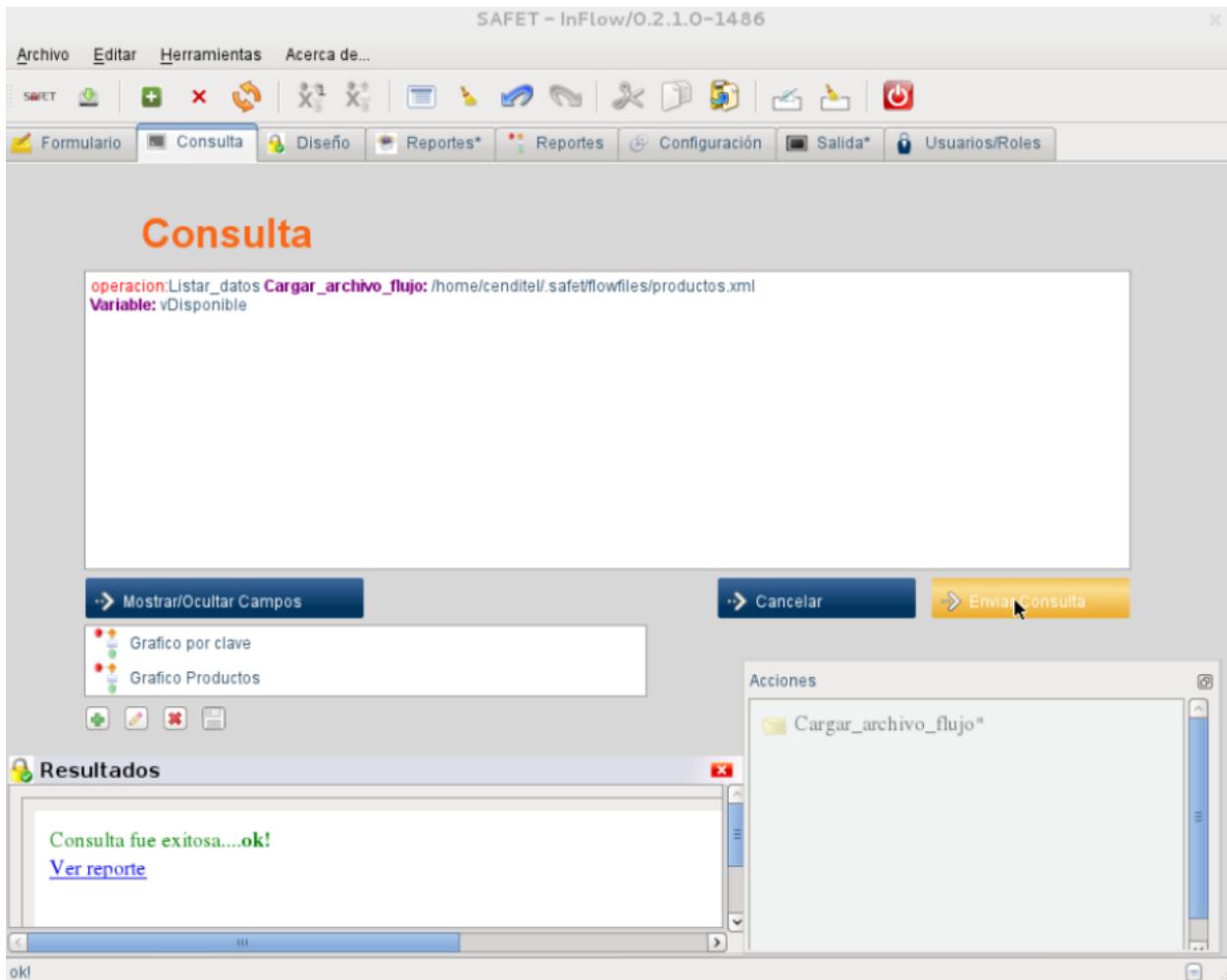


Figura 7.91: **Figura 161: Fin de la operación**

### 9° PASO

- Damos click al resultado (**Ver reporte**) para ver el listado de reporte, como se muestra en la siguiente *Figura 162: Ver reporte*

---

**Nota:** Se nos muestra el reporte de datos *Figura 163: Resultado de la operación*

---

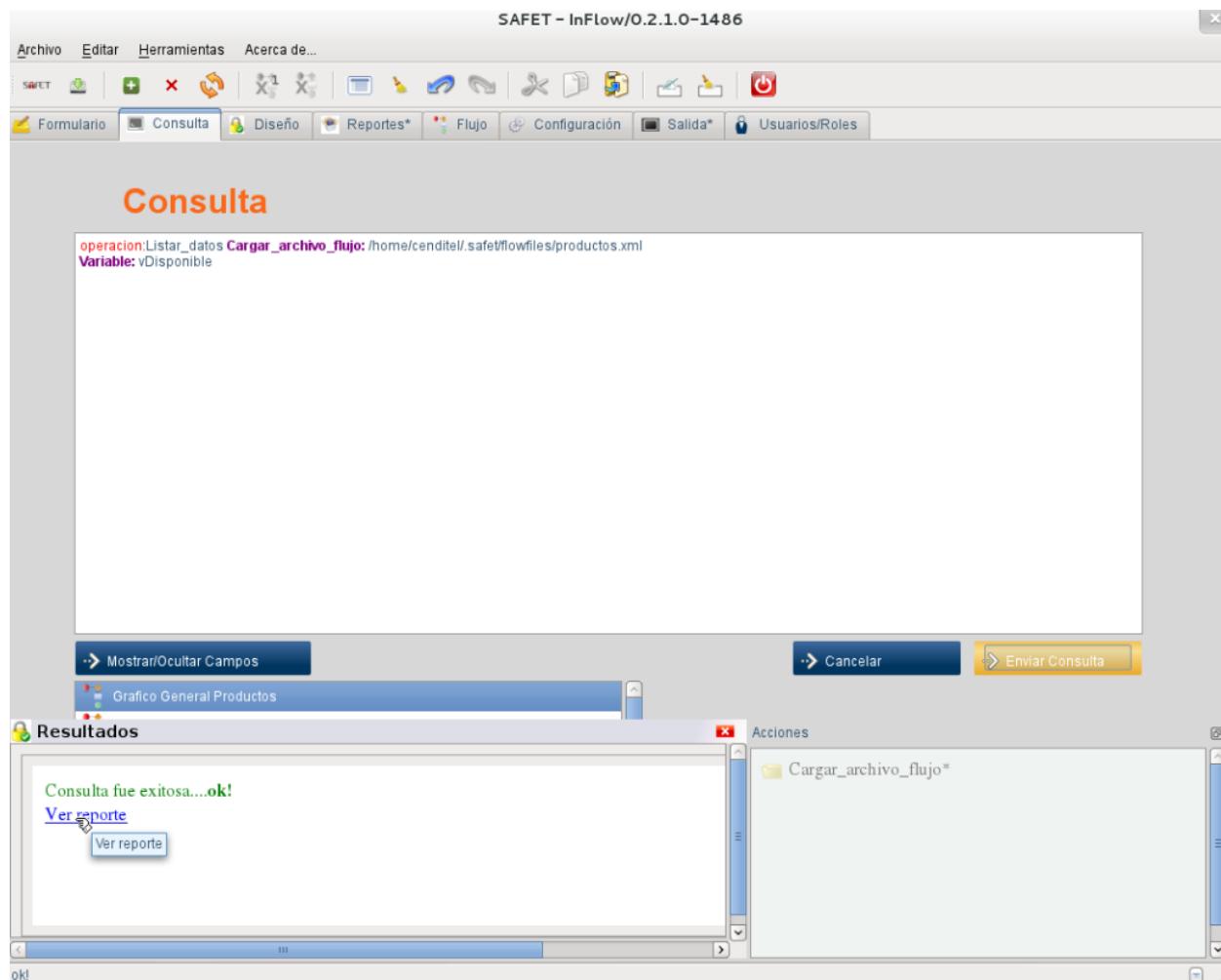


Figura 7.92: Figura 162: Ver reporte

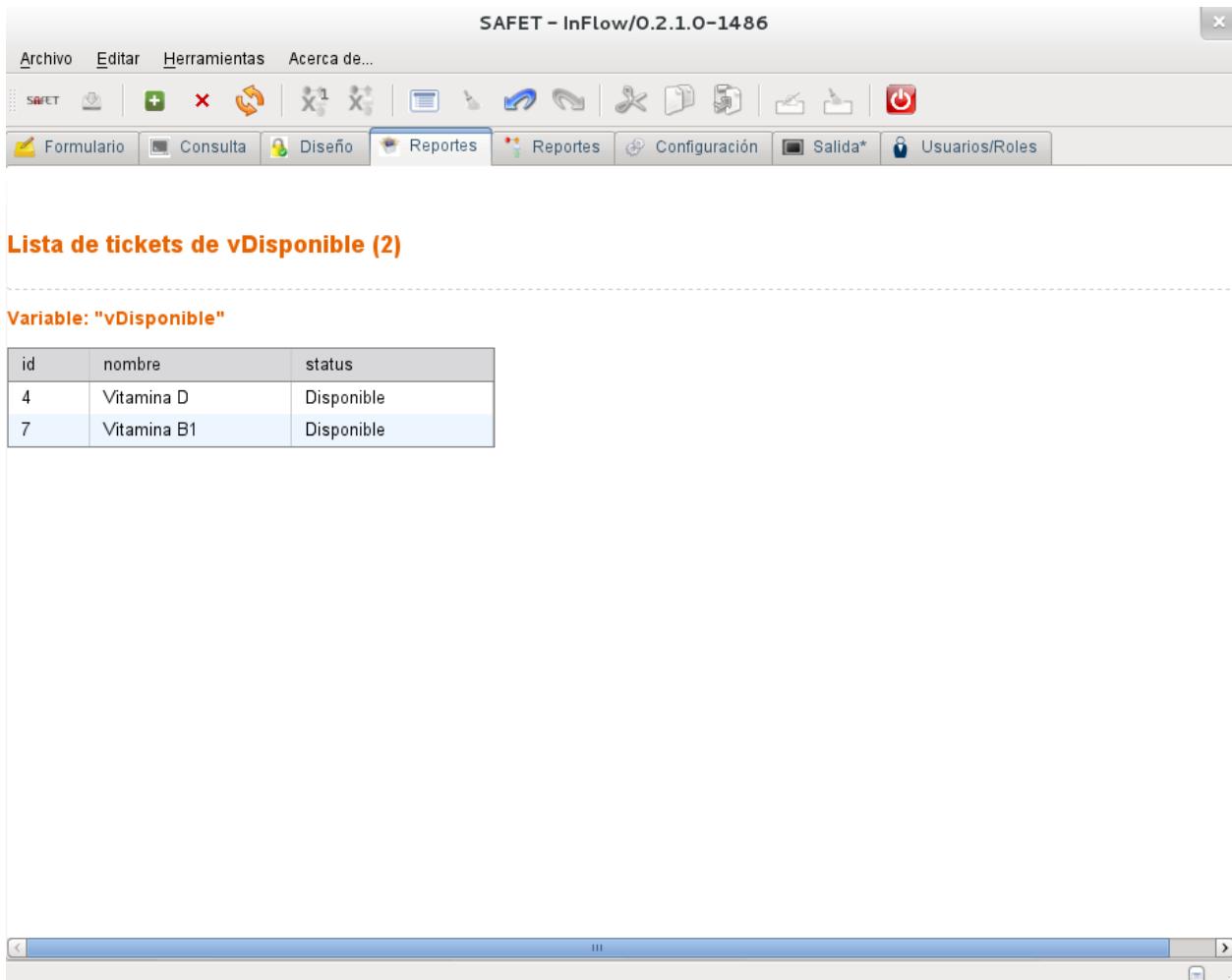


Figura 7.93: Figura 163: Resultado de la operación

## 7.4 Ejecución de listados de reportes con parámetros

En este tutorial explicaremos el **ejemplo de inventario** utilizando el listados de reportes de datos con parámetros, para ello debemos tener instalado inflow, sino damos un click al siguiente enlace. Instalación de la interfaz gráfica (inflow).

**A continuación seguimos los siguientes pasos para abrir inflow y obtener el listado de reportes con parámetros:**

### 7.4.1 A.- Editamos el archivo (productos.xml)

Abrimos el archivo (**productos.xml**) del directorio (**/home/usuario/.safet/floefiles/**) y remplazamos la **condición (inicial)** por el siguiente **código de parámetro** :

```
<!--
*****
/ Condición inicial con el Parámetro (Empieza_por) /
*****
-->

<parameter title="Empieza_por" options="" type="string" mandatory="no" />
<condition type="start" id="inicial">
<port side="forward" type="split">

<connection query="select nombre from productos"
options="LIKE '#Empieza_por'%'" source="Por_nombre"/>

</port>
</condition>

<!--
*****
/ Nueva tarea Buscar (Por_nombre)   /
*****
-->

<task title="" id="Por_nombre">

<port side="forward" type="split">
<connection query="select status from productos"
options="Registrado" source="Registrado"/>
</port>

<variable config="1" documentsource="select nombre,status
from productos" type="sql" tokenlink="" id="vEmpieza" rolfield="(select
rol from productos_registro where productoid=productos.id and
regstatus='Registrado') as rol" scope="task" timestampfield="(select fecha
from productos_registro where productoid=productos.id and
regstatus='Registrado') as fecha"/>

</task>
```

## 7.4.2 B.- Abrimos la interfaz gráfica de inflow

### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 164: Autenticación de Inflow*.



Figura 7.94: **Figura 164: Autenticación de Inflow.**

### 3° TERCER PASO

- Damos click a la segunda opción (**Realizar consultas**), como se muestra en la siguiente *Figura 165: Realizar consultas*

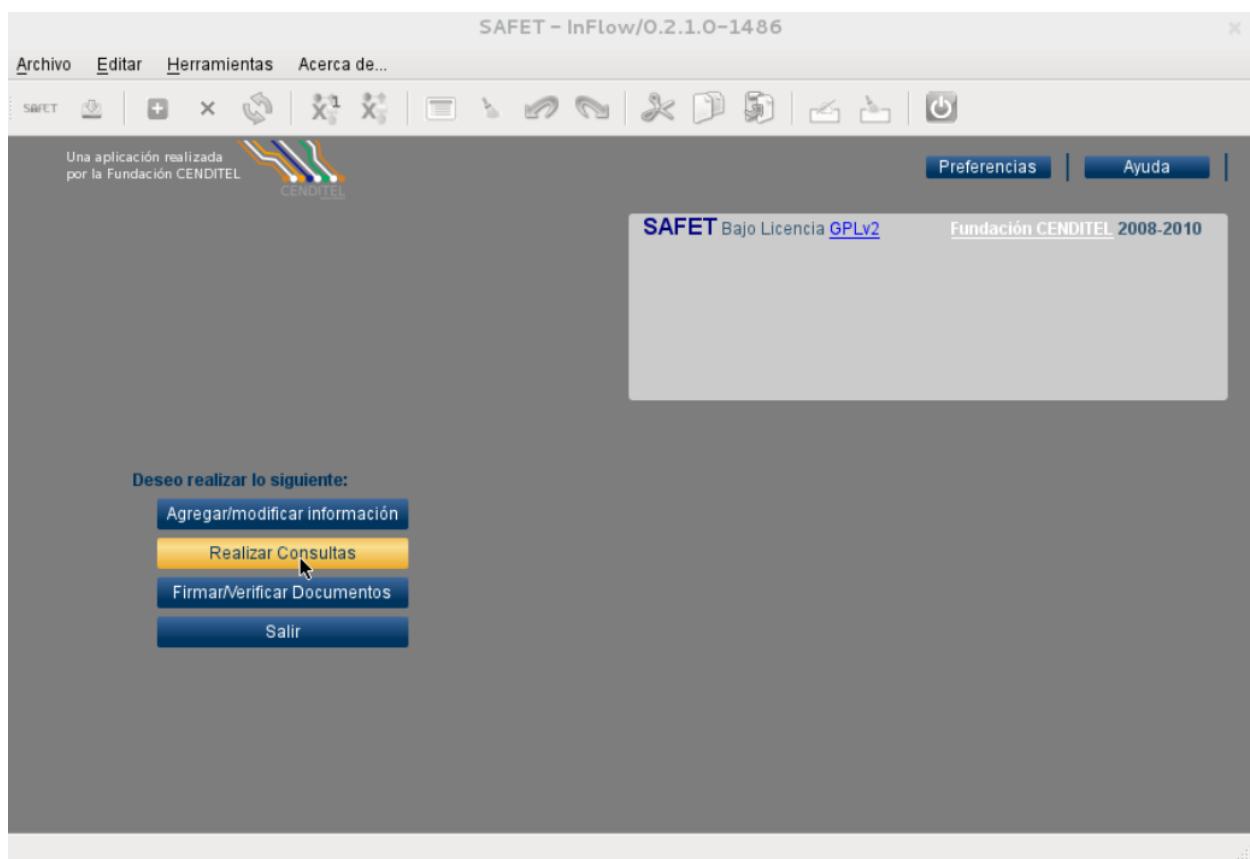


Figura 7.95: **Figura 165: Realizar consultas**

### 4° CUARTO PASO

- Damos click a la opción (**Mostrar/Ocultar Campos**), la cual se nos mostrara las acciones la **Opciones de listados**, como se muestra en la siguiente *Figura 166: Acciones de reportes*

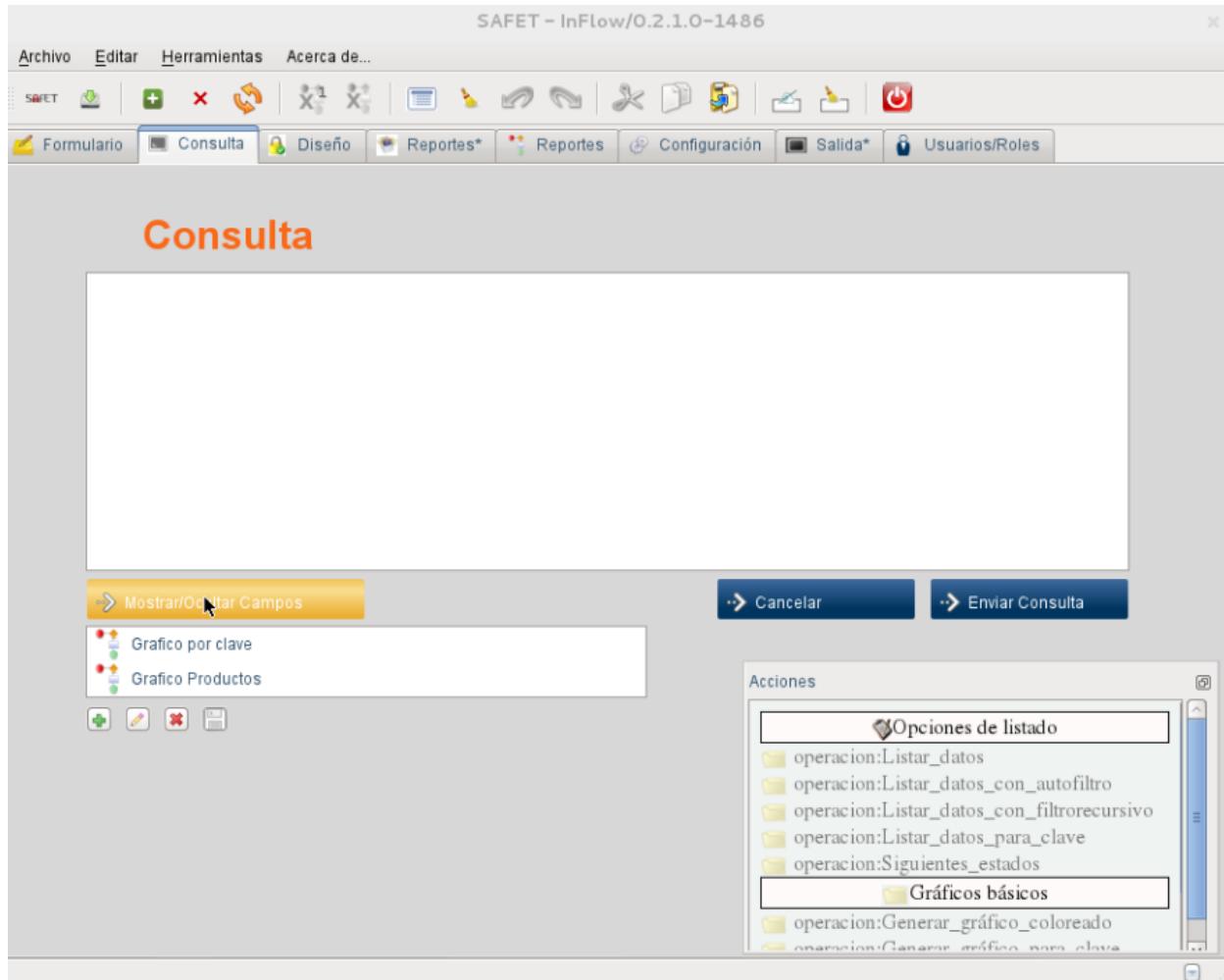


Figura 7.96: **Figura 166: Acciones de reportes**

### 7.4.3 C.- Listar los datos con parámetros

#### 1° PRIMER PASO

- Damos click a la operación (**operacion>Listar\_datos**), como se muestra en la siguiente *Figura 167: Operación (Listar\_datos)*

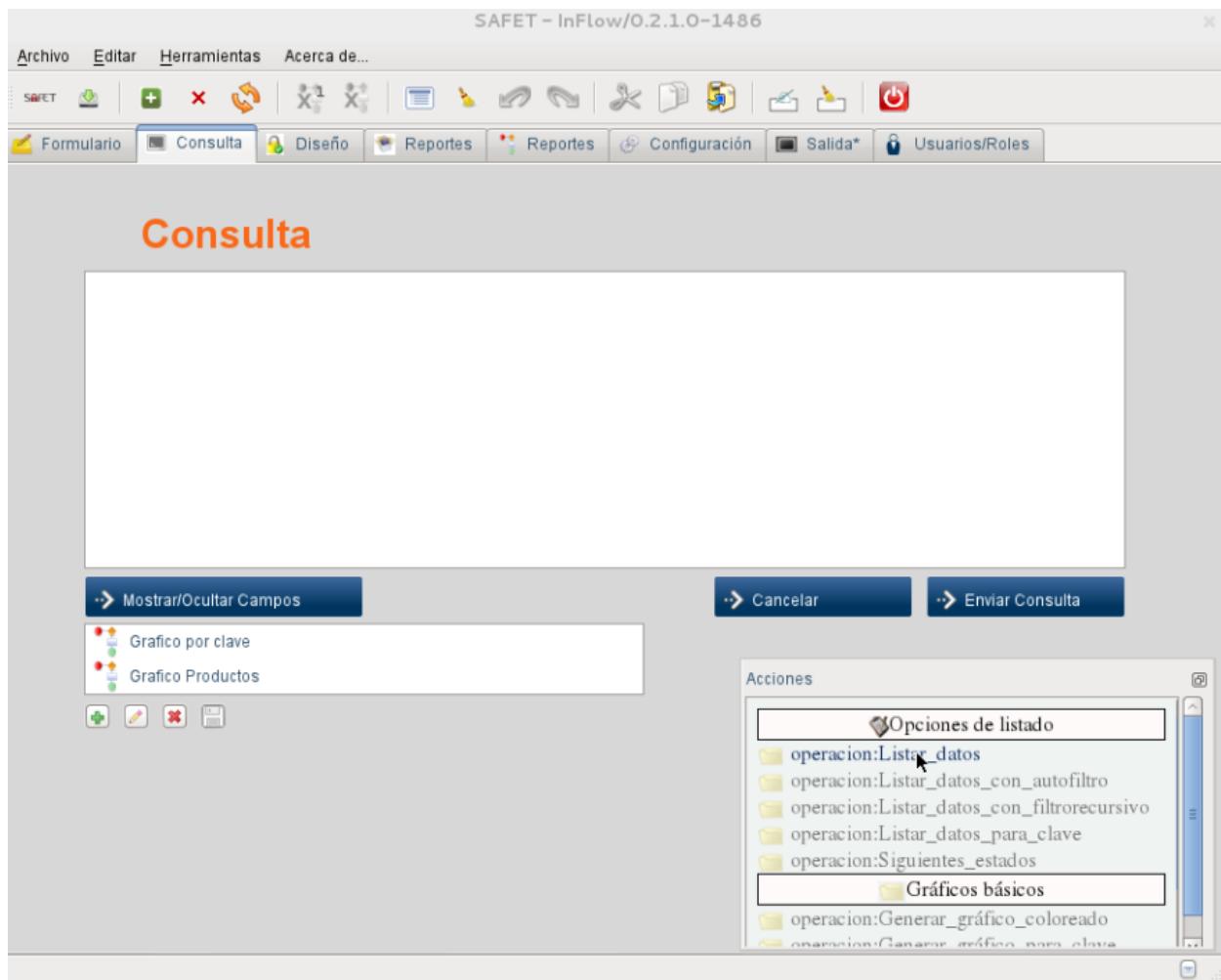


Figura 7.97: Figura 167: Operación (Listar\_datos)

### 2° SEGUNDO PASO

- Damos click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 168: Campo (Cargar\_archivo\_flujo\*)*

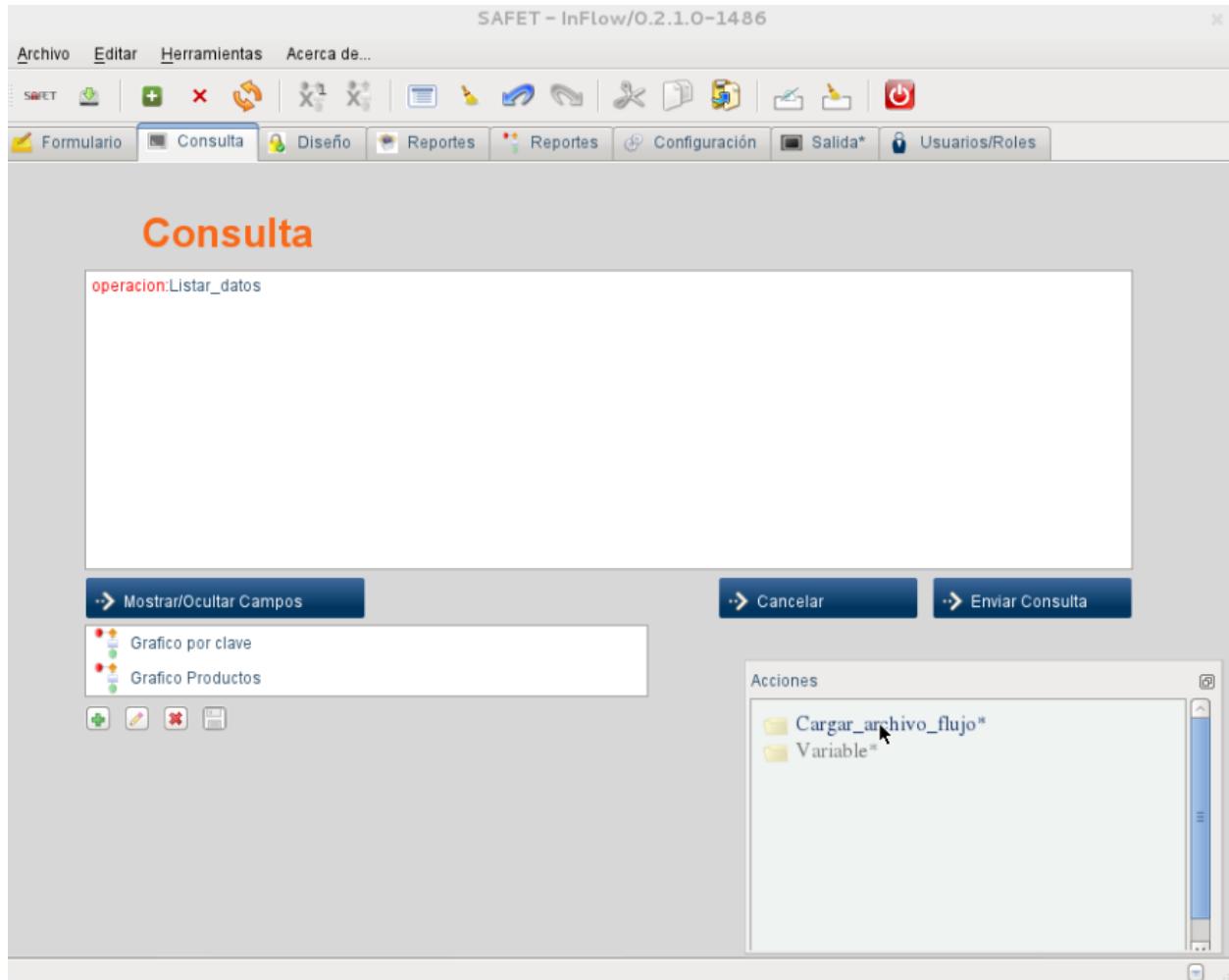


Figura 7.98: **Figura 168: Campo (Cargar\_archivo\_flujo\*)**

### 3° TERCER PASO

- Seleccionamos el archivo **productos.xml** que se encuentra en el directorio <HOME>.safet/flowfiles/, como se muestra en la siguiente *Figura 169: Archivo (xml)*

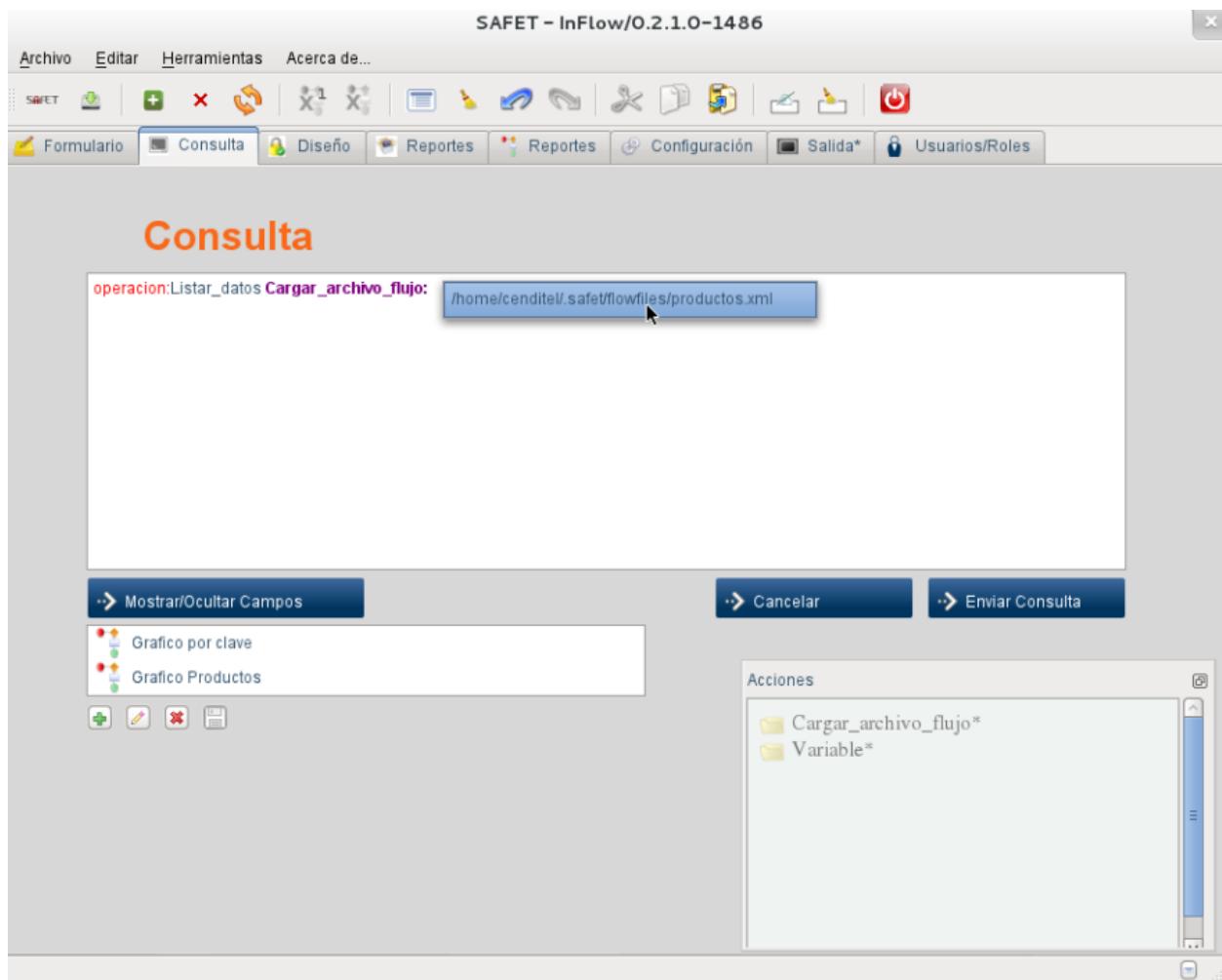


Figura 7.99: **Figura 169: Archivo (xml)**

### 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 170: Botón*

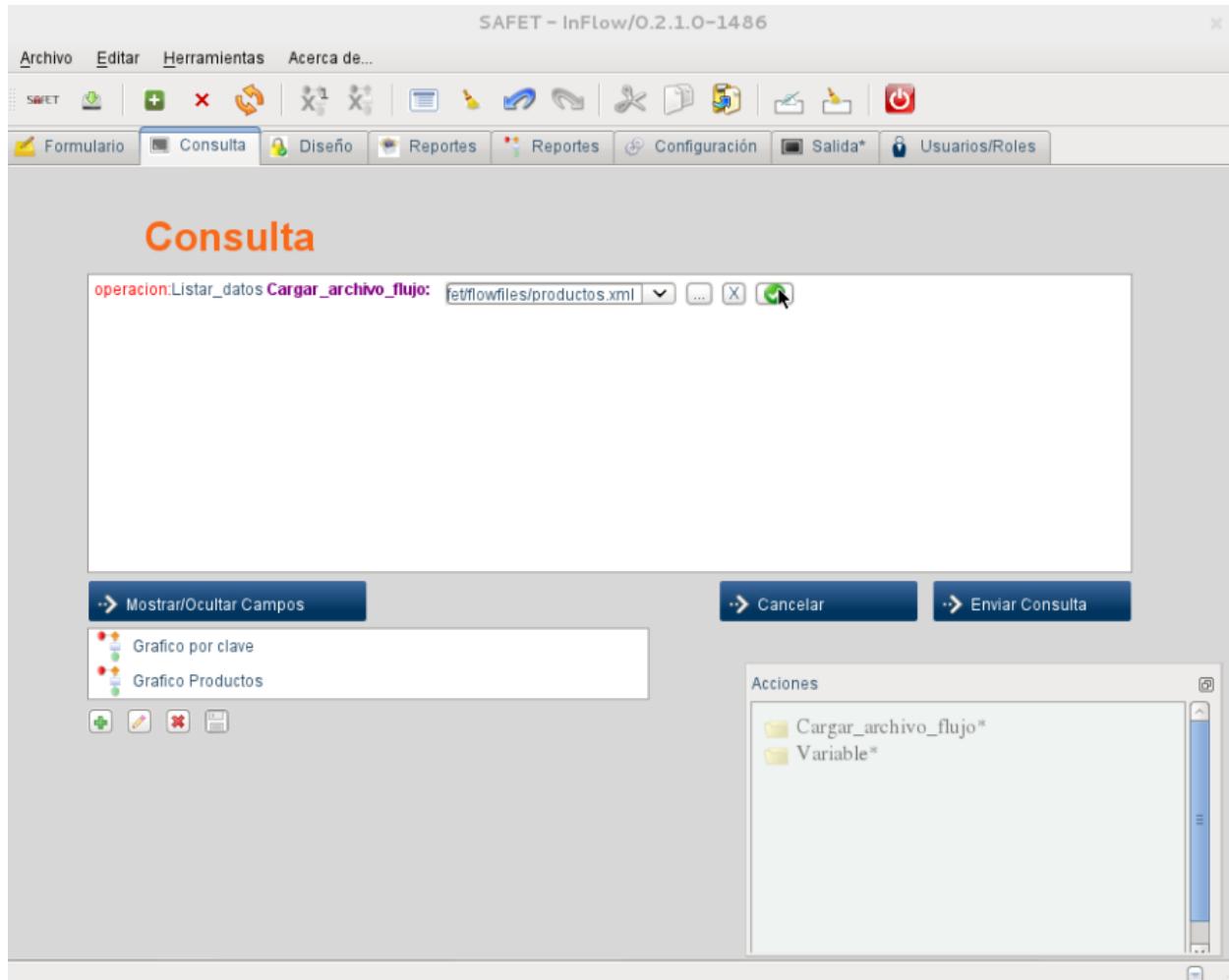


Figura 7.100: **Figura 170: Botón**

### 5° QUINTO PASO

- Damos un click al obligatorio (**Variable\***), como se muestra en la siguiente *Figura 171: Campo (Variable\*)*

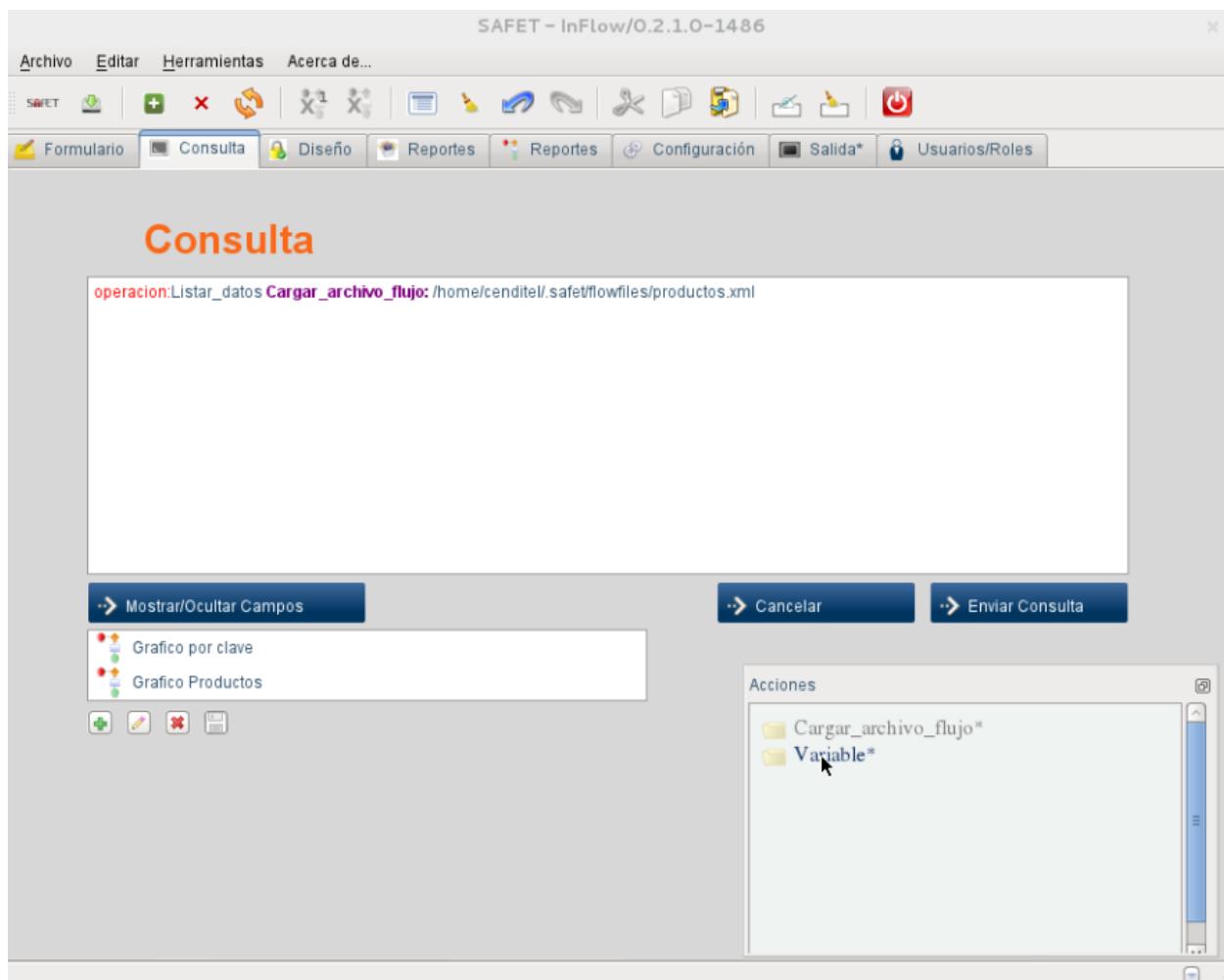


Figura 7.101: Figura 171: Campo (Variable\*)

### 6° SEXTO PASO

- Seleccionamos la variable a consultar por ejemplo **vDisponible**, significa que va a mostrar los productos que estan disponibles, como se muestra en la siguiente *Figura 172: Variable a consultar*

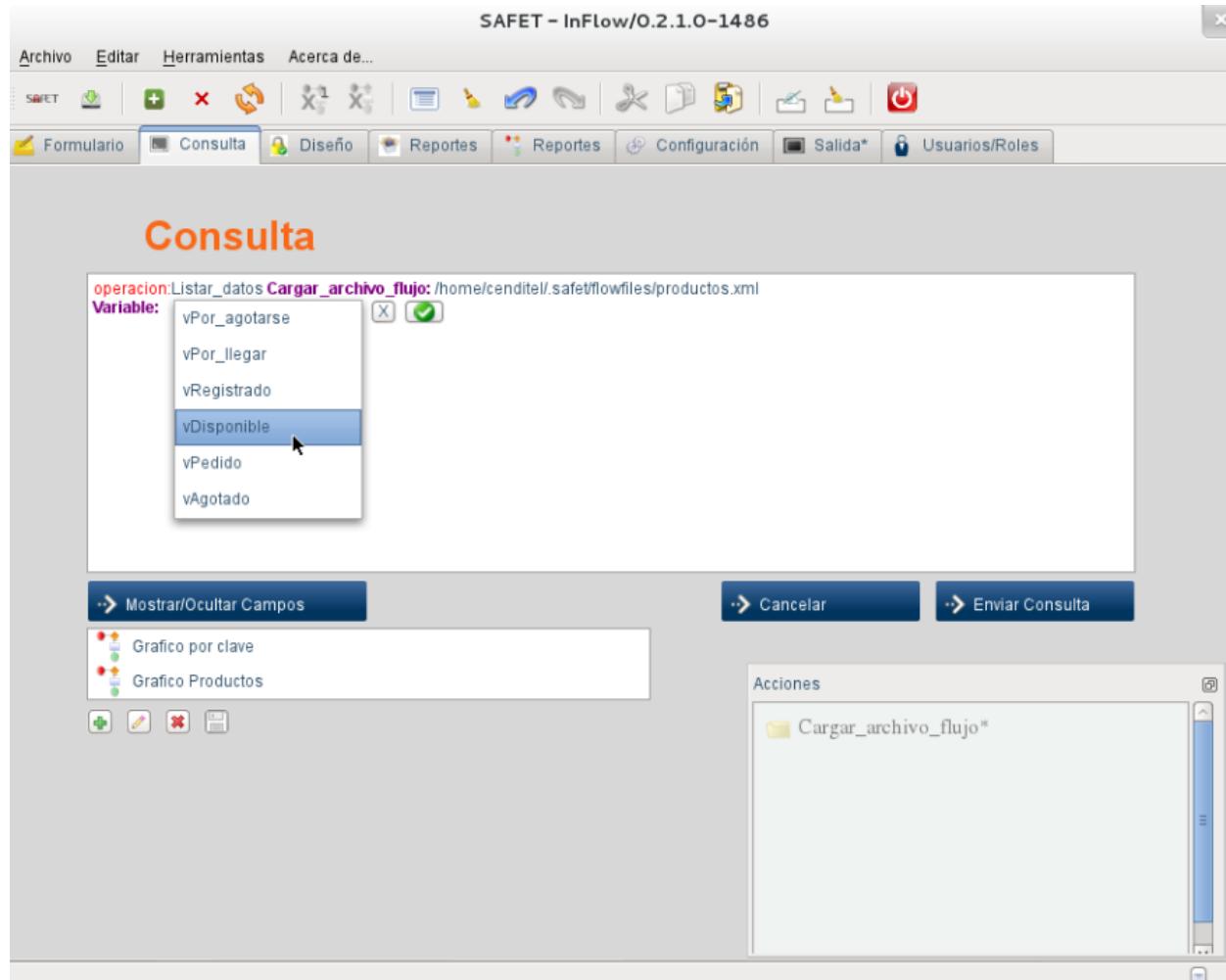


Figura 7.102: **Figura 172: Variable a consultar**

### 7° SEPTIMO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 173: Botón*

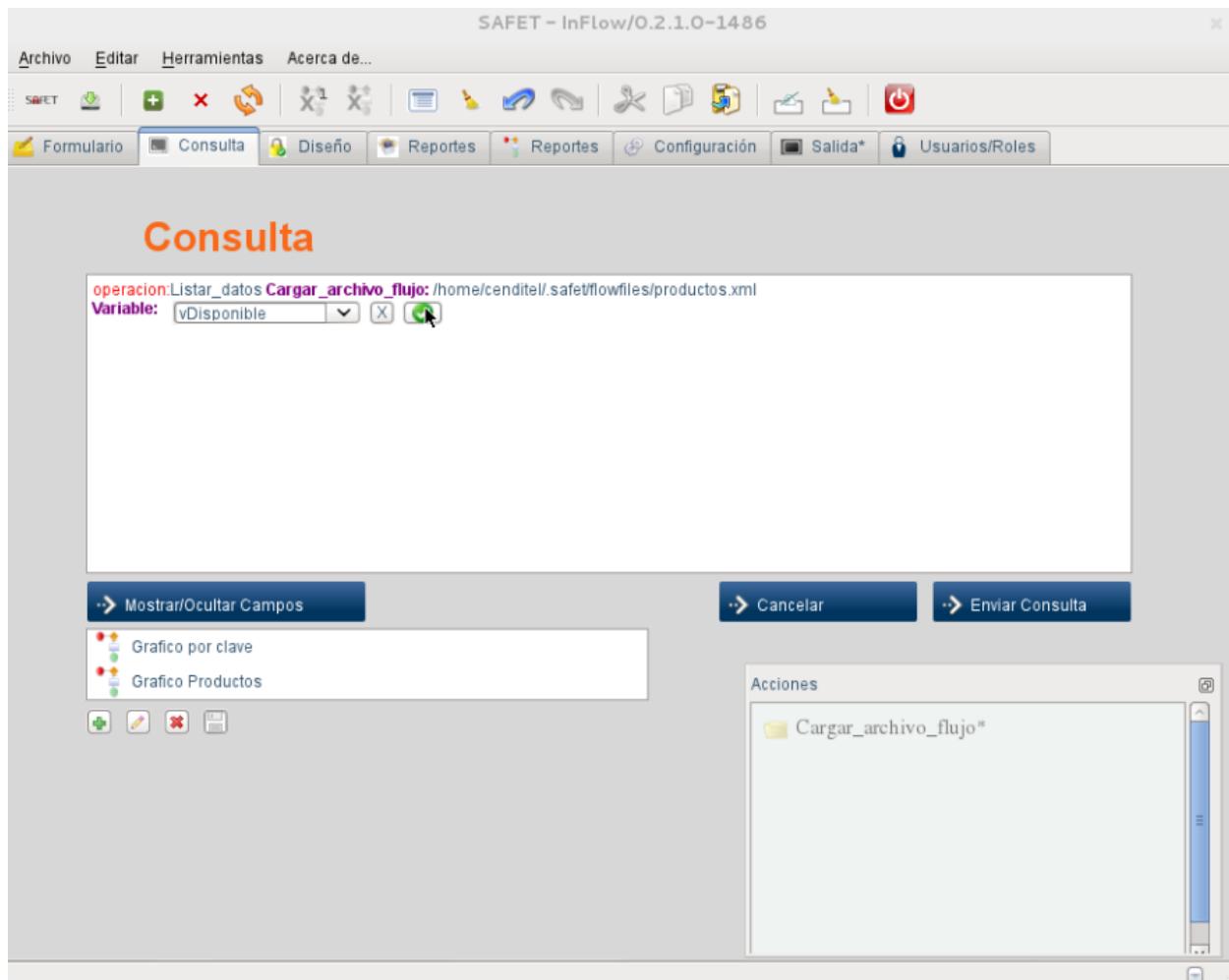


Figura 7.103: Figura 173: Botón

### 8° OCTAVO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**Consulta fue exitosa....ok! (Ver reporte)** ), como se muestra en la siguiente *Figura 174: Fin de la operación*

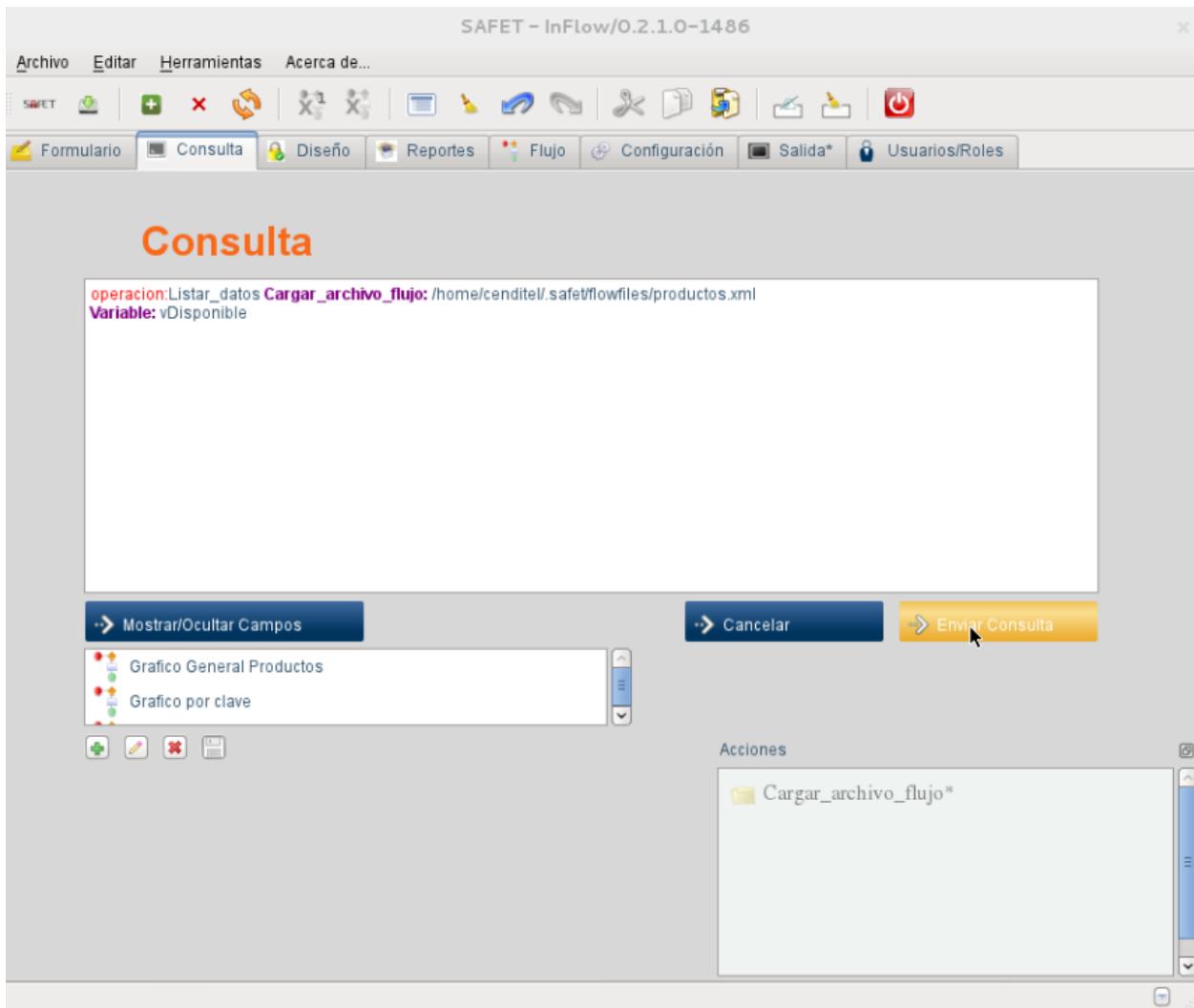


Figura 7.104: **Figura 174: Fin de la operación**

### 9° PASO

- Nos saldrá el gráficamente el **parámetro** la permitirá buscar cualquier dato que comience por cualquier palabra, por ejemplo que liste los datos que comiencen por (**vita**) y pulsamos en el botón (**ok**) para ejecutar la operación, como se muestra en la siguiente *Figura 175: Opción de parámetro*

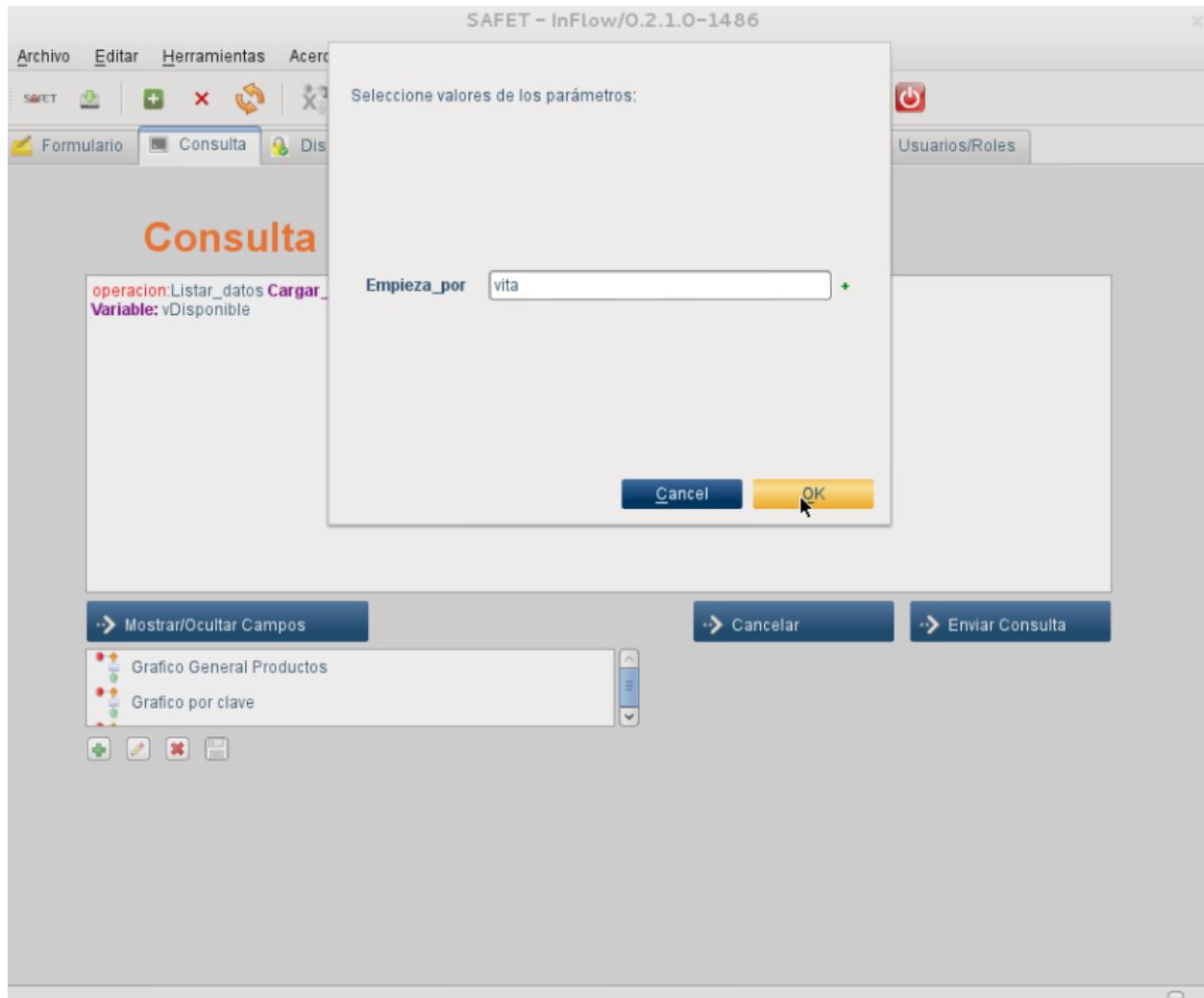


Figura 7.105: Figura 175: Opción de parámetro

### 10° PASO

- Damos click al resultado (**Ver reporte**) para ver el listado de datos, como se muestra en la siguiente *Figura 176: Ver reporte*

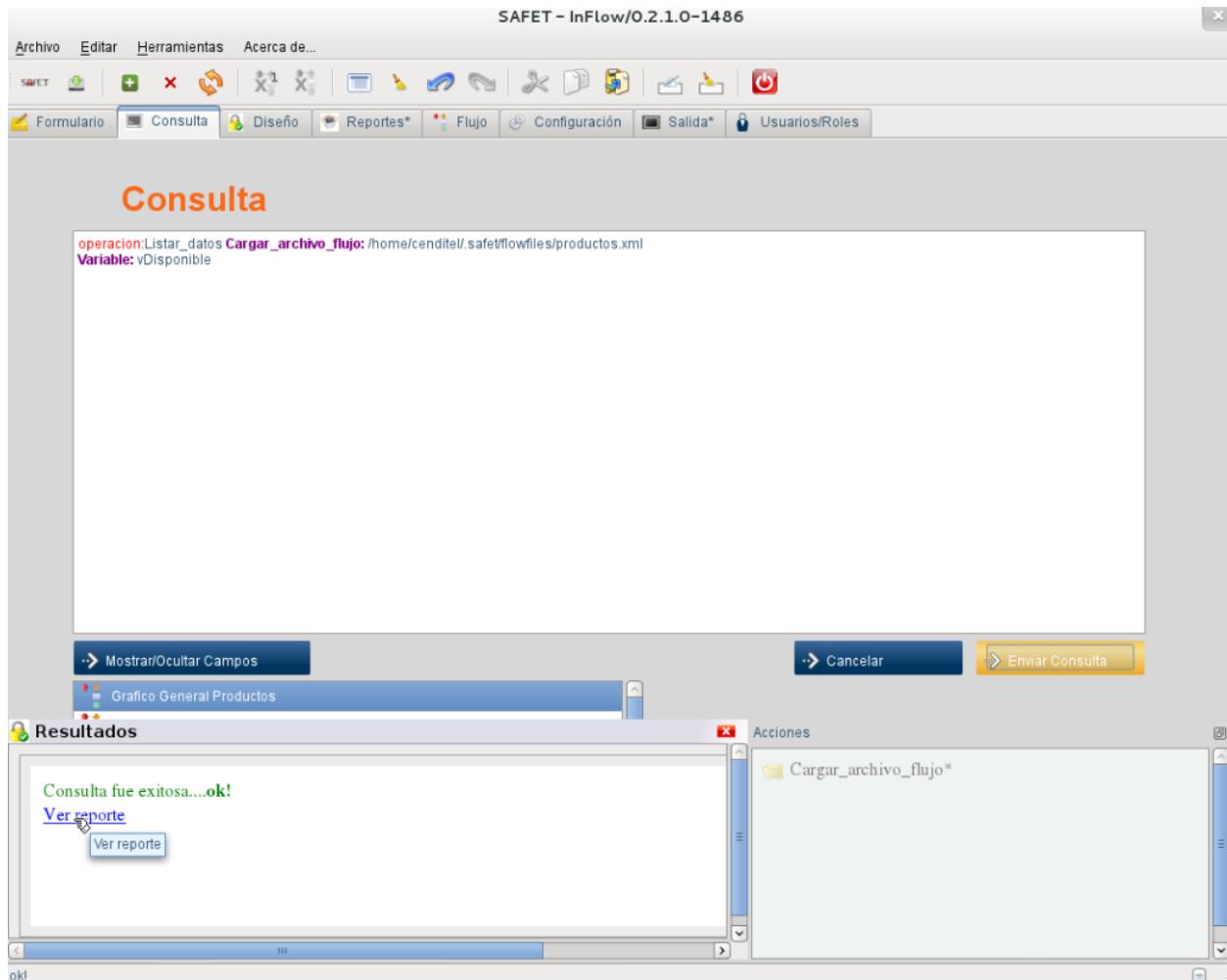


Figura 7.106: **Figura 176: Ver reporte**

---

**Nota:** Observen el *Figura 177: Reporte* de datos que solo comienza por la palabra (vita)

---

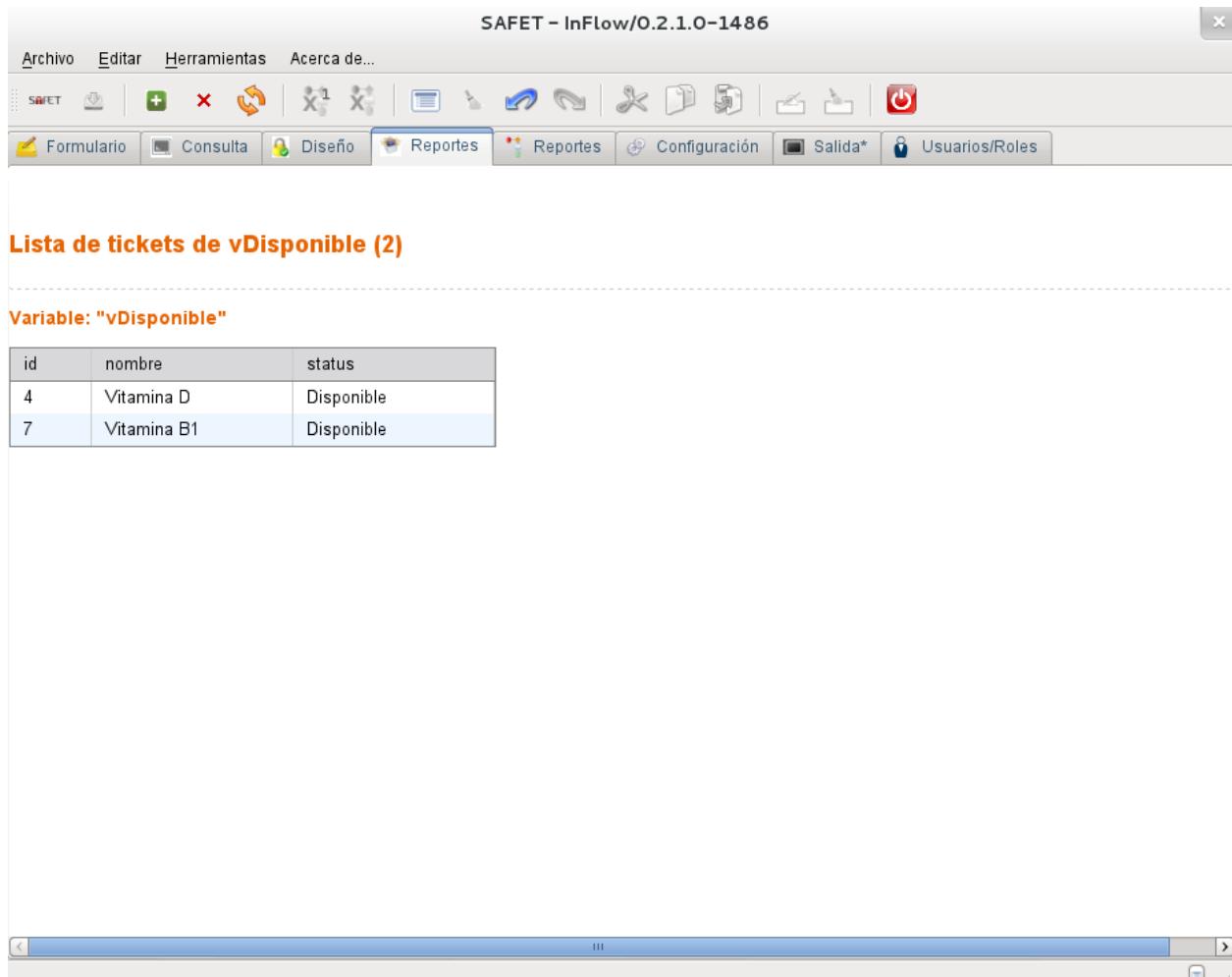


Figura 7.107: **Figura 177: Reporte**

## 7.5 Ejecución de listados de reportes con Autofiltros

### 7.5.1 A).- Editar el archivo (productos.xml)

#### 1° PRIMER PASO

- Abrimos el archivo **productos.xml** que esta ubicado en el directorio **<HOME>.safet/flowfiles/**.

#### 2° SEGUNDO PASO

- Copiamos esta etiqueta **<autofilter>**

```
<autofilter
    query="select categoria from productos"
    report="yes"
    type="split"
    tokenlink=""
    id="por_categoria"
    source="Pedido"
    />
```

#### 3° TERCER PASO

- Insertamos la etiqueta **<autofilter>** en una de las tareas de nuestro archivo **productos.xml** como por ejemplo **Tarea(vRegistrado)**, como se muestra en el siguiente código:

```
<!-- TAREA
*****
/ Registrado /
*****
-->
<task title="en inventario" id="Registrado">

<port side="forward" type="split">
    <connection query="select status from productos"
        options="Pedido" source="Pedido"/>
</port>

<!-- Autofiltros-->
<autofilter
    query="select categoria from productos where status = 'Registrado'"
    report="yes"
    type="split"
    tokenlink=""
    id="por_categoria"
    source="Pedido"
    />

<variable config="1"
    documentsource="select nombre,status from productos"
```

```

type="sql" tokenlink="" id="vRegistrado"
rolfield="(select rol from productos_registro where
productoid=productos.id and regstatus='Registrado') as rol"
scope="task" timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Registrado') as fecha"/>
</task>
```

## 7.5.2 B).- Agregar campo en la tabla productos

### 1° PRIMER PASO

- Abrimos el navegador web.

### 2° SEGUNDO PASO

- Buscamos en la barra de herramientas del navegador web, el plugin '**sqlite-manager**' , como se muestra en la siguiente *Figura 178: Listar datos*

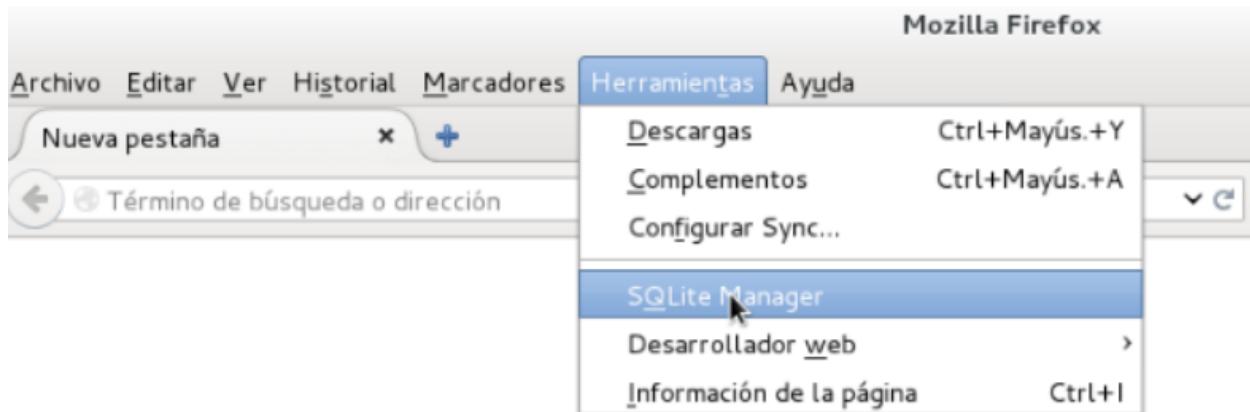


Figura 7.108: **Figura 178: Listar datos**

### 3° TERCER PASO

- Damos una click en el botón **conectar base de datos**, como se muestra la siguiente *Figura 179: Listar datos*:

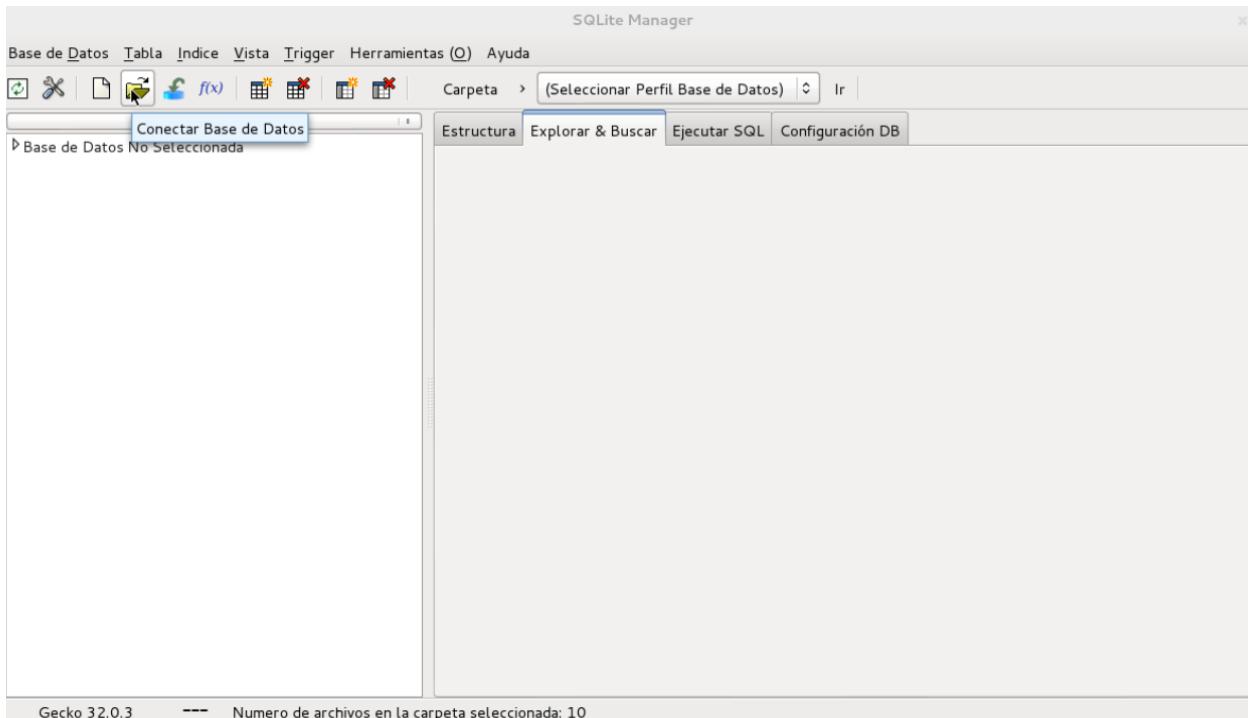


Figura 7.109: [Figura 179: Listar datos](#)

### 4° CUARTO PASO

- Abrimos nuestra base de datos por ejemplo **mydb.db**, como se muestra en la siguiente [Figura 180: Listar datos](#)

### 5° QUINTO PASO

- Nos vamos a la tabla productos y damos un click a la opción **Estructura**, como se muestra en la siguiente [Figura 181: Listar datos](#)

### 6° SEXTO PASO

- Agragamos el campo (**categoria**) de tipo **texto** y colocamos por defecto “**no\_clasificado**”, Luego pulsamos al botón (**añadir Columna**), como se muestra en la siguiente [Figura 182: Listar datos](#)

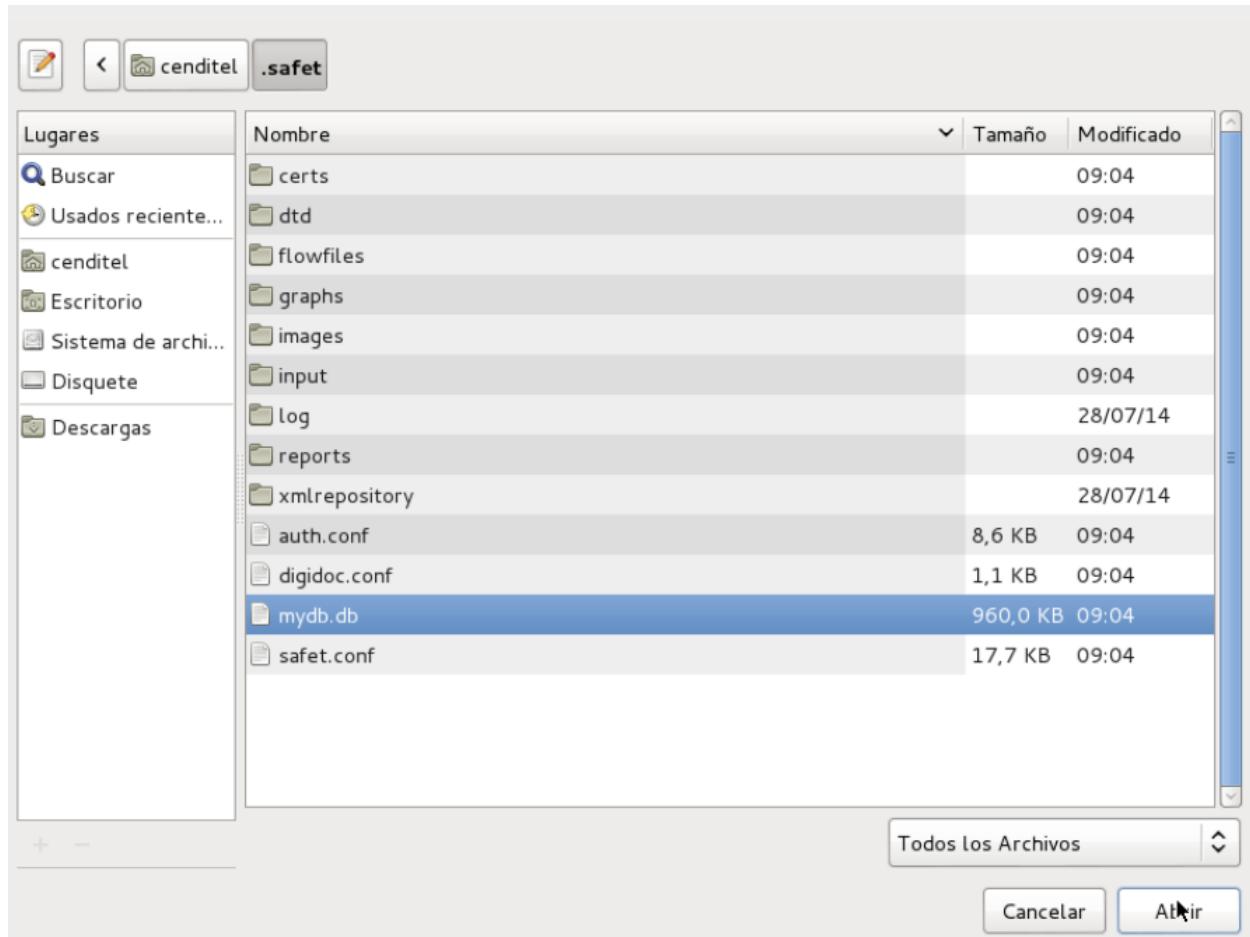


Figura 7.110: Figura 180: Listar datos

## Documentación de PySafet, Publicación

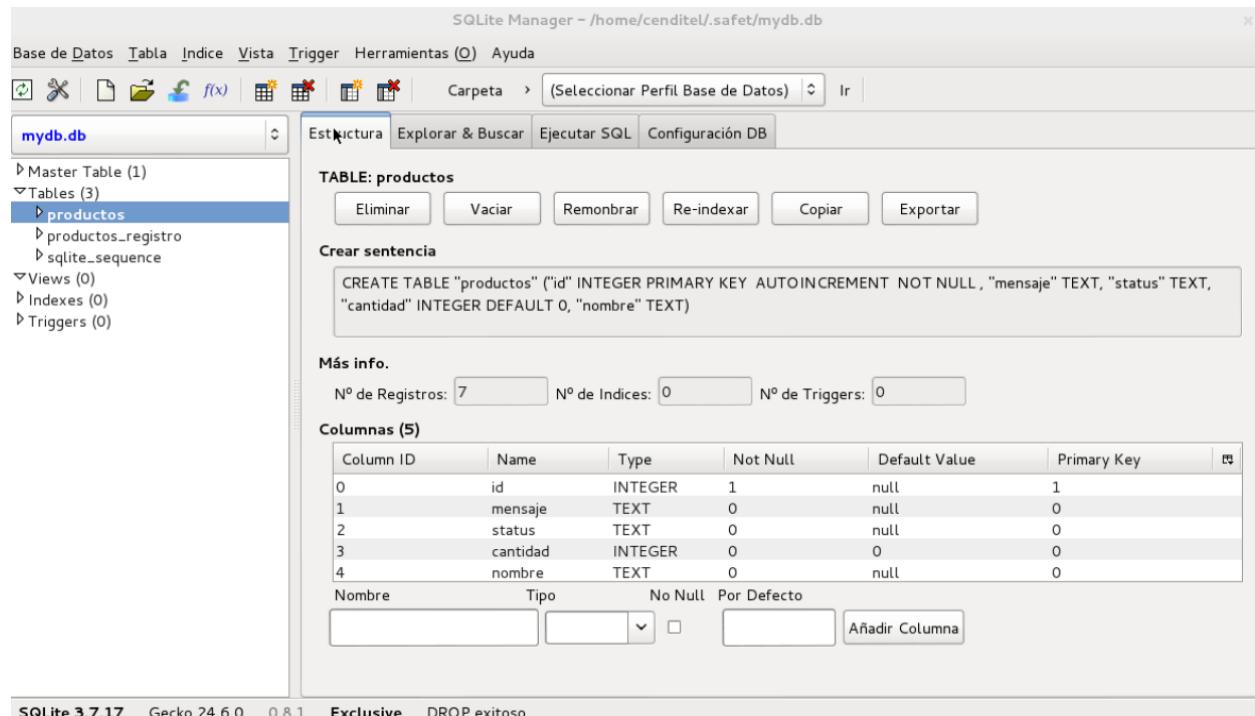


Figura 7.111: Figura 181: Listar datos

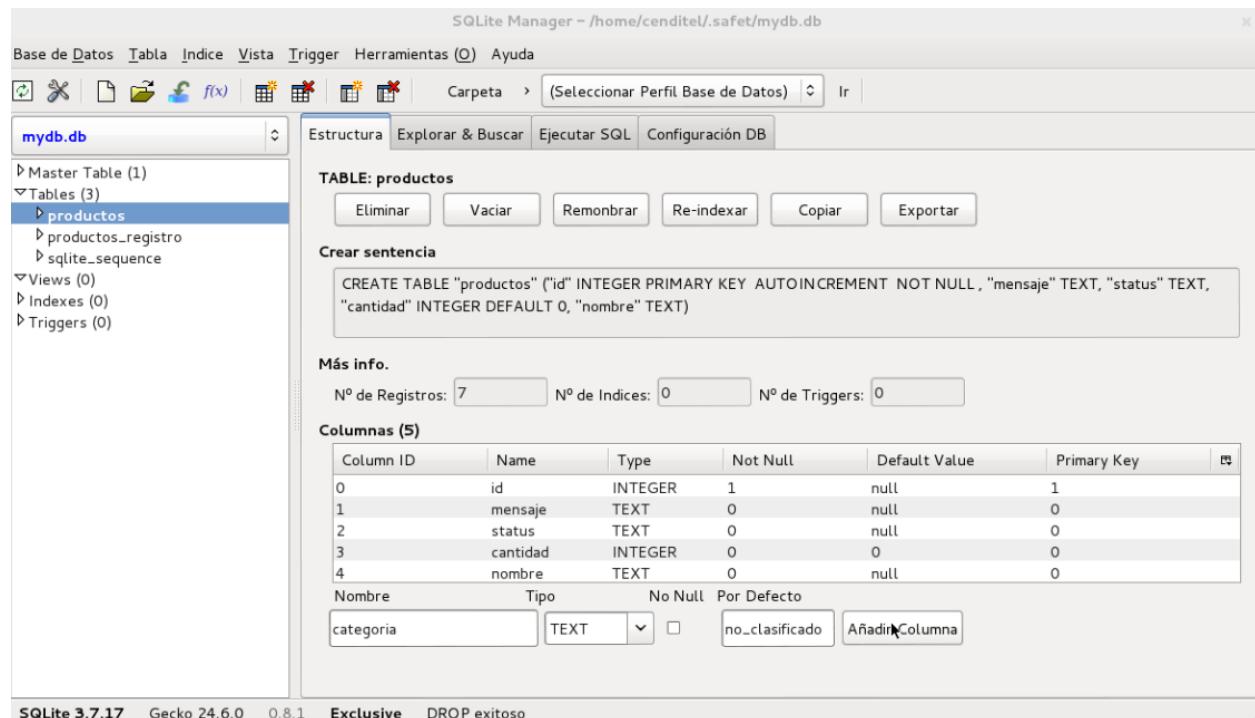


Figura 7.112: Figura 182: Listar datos

## 7° SEPTIMO PASO

- Se nos mostrara el siguiente **mensaje** , pulsamos el botón **ok**, como se muestra en la siguiente *Figura 183: Listar datos*

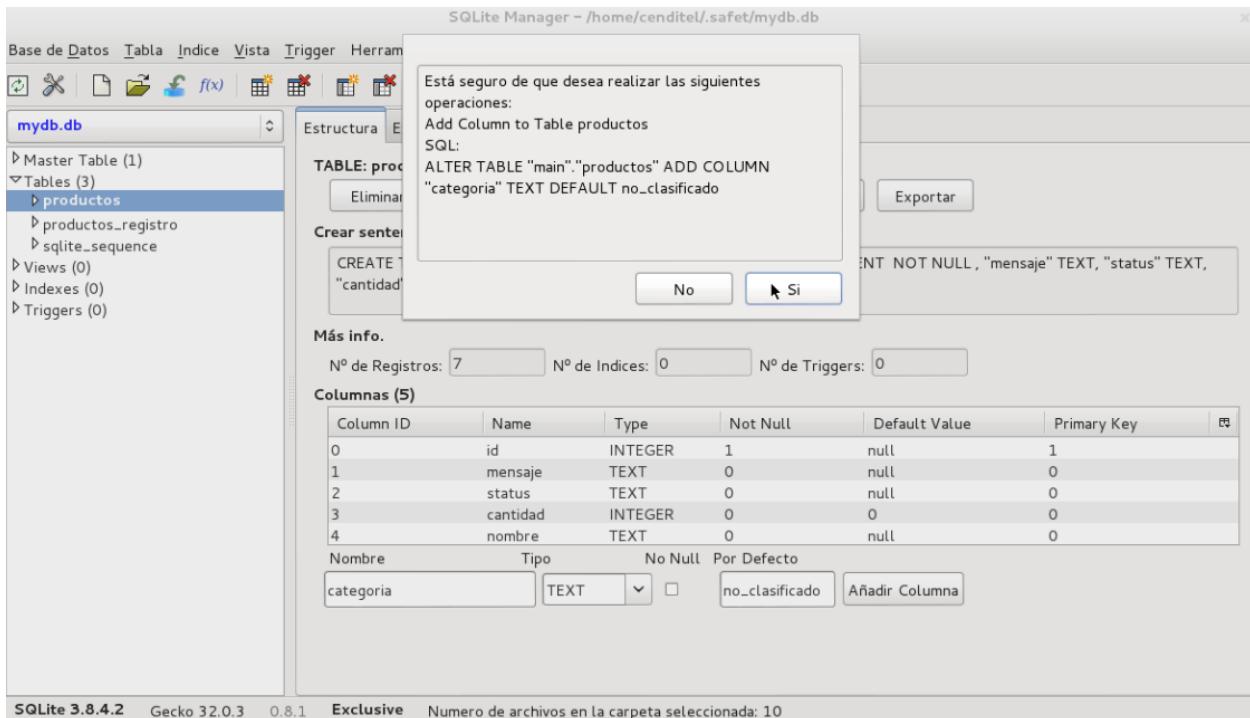


Figura 7.113: *Figura 183: Listar datos*

## 8° OCTAVO PASO

- Observamos el campo agregado, como se muestra en la siguiente *Figura 184: Listar datos*
- Observamos el campo en la opción **Explorar & buscar**, como se muestra en la siguiente *Figura 185: Listar datos*

### 7.5.3 C).- Autofiltros con la interfaz gráfica inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

## Documentación de PySafet, Publicación

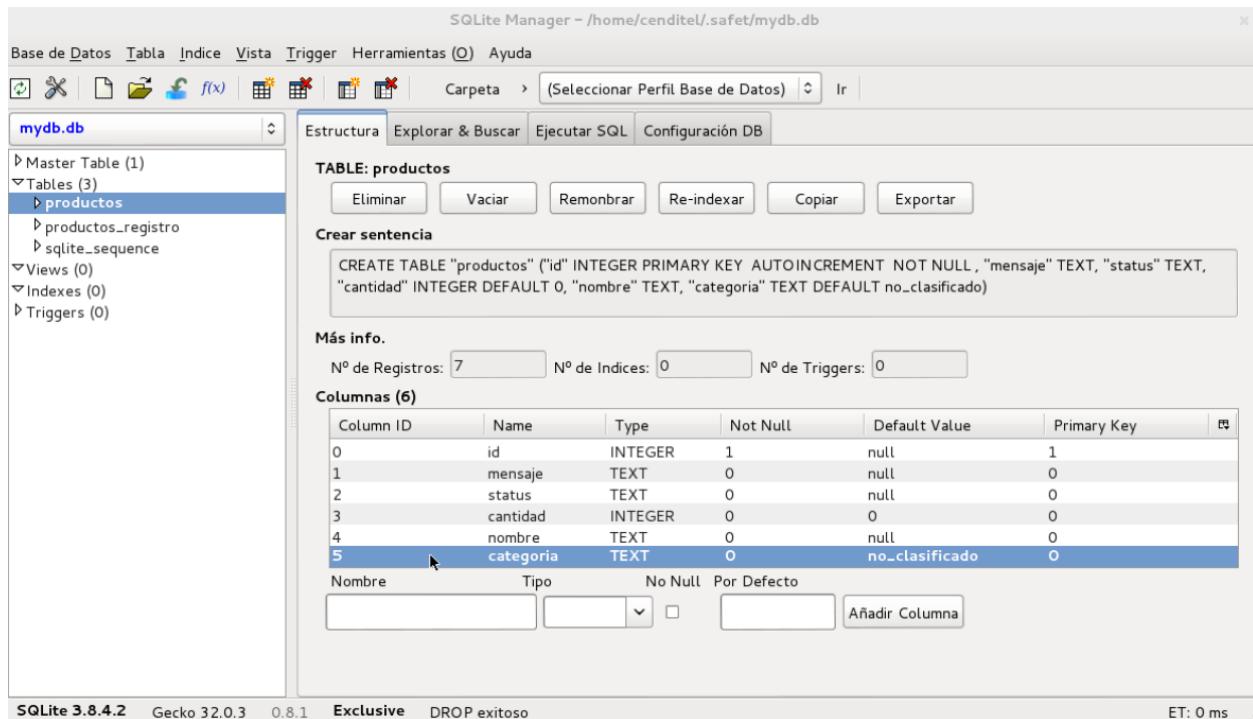


Figura 7.114: Figura 184: Listar datos

The screenshot shows the SQLite Manager interface for the same database. The 'Explorar & Buscar' tab is active. The 'productos' table is selected. The table data is displayed in a grid:

id	mensaje	status	cantidad	nombre	categoria
1	Pedido	Pedido	1000	Jabón Protex	no_clasificado
2	Pedido	Registrado	220	Pañales Pamper	no_clasificado
3	Registrado	Registrado	1200	Vitamina C	no_clasificado
4	Registrado	Registrado	30	Vitamina D	no_clasificado
5	Registrado	Registrado	300	Vitamina B12	no_clasificado
7	Pedido	Pedido	600	Vitamina B1	no_clasificado
8	Registrado	Registrado	25	Vitamina B8	no_clasificado

Figura 7.115: Figura 185: Listar datos

\$ inflow

## 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 186: Autenticación de Inflow*.



Figura 7.116: **Figura 186: Autenticación de Inflow.**

## 3° TERCER PASO

- Damos click a la segunda opción (**Realizar consultas**), como se muestra en la siguiente *Figura 187: Realizar consultas*

## 4° CUARTO PASO

- Damos click a la opción (**Mostrar/Ocultar Campos**), la cual se nos mostrara las acciones la **Opciones de listados**, como se muestra en la siguiente *Figura 188: Acciones de reportes*

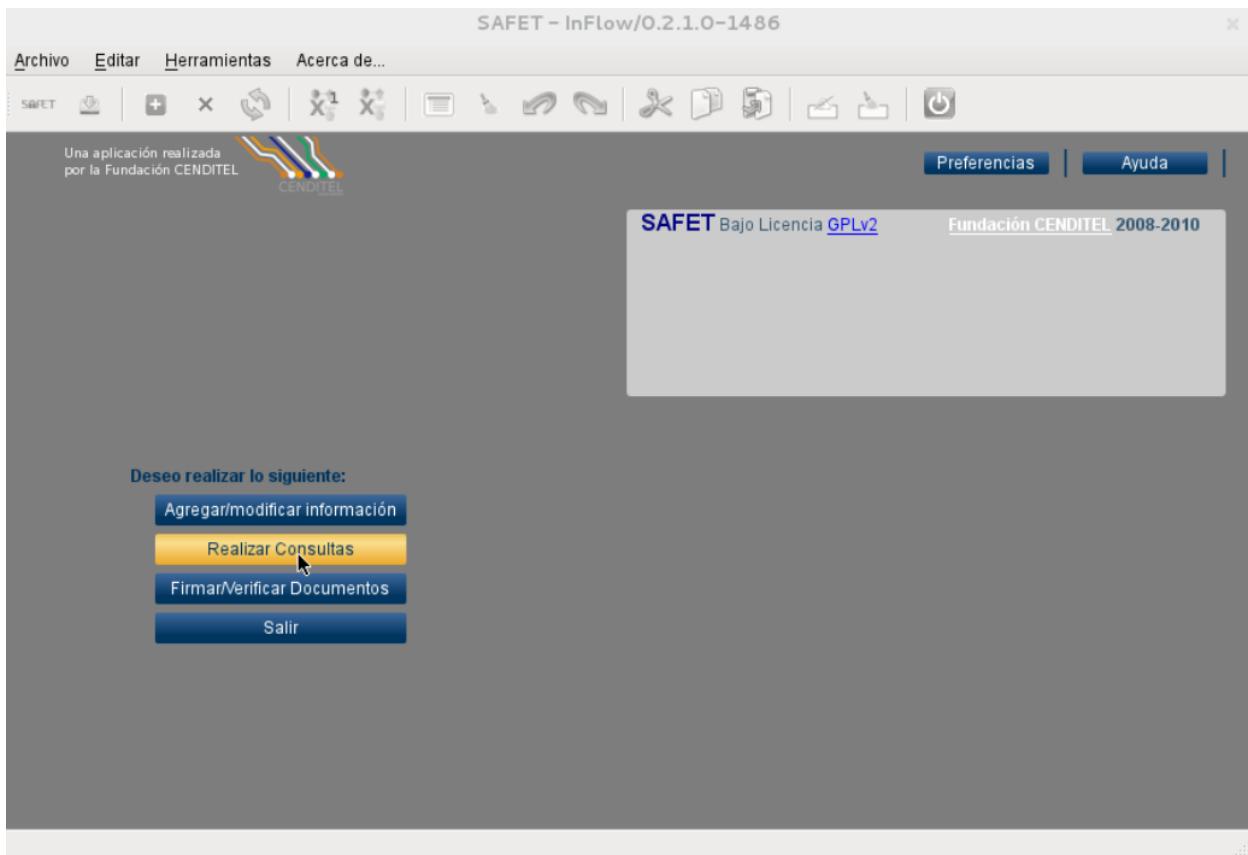


Figura 7.117: **Figura 187: Realizar consultas**

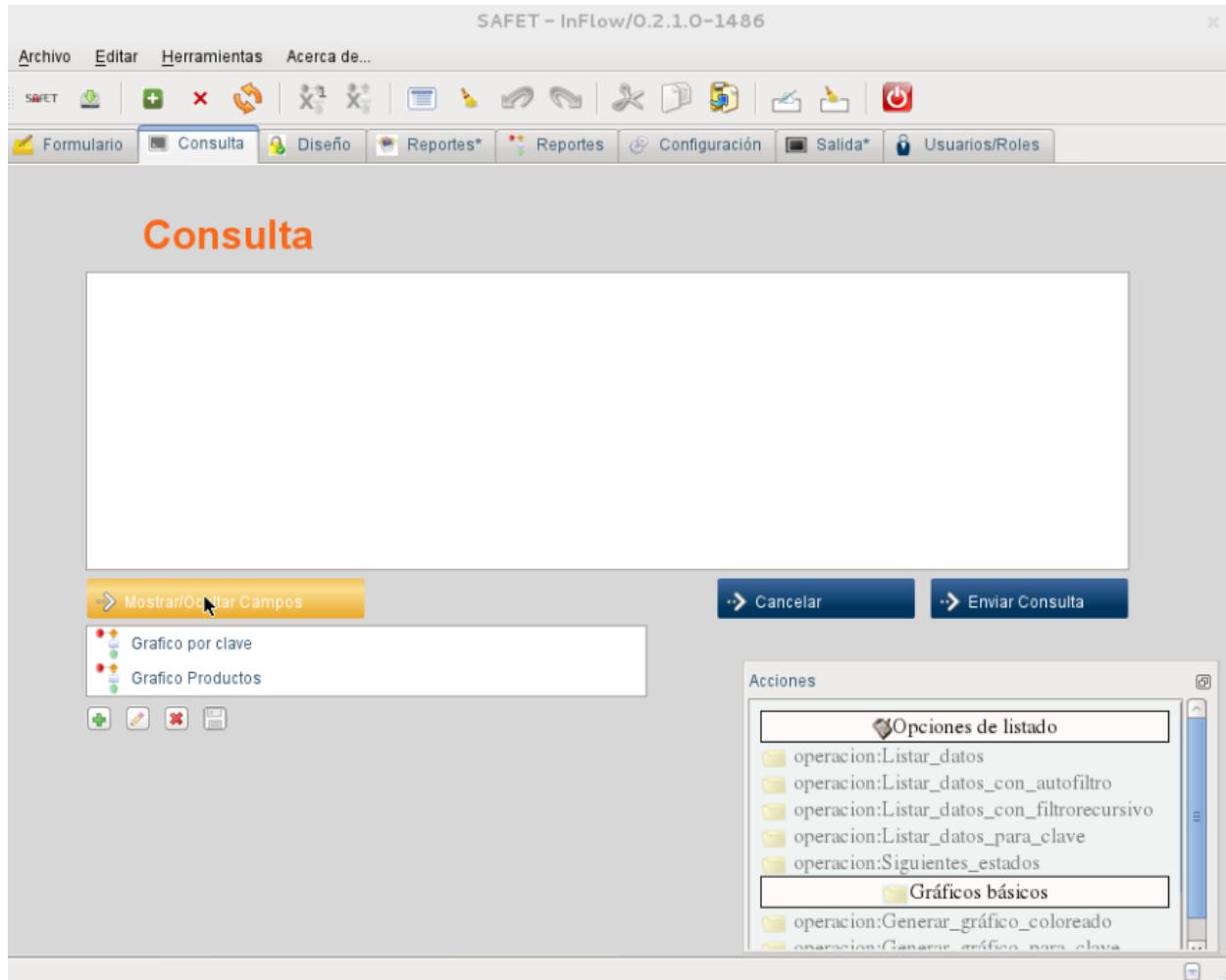


Figura 7.118: Figura 188: Acciones de reportes

### 5° QUINTO PASO

- Damos un click a la opción (**operacion:Listar\_datos\_con\_autofiltro**), como se muestra en la siguiente *Figura 189: operacion:Listar\_datos\_con\_autofiltro*

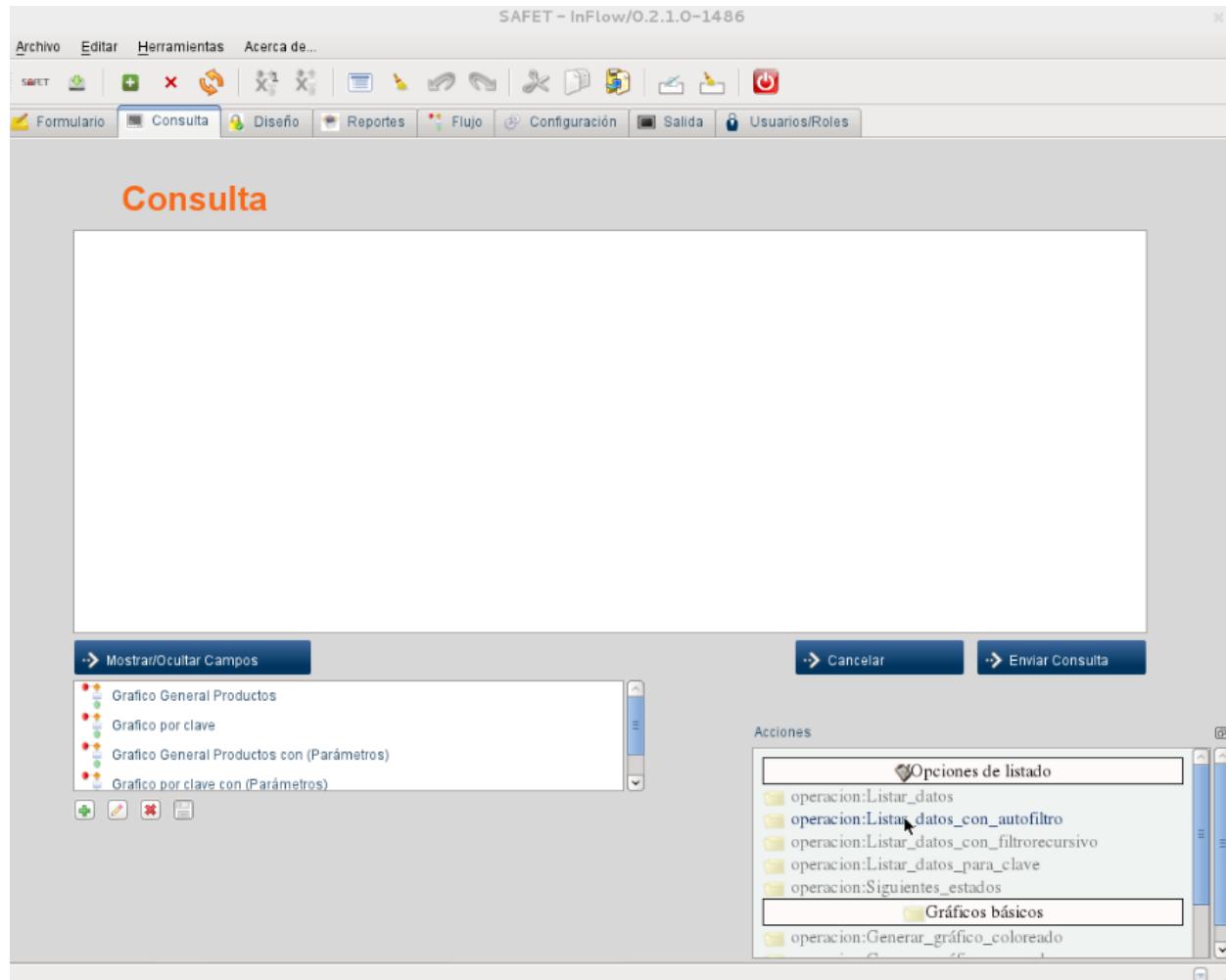


Figura 7.119: **Figura 189: operacion:Listar\_datos\_con\_autofiltro**

### 6° SEXTO PASO

- Damos un click a campo (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 190: Cargar\_archivo\_flujo\**

### 7° SEPTIMO PASO

- Seleccionamos el archivo **productos.xml**, como se muestra en la siguiente *Figura 191: Archivo (XML)*

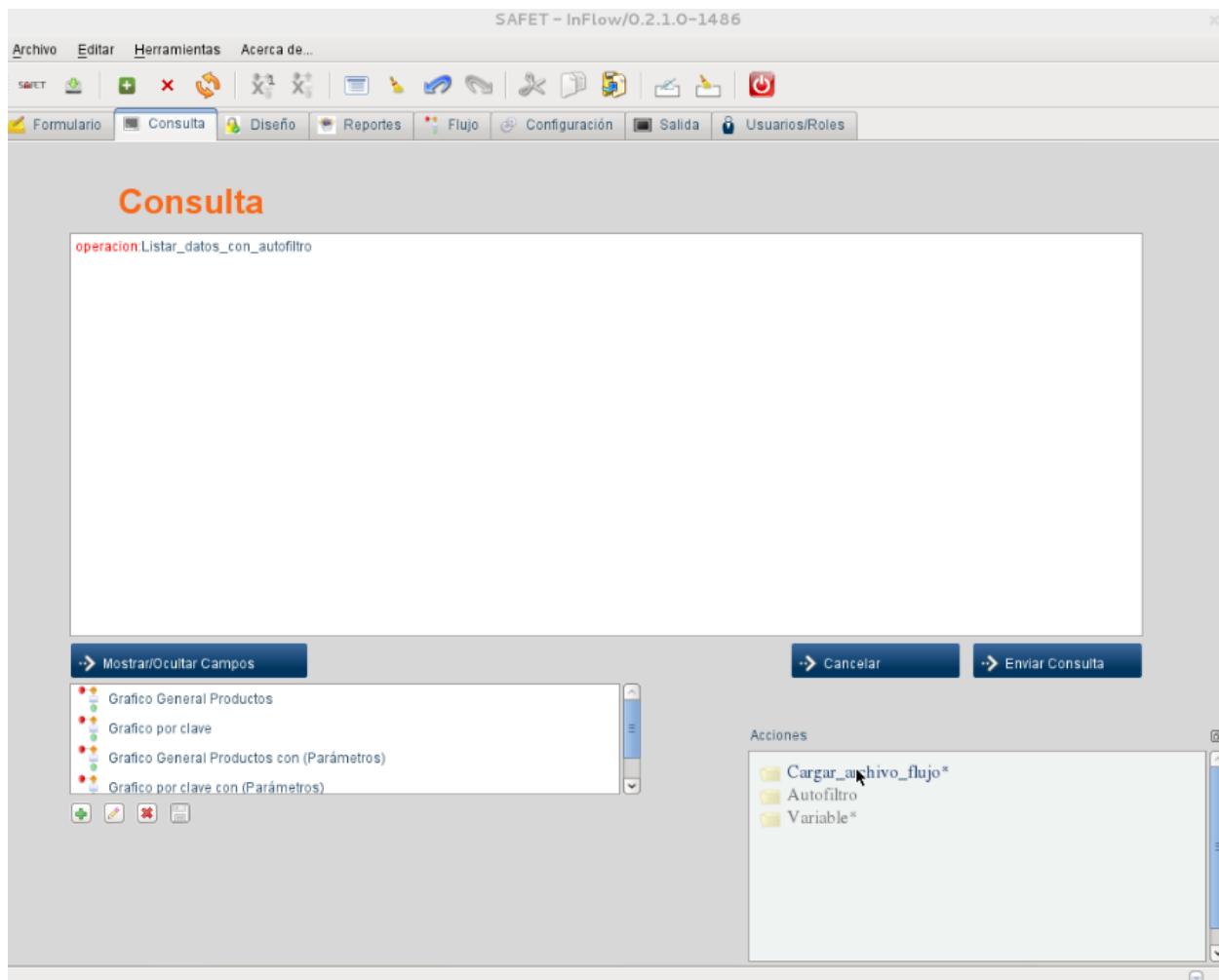


Figura 7.120: Figura 190: Cargar\_archivo\_flujo\*

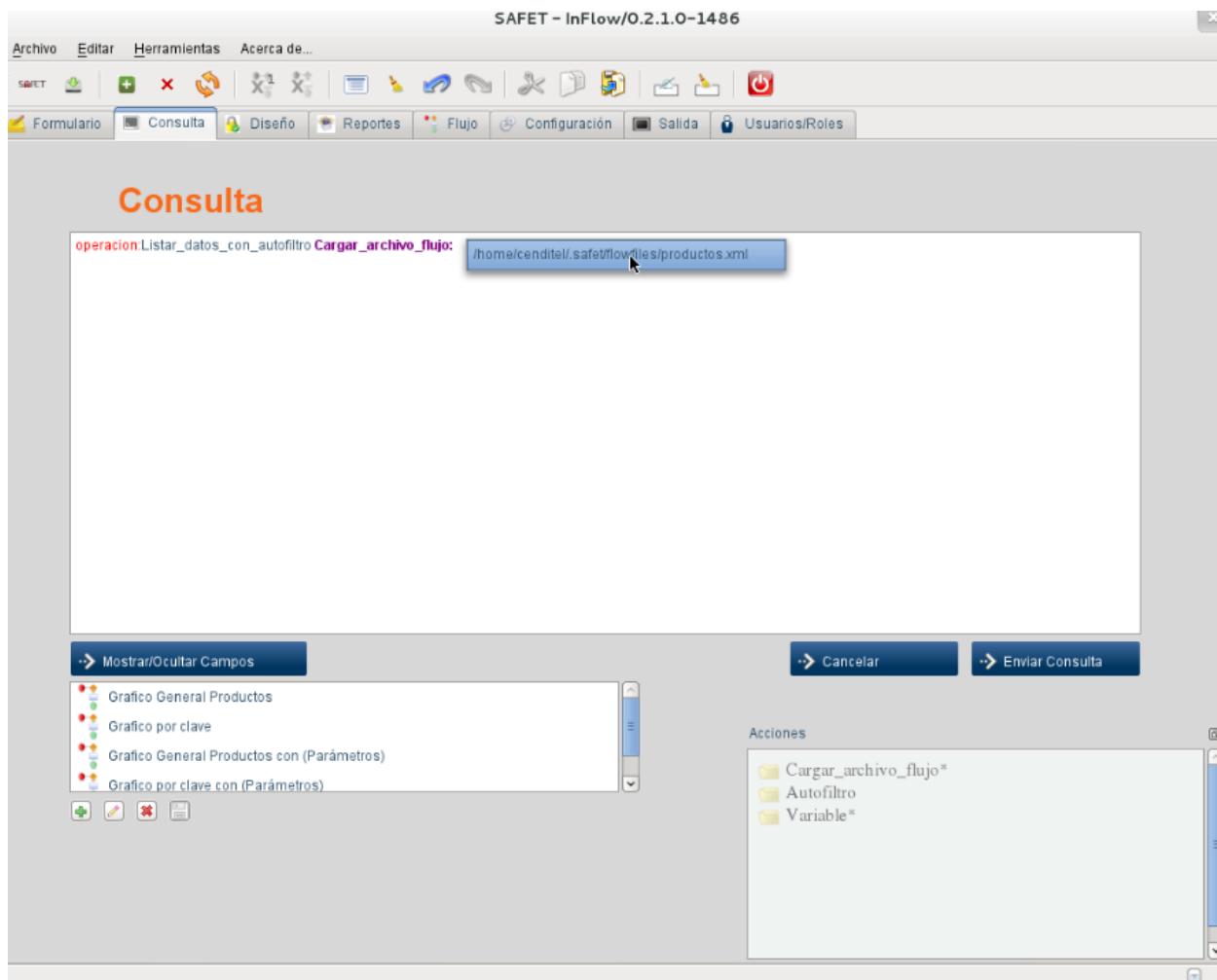


Figura 7.121: **Figura 191: Archivo (XML)**

## 8° OCTAVO PASO

- Damos un click al **botón** con la flecha verde, significando que ya está lleno el campo, como se muestra en la siguiente *Figura 192: Resultado de la operación*

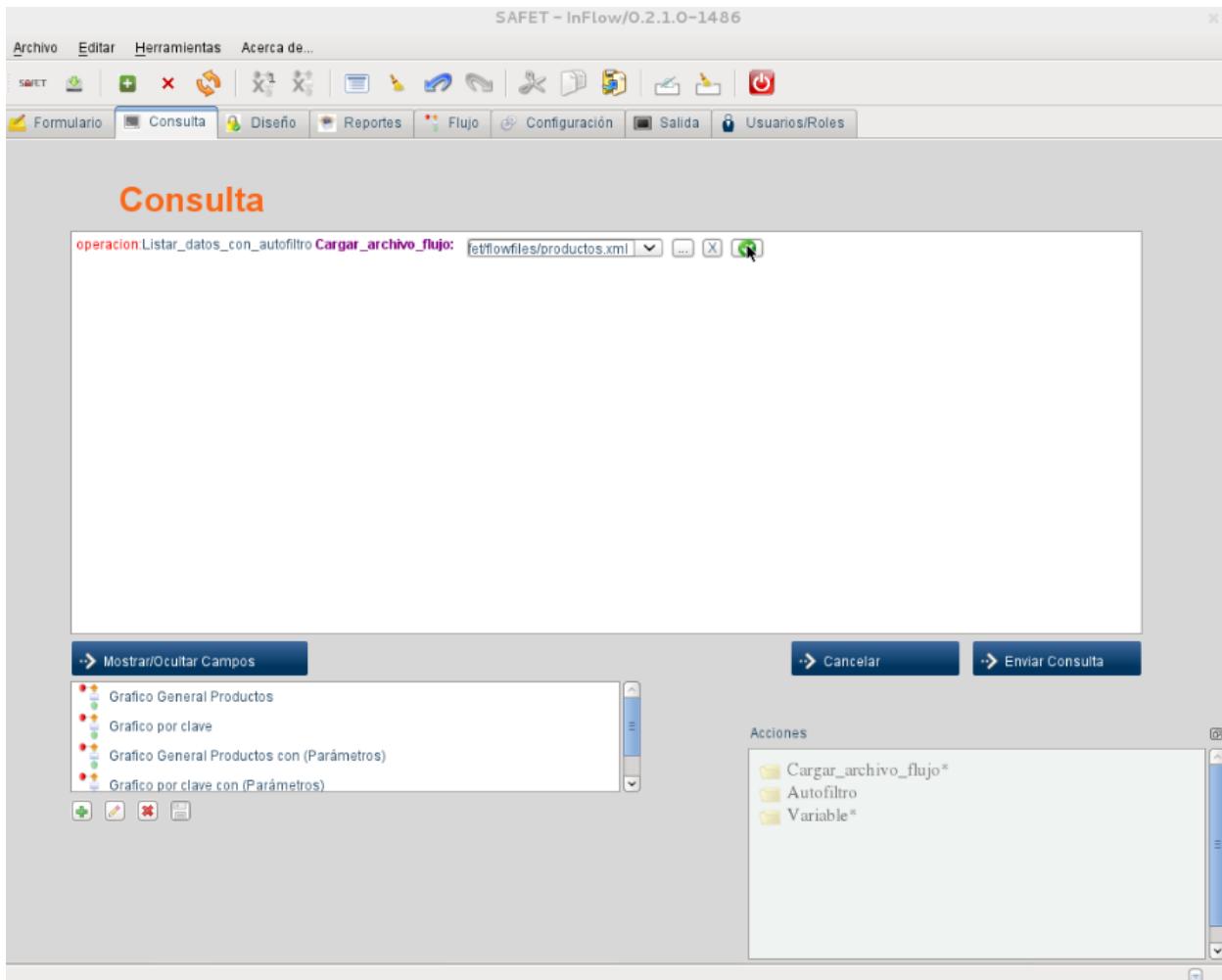


Figura 7.122: **Figura 192: Resultado de la operación**

## 9° NOVENO PASO

- Damos un click a campo (**Autofiltro**), como se muestra en la siguiente *Figura 193: Autofiltro*

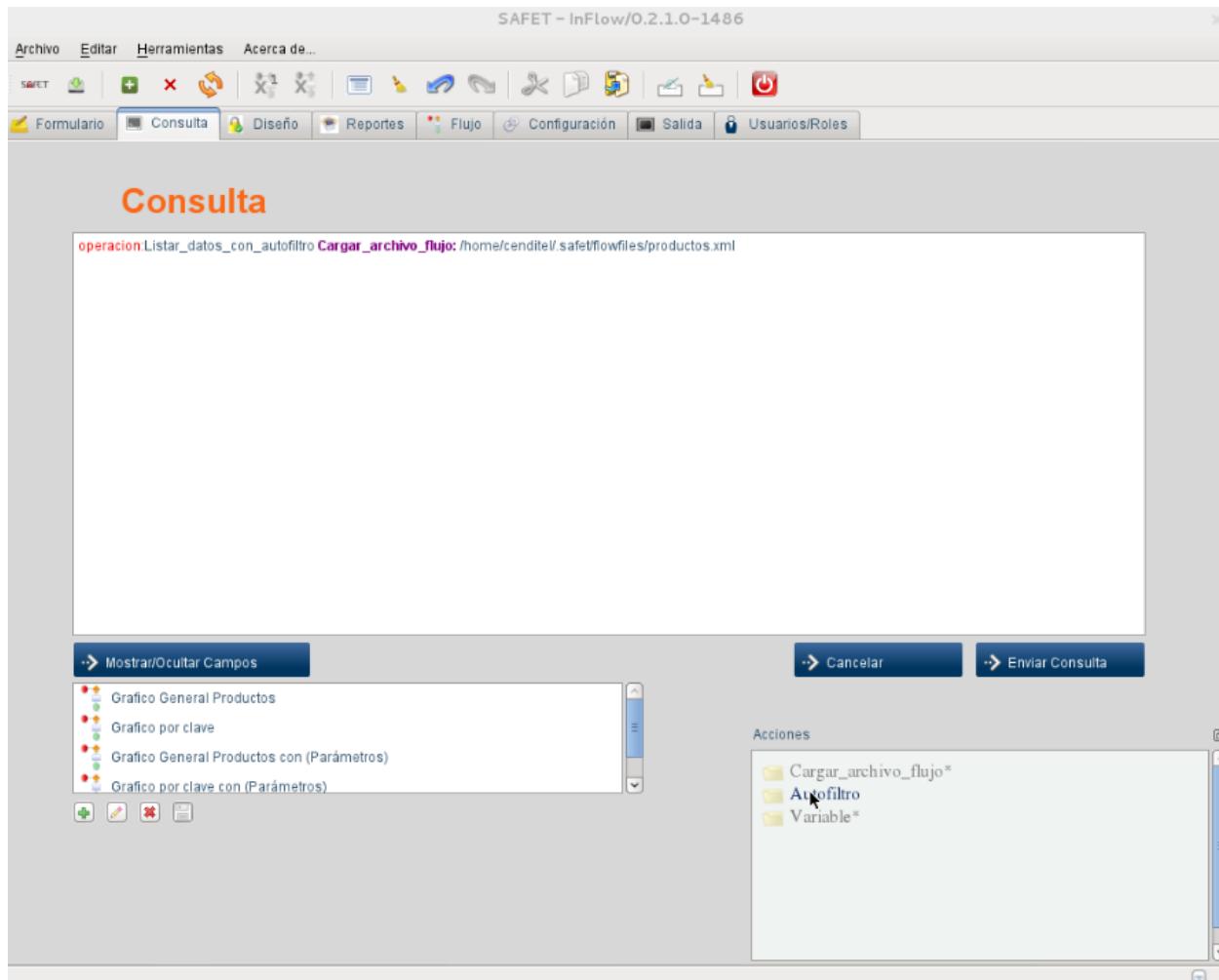


Figura 7.123: **Figura 193: Autofiltro**

## 10° DECIMO PASO

- Seleccionamos una **categoria**, como se muestra en la siguiente *Figura 194: Selección de categoria.*

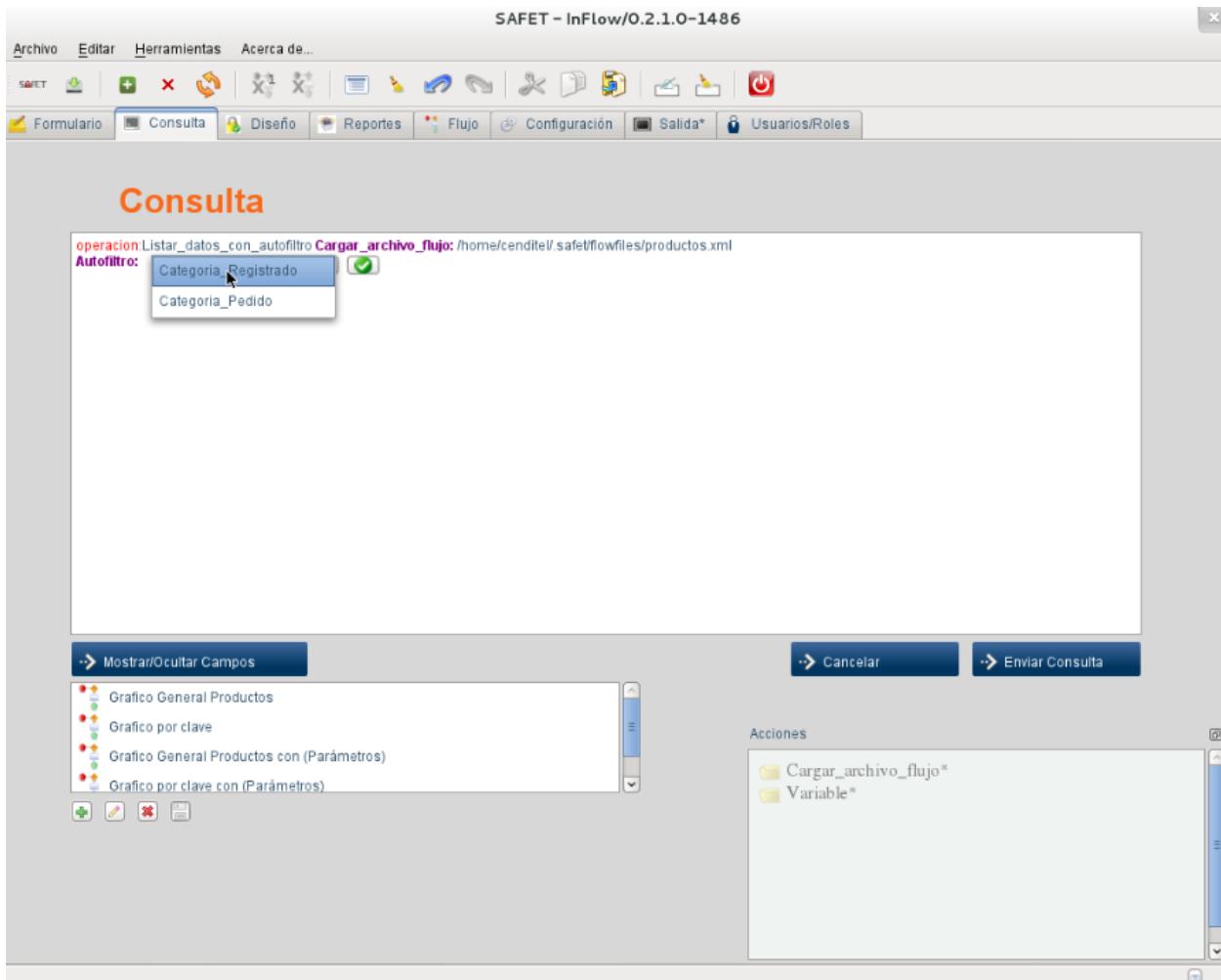


Figura 7.124: **Figura 194: Selección de categoria.**

## 11° DECIMO PRIMER PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 195: Botón (Fin de campo)*

## 12° DECIMO SEGUNDO PASO

- Damos un click a campo (**Variable\***), como se muestra en la siguiente *Figura 196: Campo (Variable\*)*

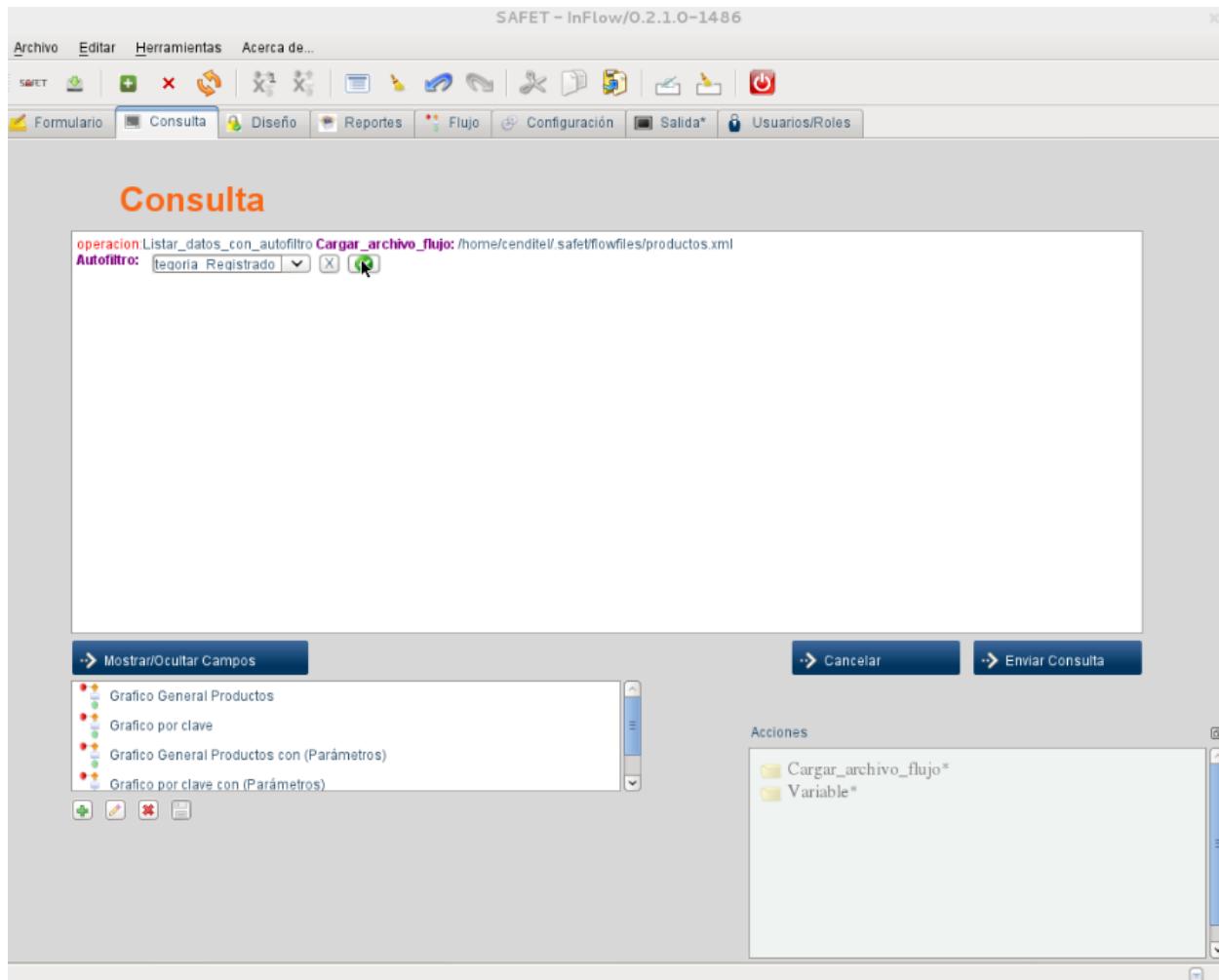


Figura 7.125: Figura 195: Botón (Fin de campo)

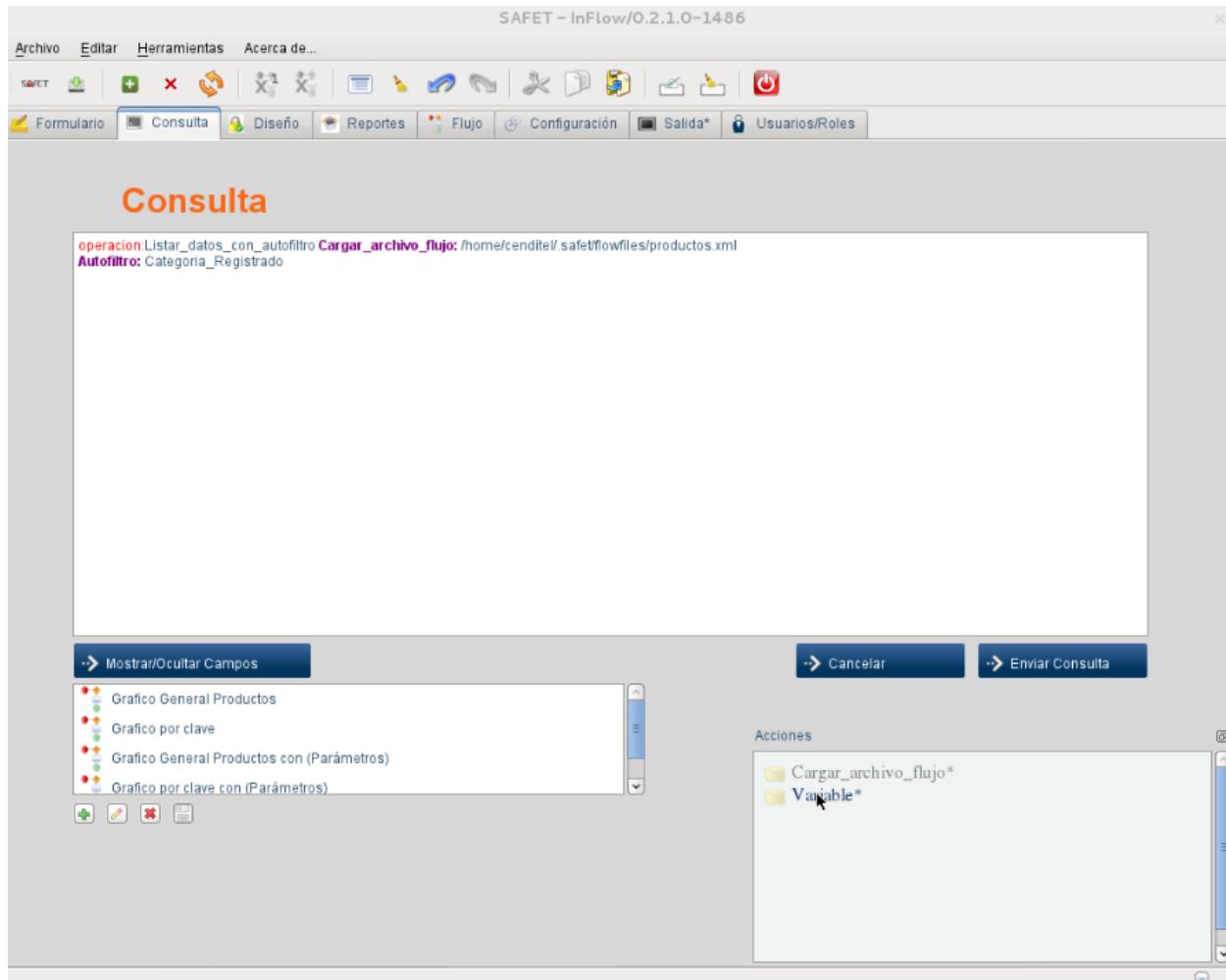


Figura 7.126: Figura 196: Campo (Variable\*)

### 13° DECIMO TERCER PASO

- Seleccionamos a la variable **vRegistrado** la cual es, la variable que estamos listando, como se muestra en la siguiente *Figura 197: Selección de la Variable*

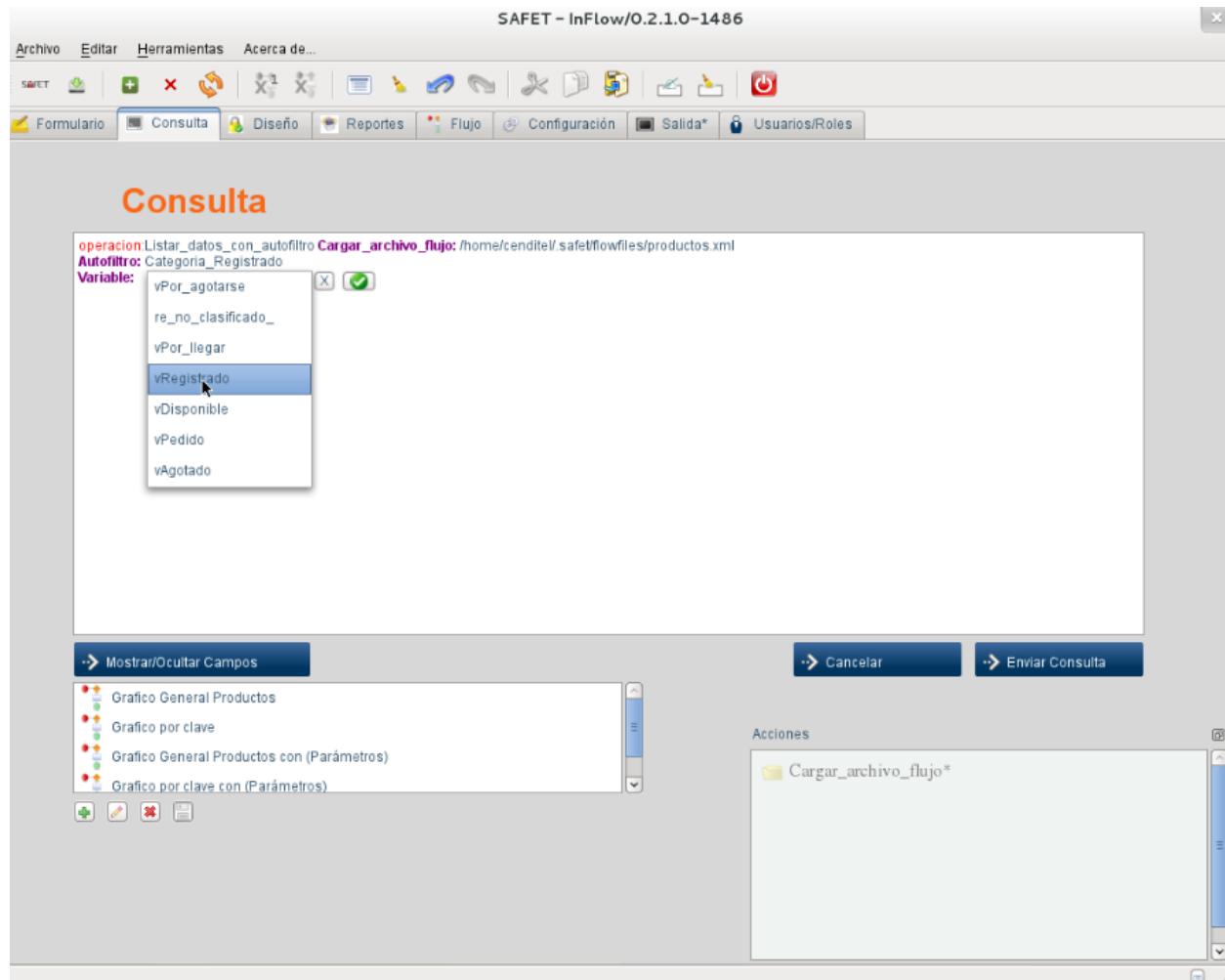


Figura 7.127: **Figura 197: Selección de la Variable**

### 14° DECIMO CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 198: Botón (Fin del campo)*

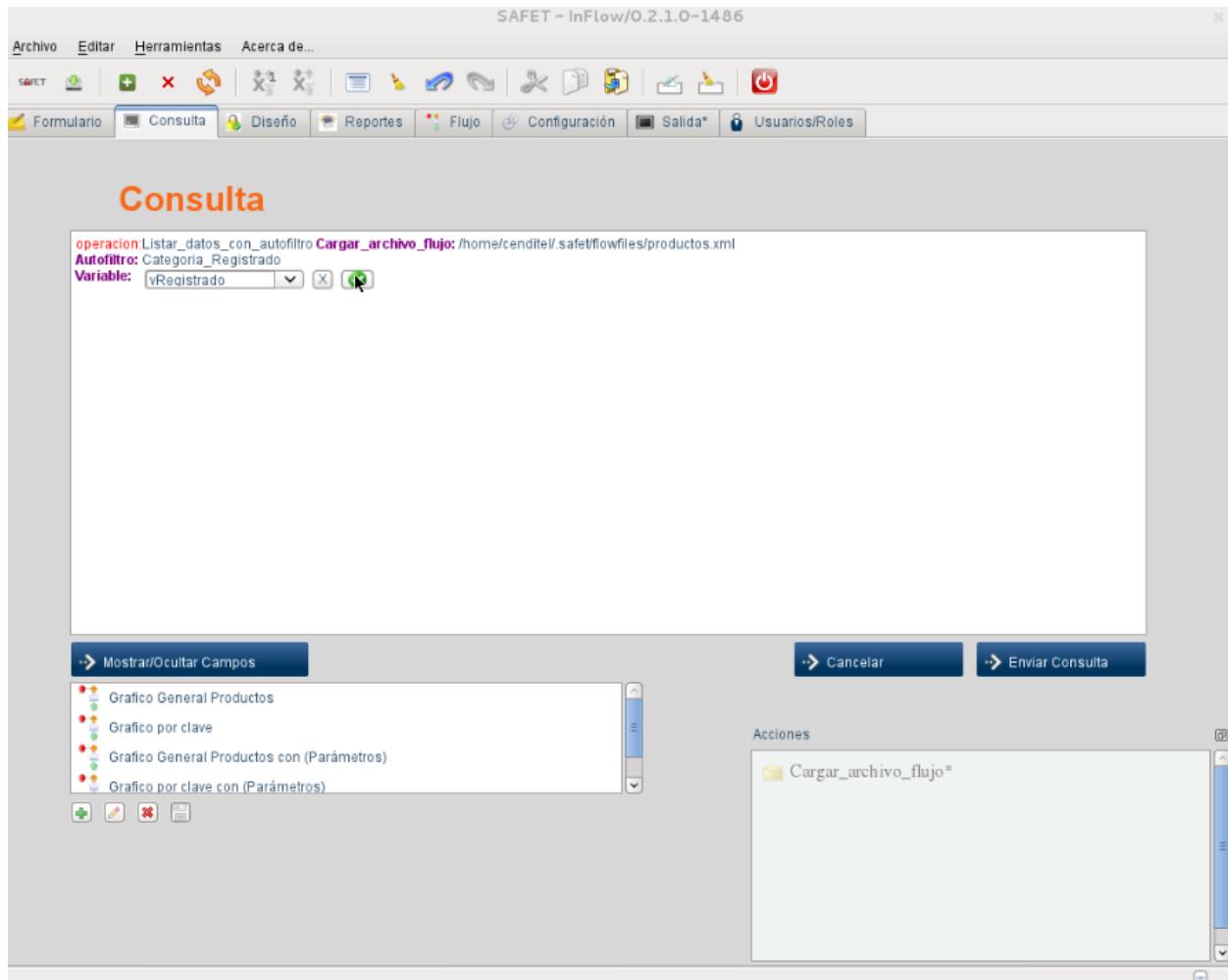


Figura 7.128: Figura 198: Botón (Fin del campo)

### 15° DECIMO QUINTO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**Consulta fue exitosa....ok!** (*Ver reporte*)), como se muestra en la siguiente *Figura 199: Botón (Enviar formulario)*

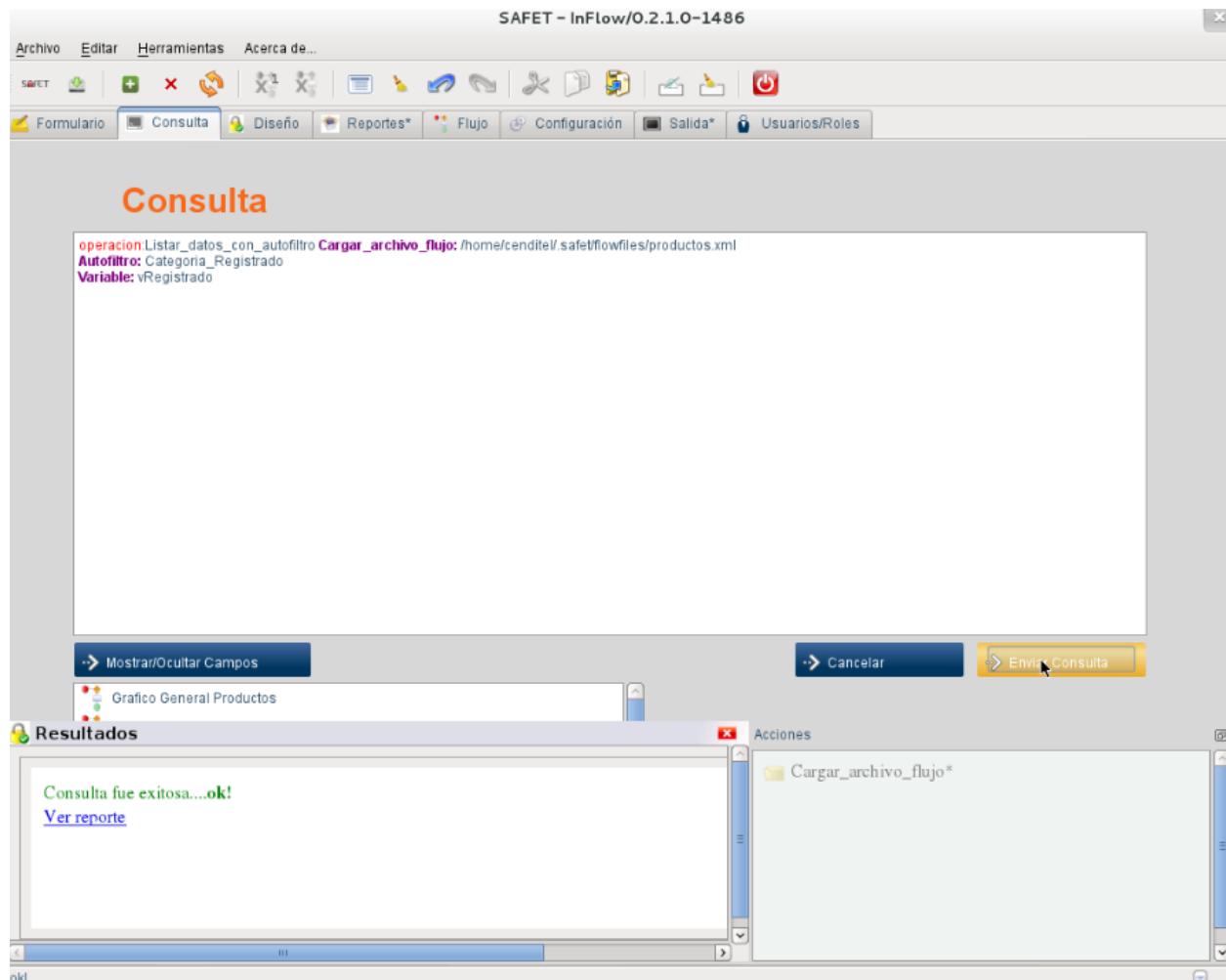


Figura 7.129: Figura 199: Botón (Enviar formulario)

### 16° DECIMO SEXTO PASO

- Damos un click al link (**Ver reporte**), como se muestra en la siguiente *Figura 200: Ver reporte*

---

**Nota:** Se nos mostrara una tabla con los datos que estamos filtrando, en este caso es la variable **vRegistrado**, como se muestra en la siguiente *Figura 201: Tabla*

---

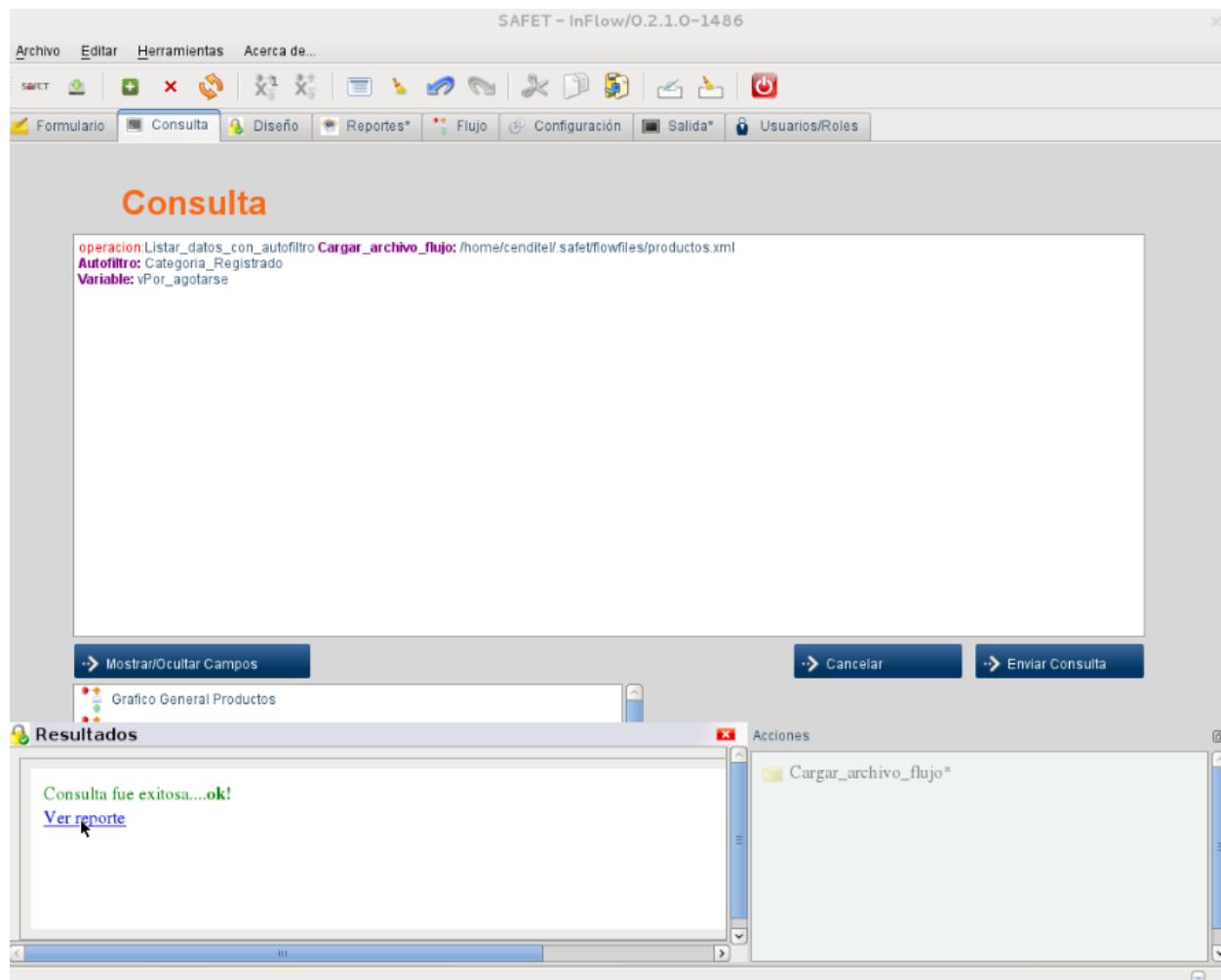


Figura 7.130: **Figura 200: Ver reporte**

The screenshot shows the SAFET - InFlow interface. At the top, there is a menu bar with options: Archivo, Editar, Herramientas, Acerca de..., followed by a toolbar with various icons. Below the toolbar, a navigation bar contains links: Formulario, Consulta, Diseño, Reportes (which is highlighted), Flujo, Configuración, Salida\*, and Usuarios/Roles. The main content area is titled "Lista de tickets de vRegistrado (4)". It displays a table with the following data:

id	nombre	status	categoria
3	Vitamina C	Registrado	no_clasificado
4	Vitamina D	Registrado	no_clasificado
5	Vitamina B12	Registrado	no_clasificado
8	Vitamina B8	Registrado	no_clasificado

Below the table are two sets of navigation buttons: "<< primero < anterior 1 próximo > último >>" and "25" with a dropdown arrow.

Figura 7.131: Figura 201: Tabla

## 7.6 Ejecución de gráficos usando flujos de trabajos normal

En este tutorial explicaremos el **ejemplo de inventario**, la cual generaremos gráficos de flujo de trabajo con la herramienta **inflow**, para ello debemos tener **instalado inflow**. Ver el siguiente enlace. Instalación de la interfaz gráfica (inflow).

A continuación seguimos los siguientes pasos la cual abriremos inflow y y luego ejecutaremos la acciones:

### 7.6.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 202: Autenticación de Inflow*.



Figura 7.132: Figura 202: Autenticación de Inflow.

### 3° TERCER PASO

- Se le da click a la segunda opción (**Realizar consultas**), como se muestra en la siguiente *Figura 203: Realizar consultas*

## 7.6.2 B.- PRIMERA OPERACIÓN (Gráfico coloreado general)

### 1° PRIMER PASO

- Se le da click a la opción (**Mostrar/Ocultar Campos**), la cual se nos mostrara las acciones la **opciones** de listados, como se muestra en la siguiente *Figura 204: Acciones de reportes*

### 2° SEGUNDO PASO

- Se le da click a la operación (**Generar\_gráfico\_coloreado**), como se muestra en la siguiente *Figura 205: Operación (Generar\_gráfico\_coloreado)*

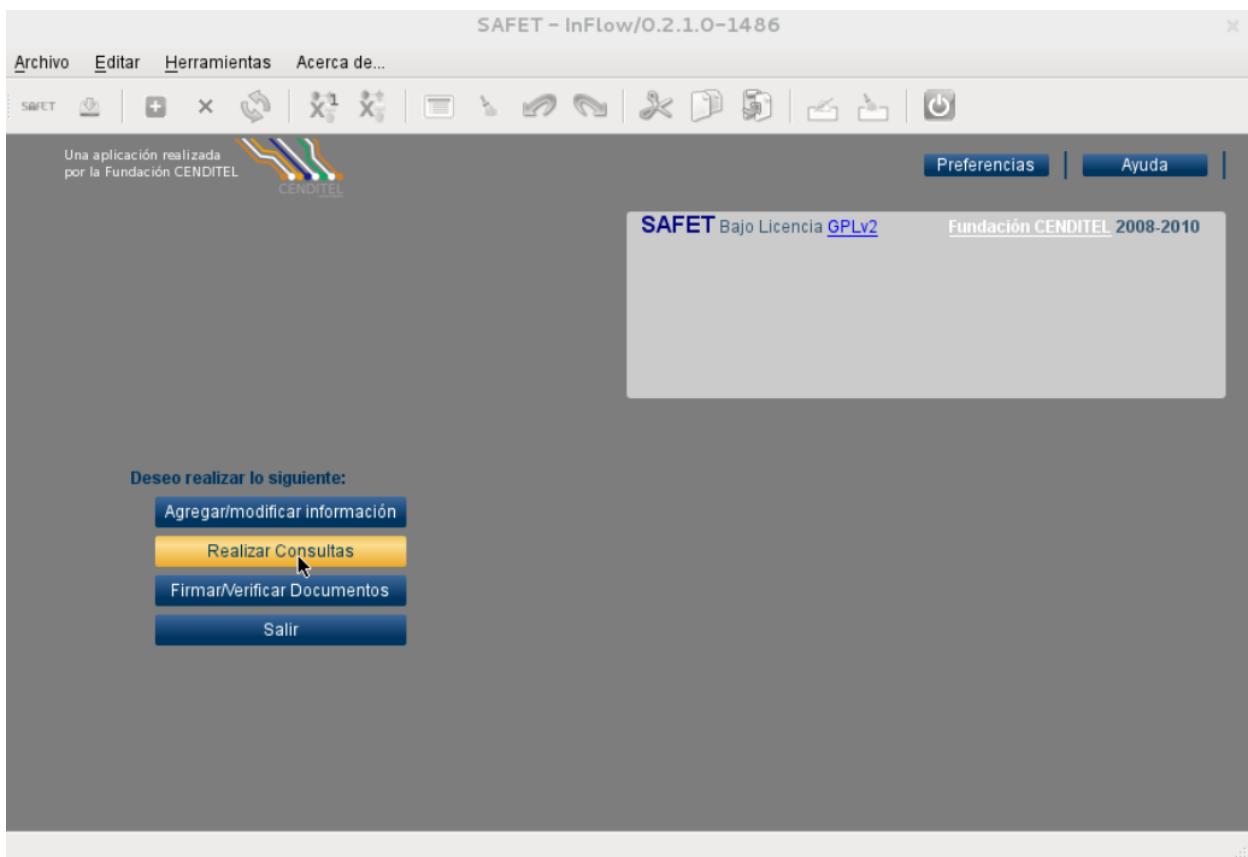


Figura 7.133: **Figura 203: Realizar consultas**

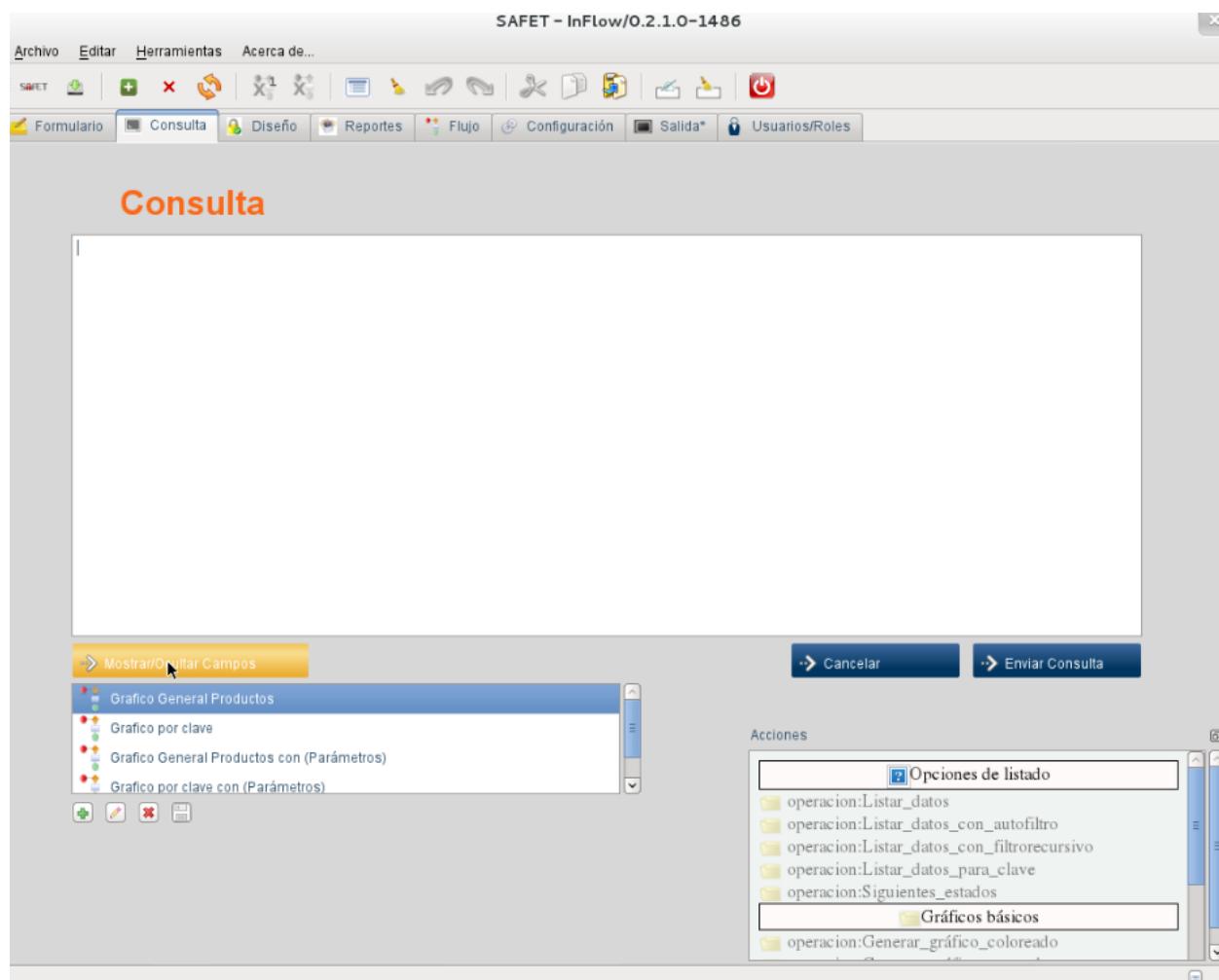


Figura 7.134: Figura 204: Acciones de reportes

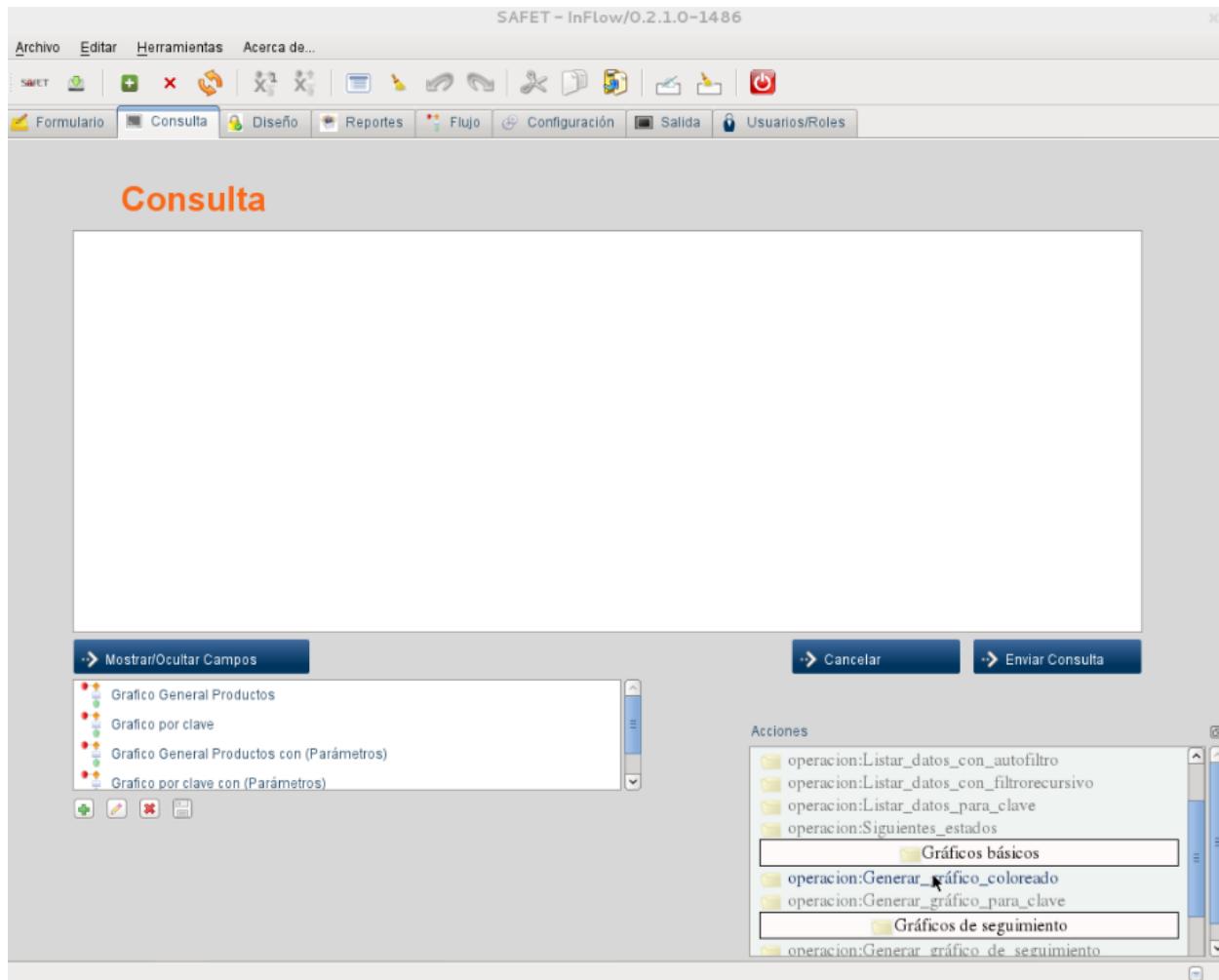


Figura 7.135: Figura 205: Operación (Generar\_gráfico\_coloreado)

### 3° TERCER PASO

- Se le da click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 206: Campo (Cargar\_archivo\_flujo\*)*

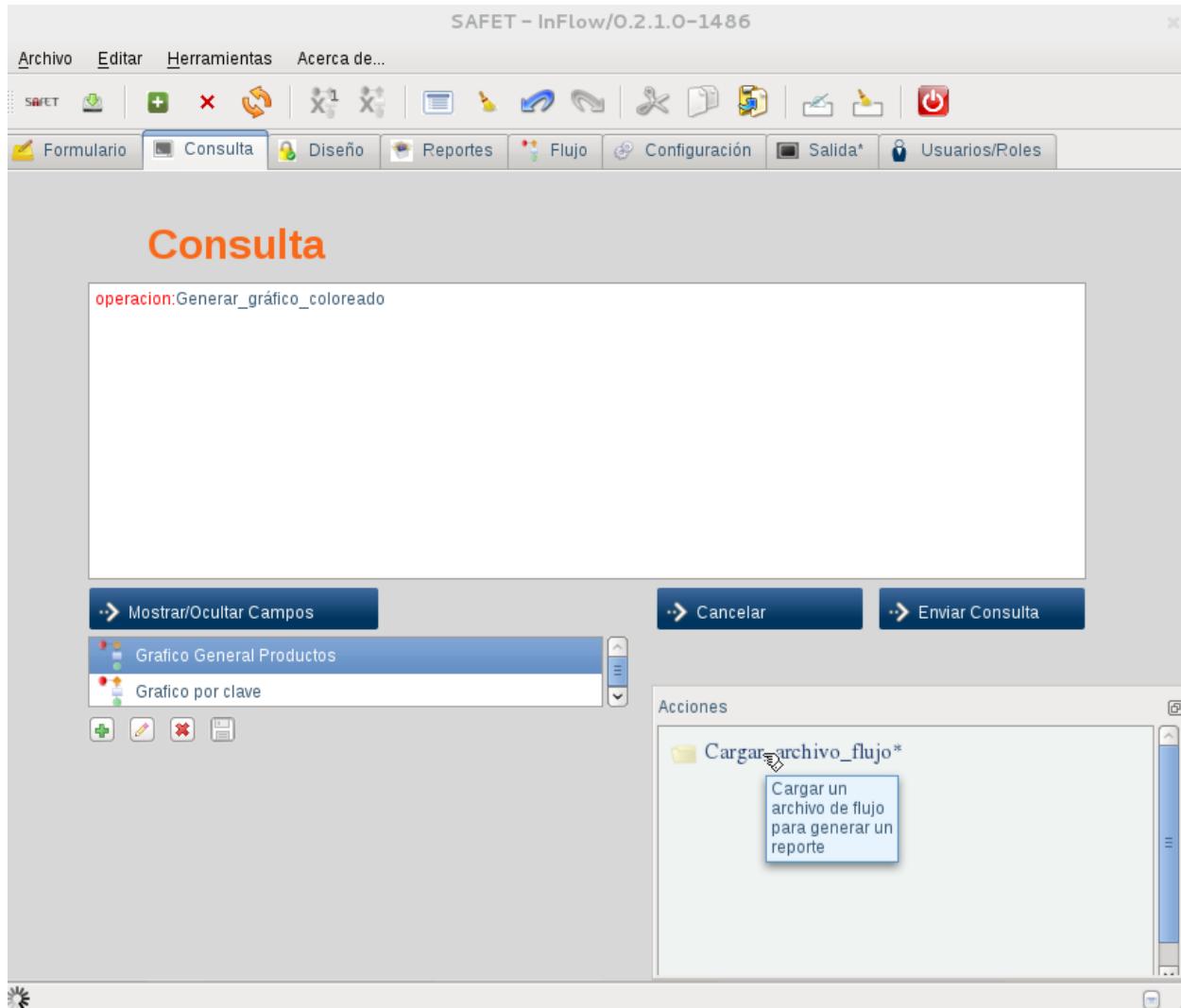


Figura 7.136: **Figura 206: Campo (Cargar\_archivo\_flujo\*)**

### 4° CUARTO PASO

- Se le da click al botón para buscar el archivo (**productos.xml**), como se muestra en la siguiente *Figura 207: Botón (buscar)*

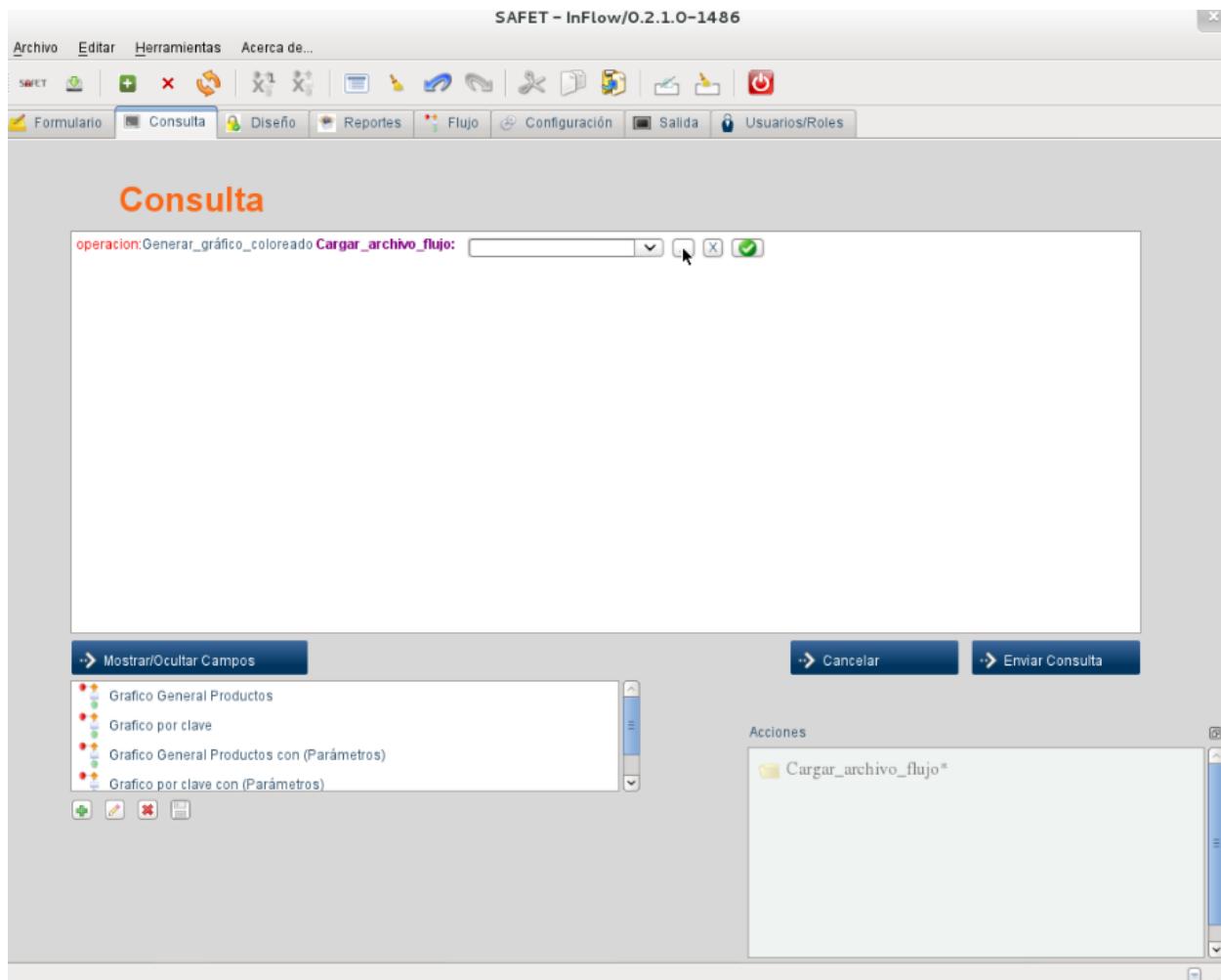


Figura 7.137: **Figura 207: Botón (buscar)**

## 5° QUINTO PASO

- Seleccionamos el archivo (**productos.xml**) del directorio (**/home/usuario/.safet/flowfiles**) y pulsamos en botón (**Abrir**), como se muestra en la siguiente *Figura 208: Seleccionar archivo (.xml)*

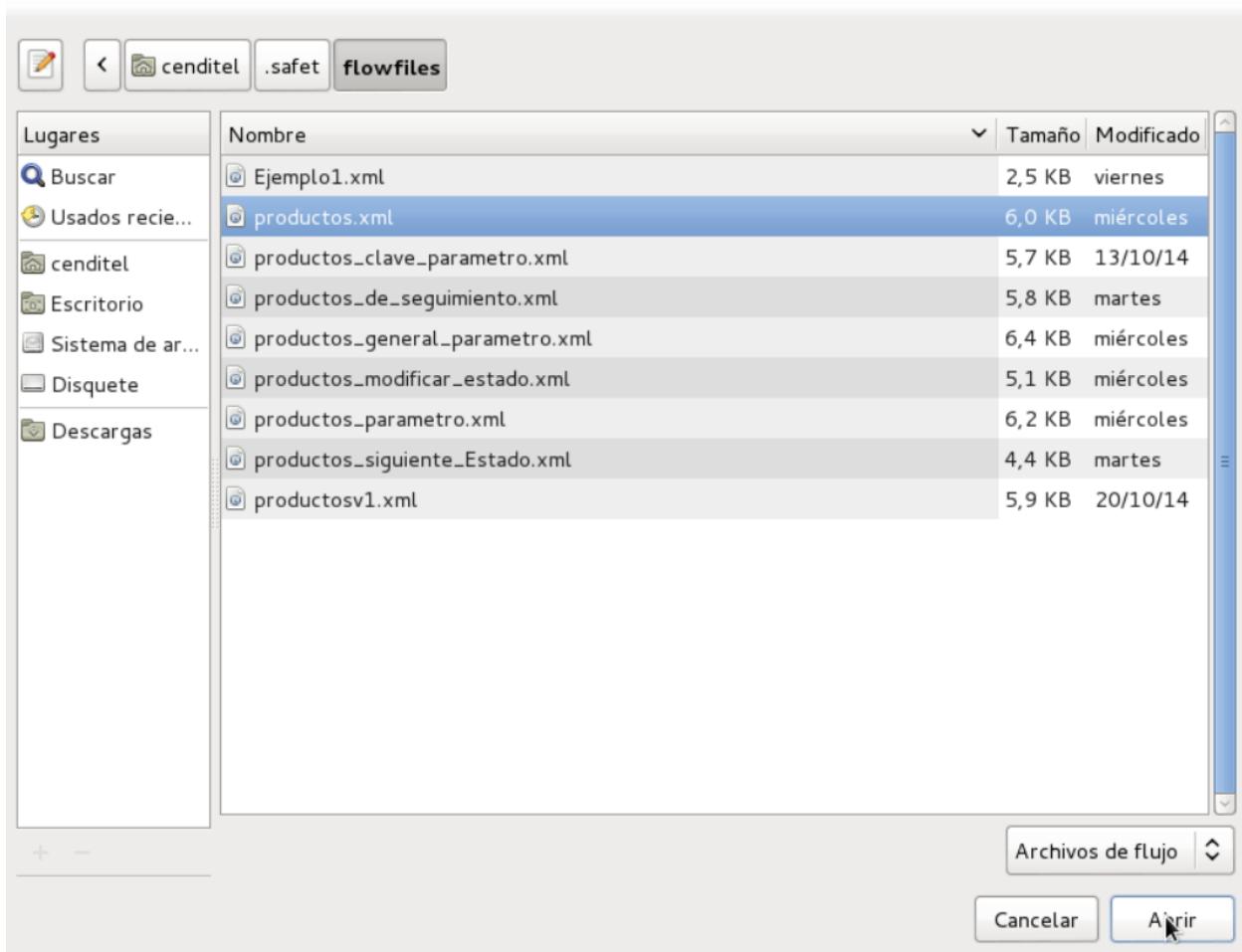


Figura 7.138: *Figura 208: Seleccionar archivo (.xml)*

## 6° SEXTO PASO

- Se le da click al botón de la flecha (**verde**) significando que ya está lleno el campo, como se muestra en la siguiente *Figura 209: Botón (Fin de campo)*

## 7° SEPTIMO PASO

- Pulsamos en el botón (**Enviar consulta**) para finalizar con la operación, se nos mostrara como resultado (**Consulta fue exitosa....ok! (Ver reporte)** ), como se muestra en la siguiente *Figura 210: Botón*

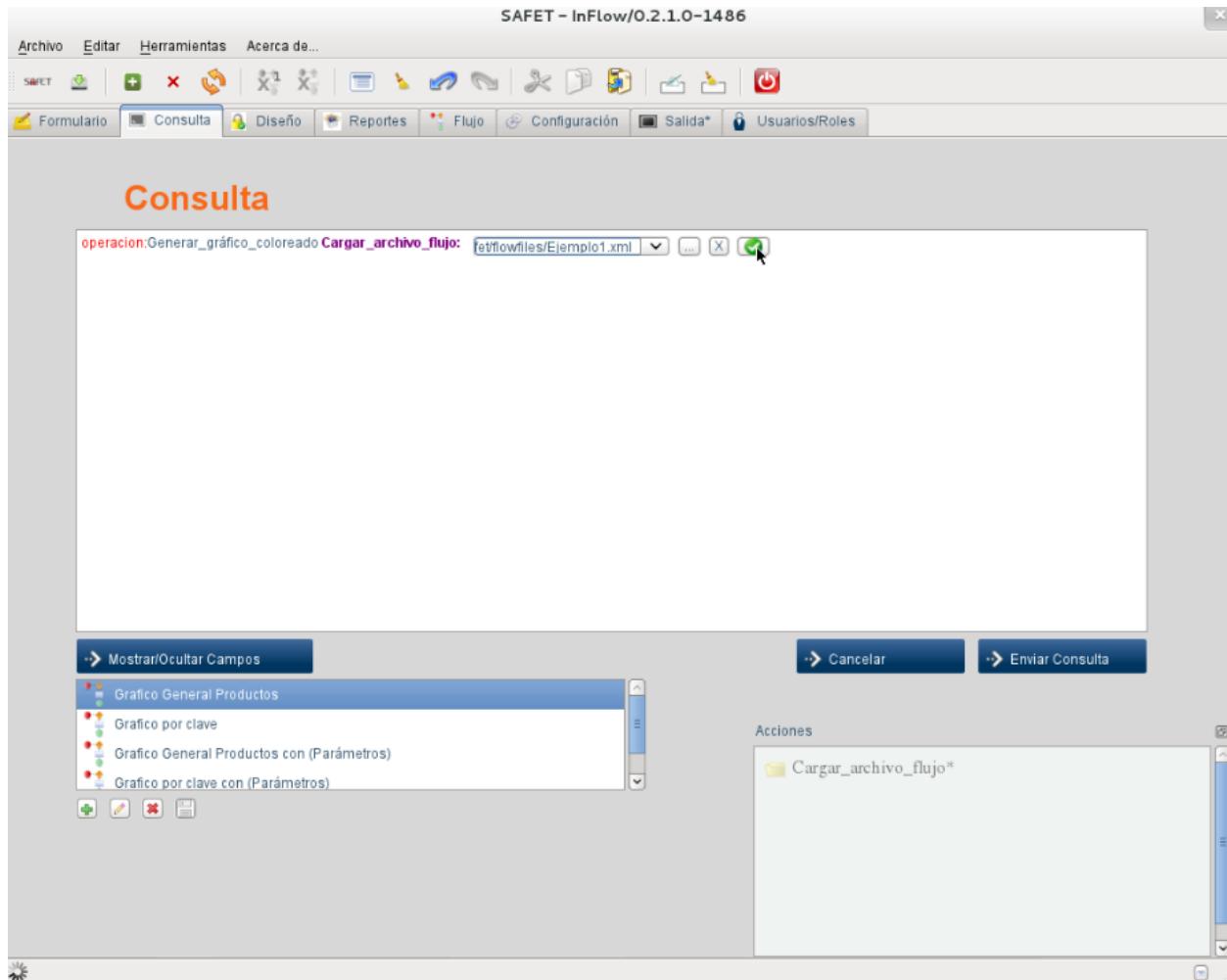


Figura 7.139: Figura 209: Botón (Fin de campo)

(Enviar consulta)

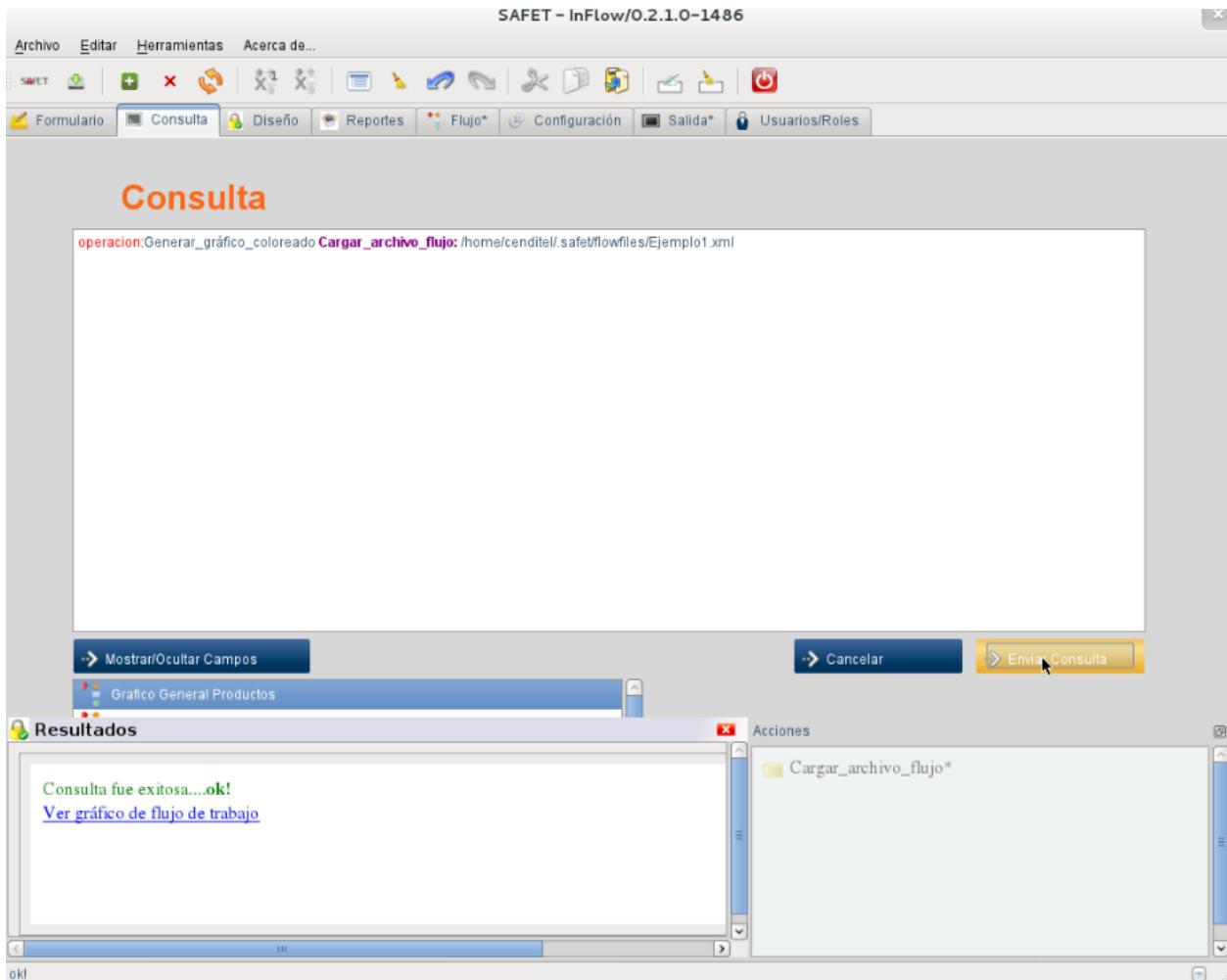


Figura 7.140: Figura 210: Botón (Enviar consulta)

## 8° OCTAVO PASO

- Se le da click al resultado (**Ver gráficos de flujo de trabajo**) para ver el gráfico, como se muestra en la siguiente *Figura 211: Resultado (Ver gráficos de flujo de trabajo)*

---

**Nota:** Se nos mostrara la siguiente gráfica *Figura 212: Gráfica general*

---

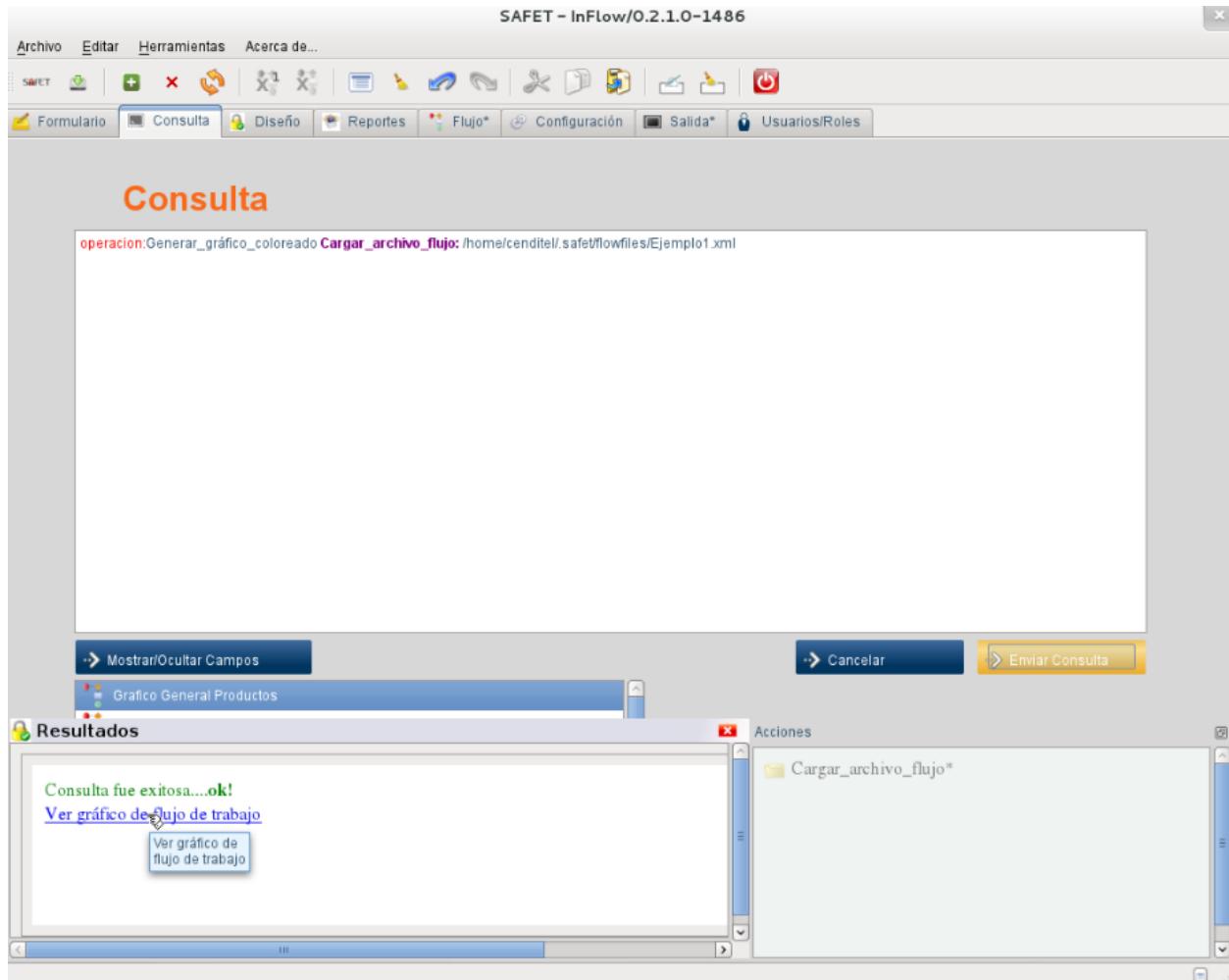


Figura 7.141: Figura 211: Resultado (Ver gráficos de flujo de trabajo)

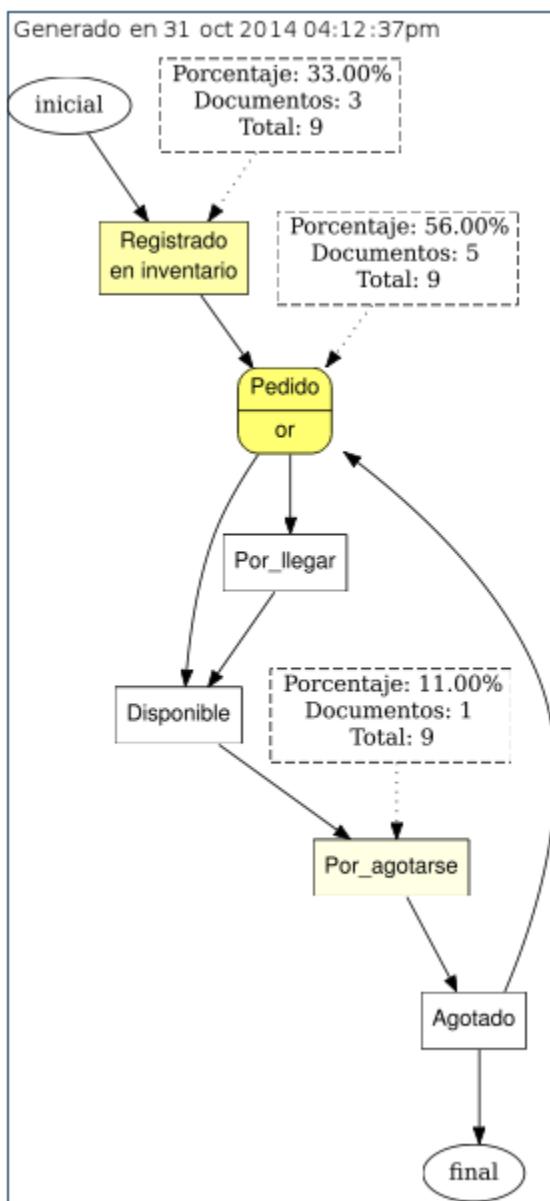


Figura 7.142: Figura 212: Gráfica general

### 7.6.3 C.- SEGUNDA OPERACIÓN (Gráfico de un producto en específico)

#### 1° PRIMER PASO

- Se le da click a la operación (**operacion:Generar\_gráfico\_para\_clave**), como se muestra en la siguiente *Figura 213: Operación (Generar\_gráfico\_para\_clave)*

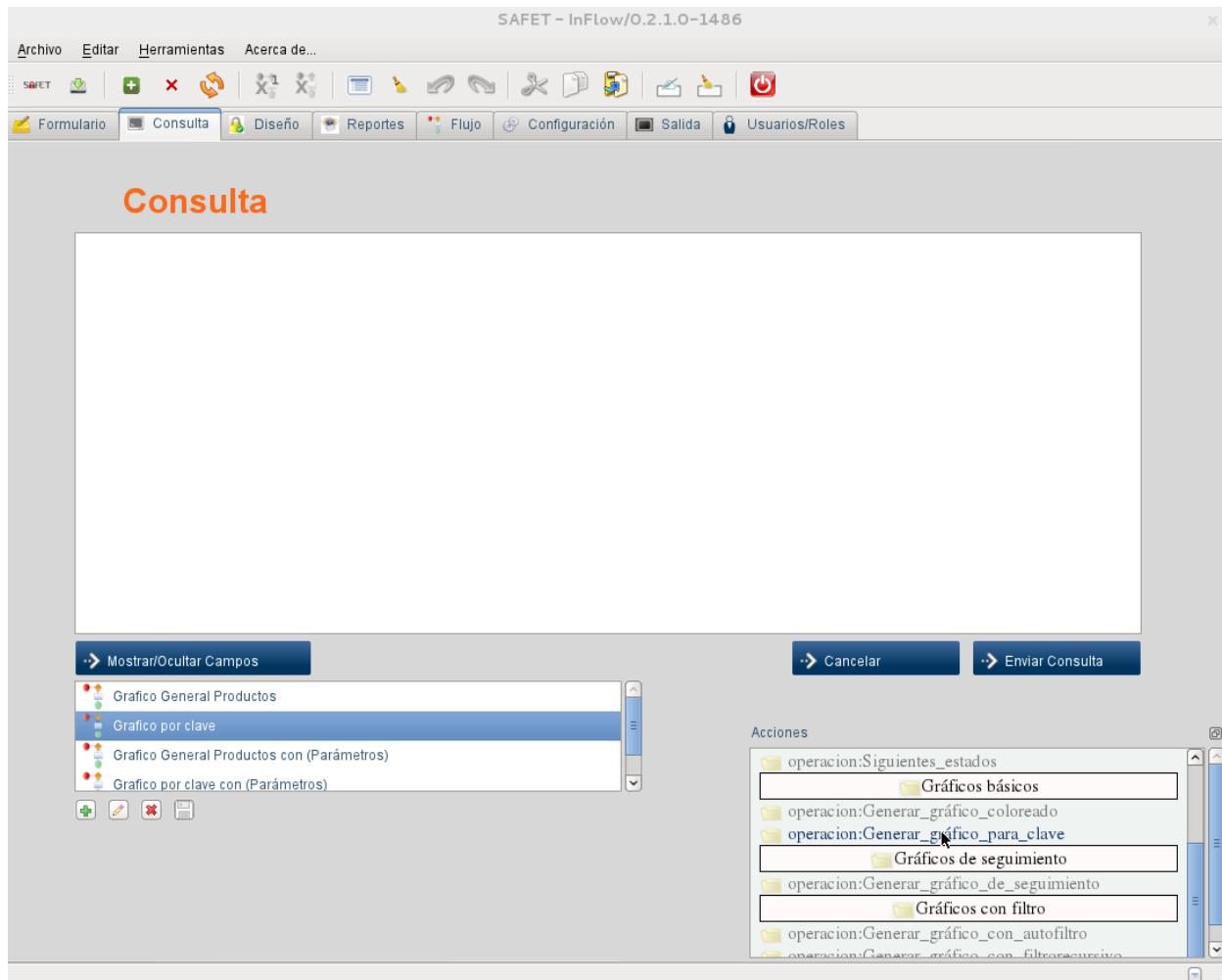


Figura 7.143: **Figura 213: Operación (Generar\_gráfico\_para\_clave)**

#### 2° SEGUNDO PASO

- Se le da click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 214: Campo (Cargar\_archivo\_flujo\*)*

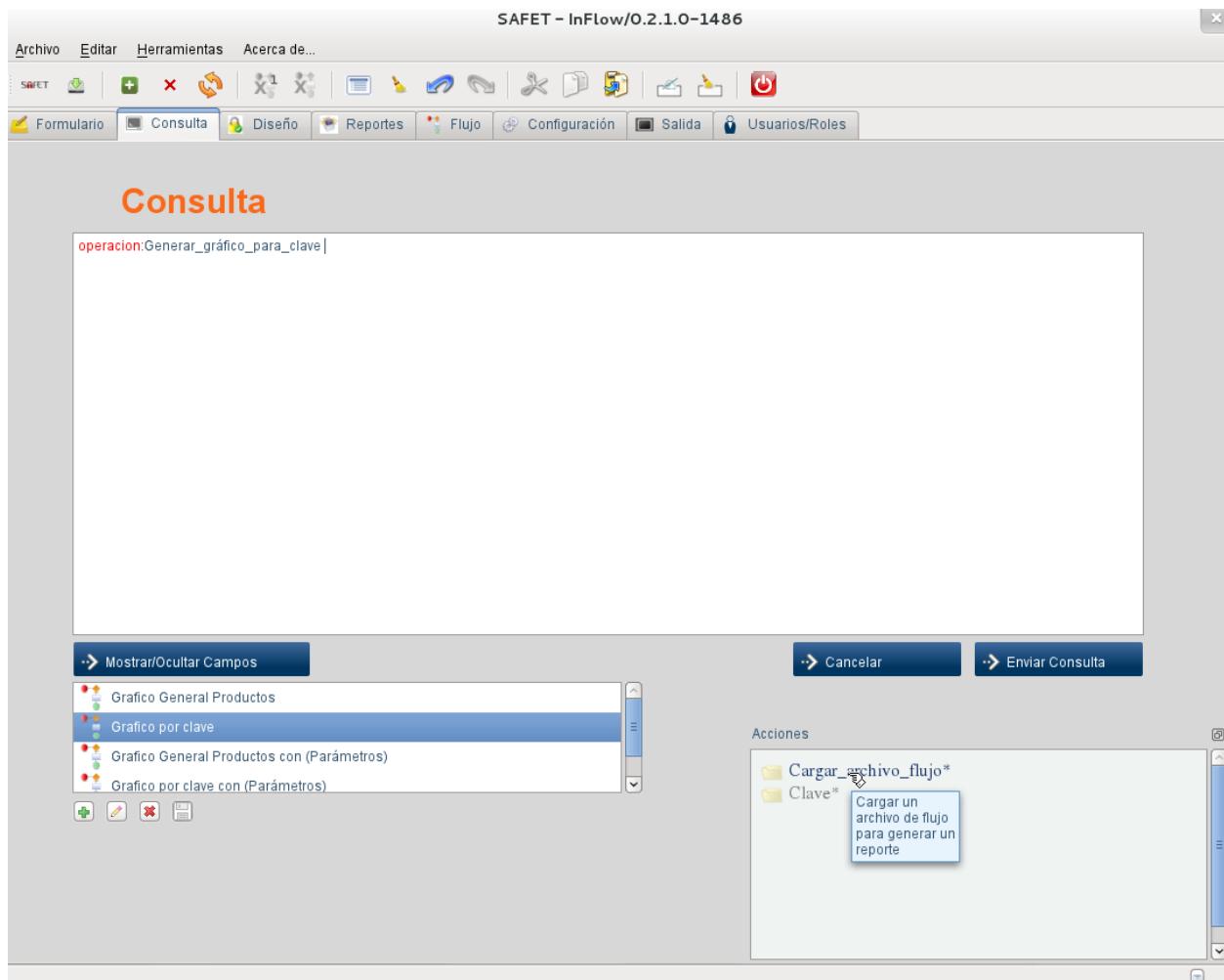


Figura 7.144: **Figura 214: Campo (Cargar\_archivo\_flujo\*)**

### 3° TERCER PASO

- Se le da click en el botón, para buscar el archivo (**productos.xml**), como se muestra en la siguiente *Figura 215: Botón (buscar)*

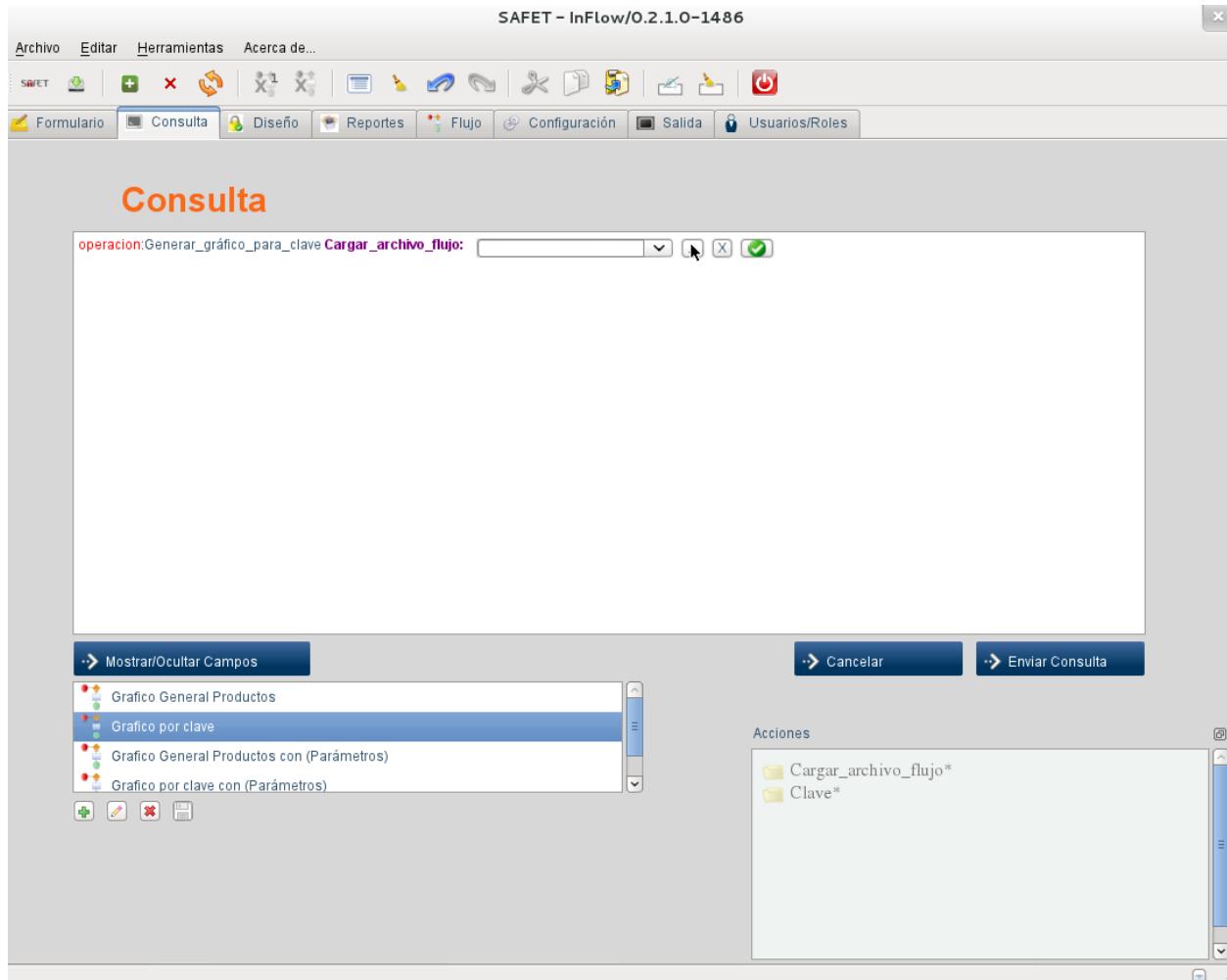


Figura 7.145: **Figura 215: Botón (buscar)**

### 4° CUARTO PASO

- Seleccionamos el archivo (**productos.xml**) del directorio (**/home/usuario/.safet/flowfiles**) y pulsamos en botón (**Abrir**), como se muestra en la siguiente *Figura 216: Seleccionar archivo (.xml)*

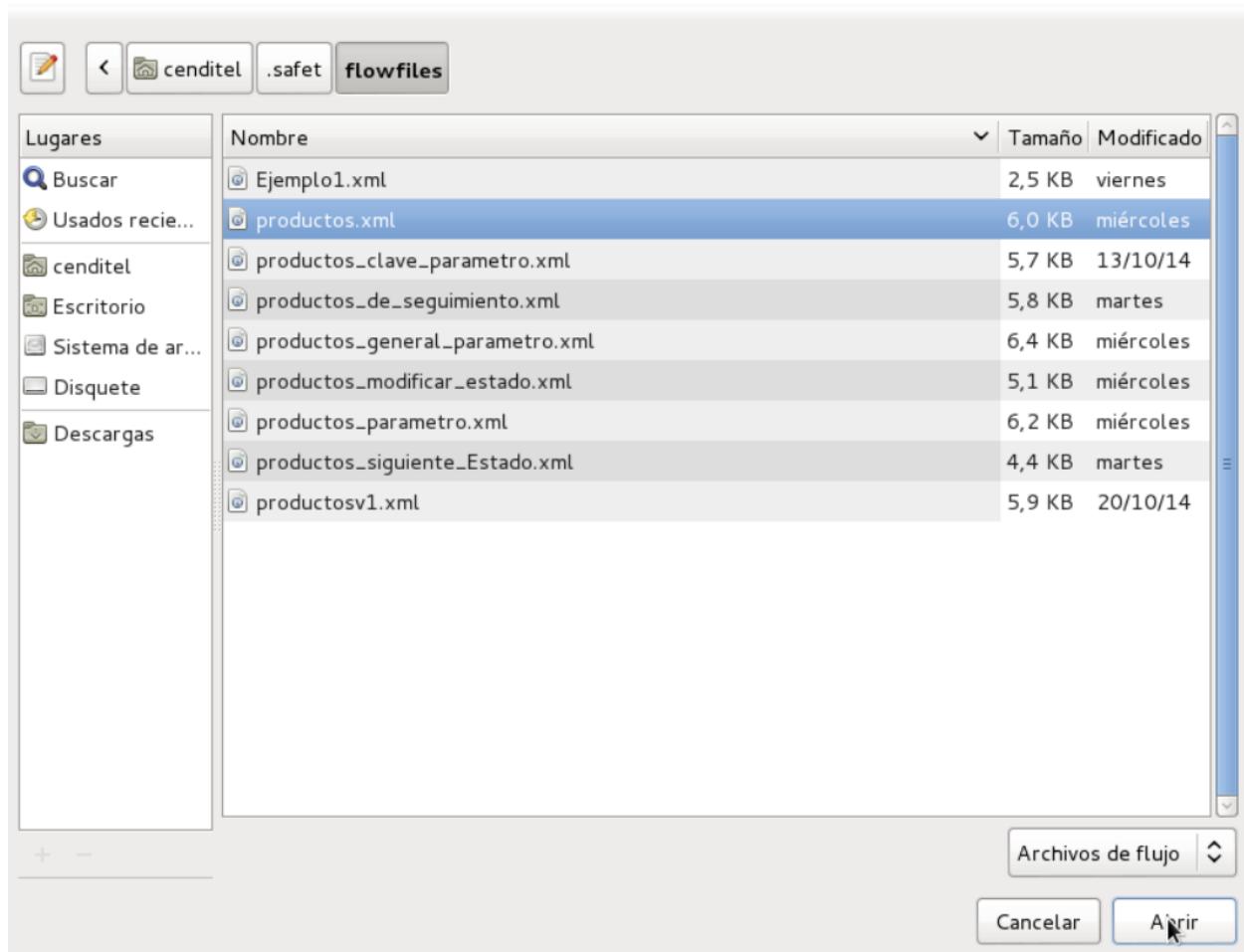


Figura 7.146: Figura 216: Seleccionar archivo (.xml)

### 5° QUINTO PASO

- Se le da click en el **botón con la flecha (verde)** significando que ya está lleno el campo, como se muestra en la siguiente *Figura 217: Botón (Fin de campo)*

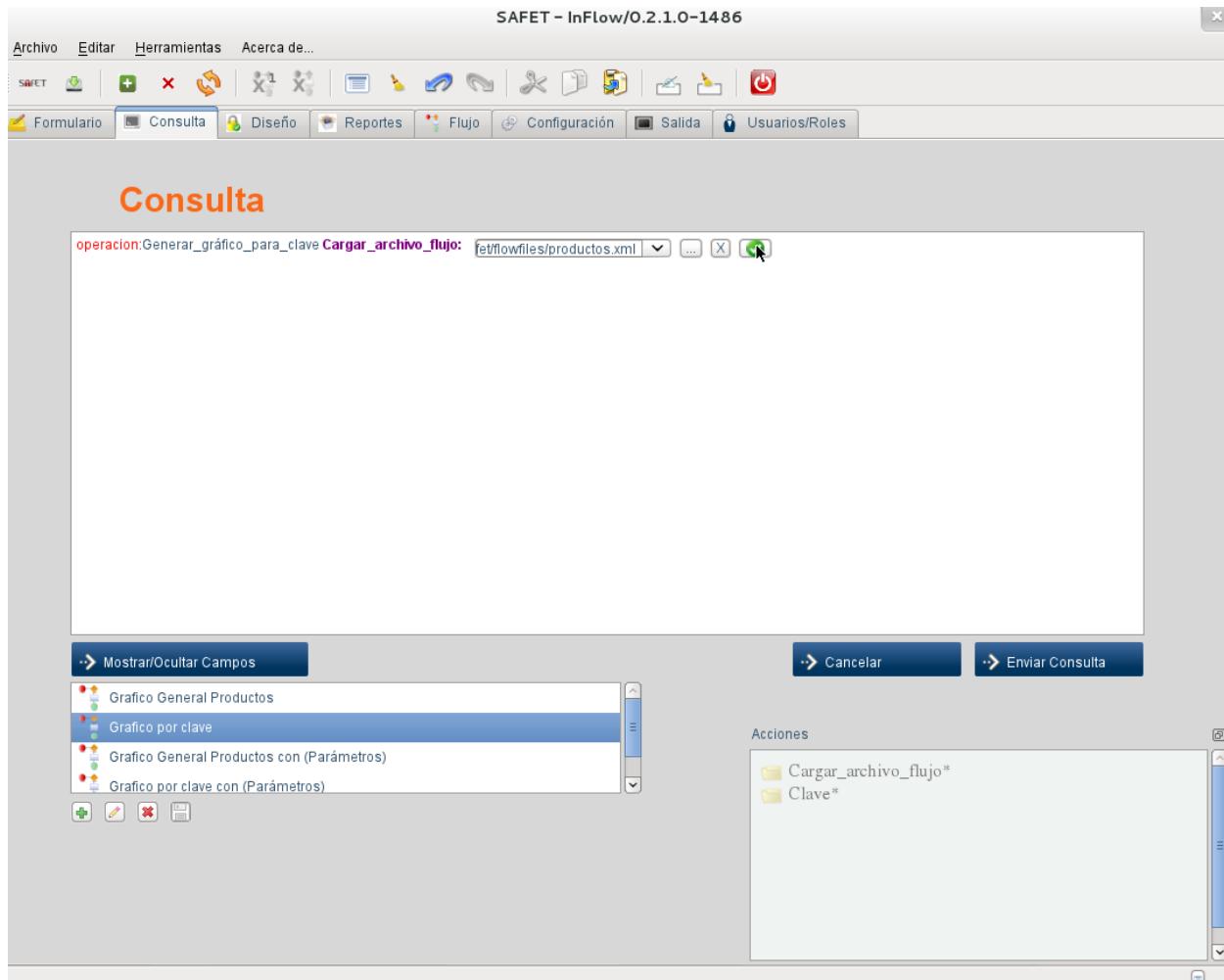


Figura 7.147: **Figura 217: Botón (Fin de campo)**

### 6° SEXTO PASO

- Se le da click en el campo obligatorio (**Clave\***), como se muestra en la siguiente *Figura 218: Campo (Clave\*)*

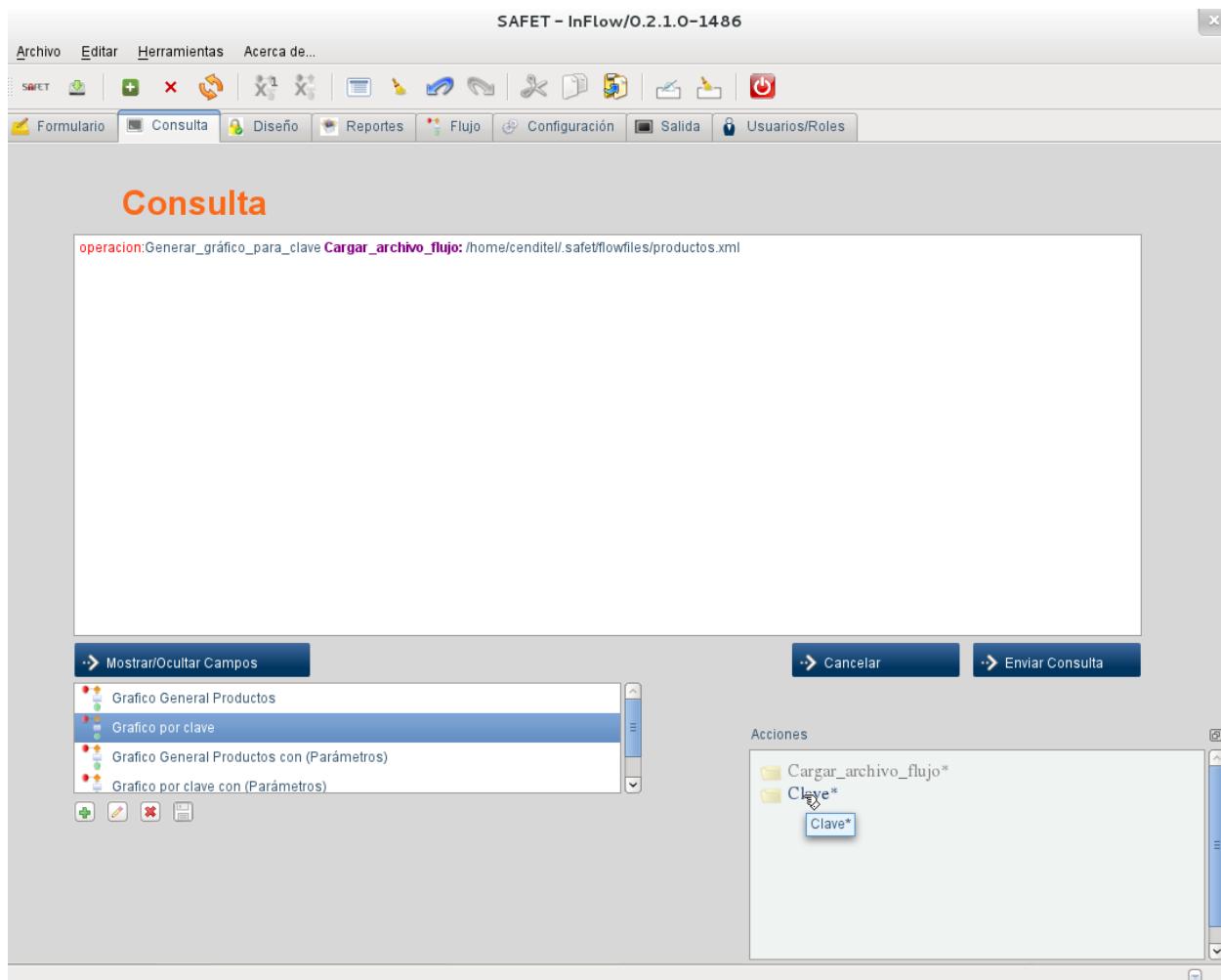


Figura 7.148: **Figura 218: Campo (Clave\*)**

### 7° SEPTIMO PASO

- Seleccionamos el producto a **mostrar** en el gráfico, por ejemplo ((6)-**VitaminaB8**), como se muestra en la siguiente *Figura 219: Selección del producto*

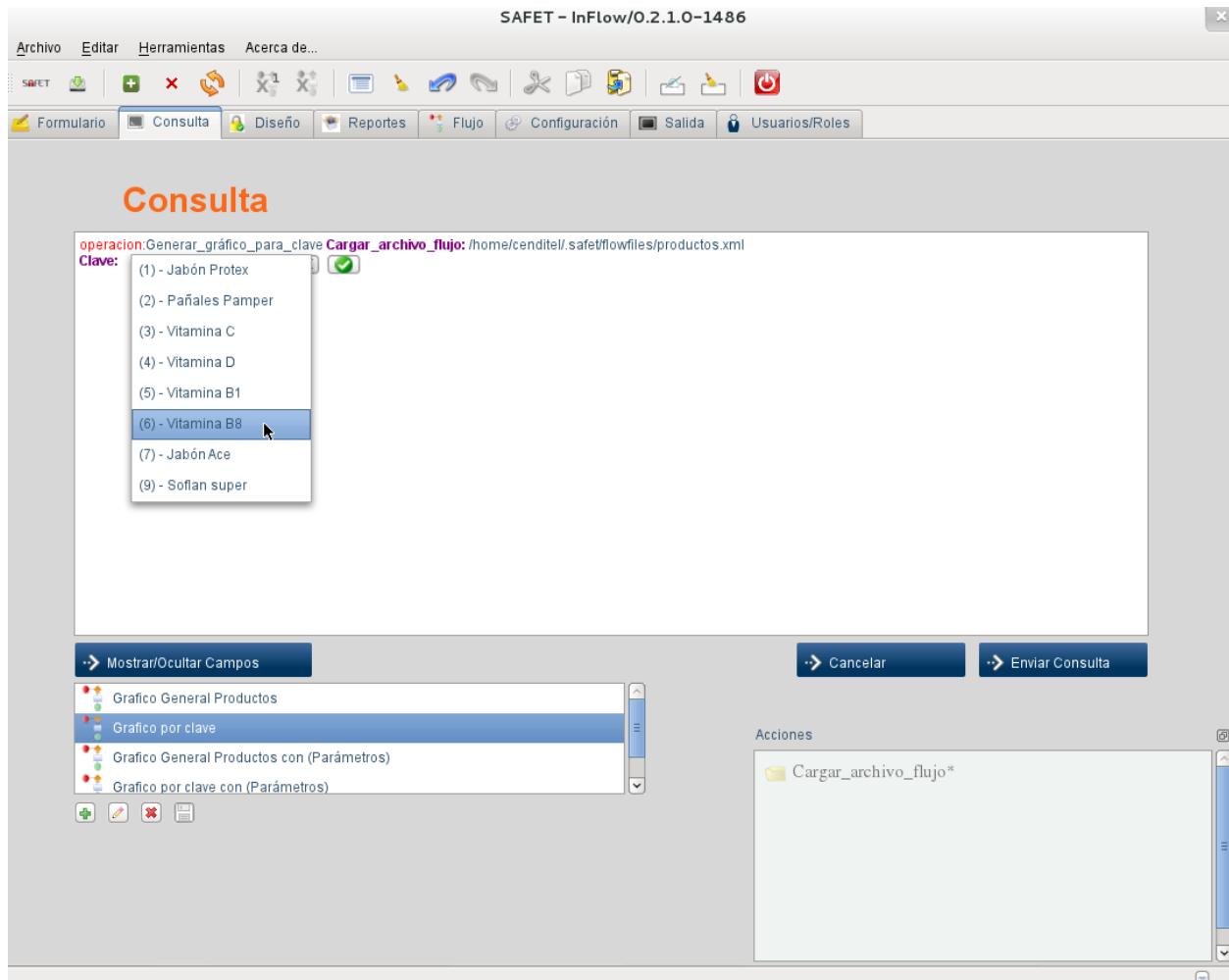


Figura 7.149: **Figura 219: Selección del producto**

### 8° OCTAVO PASO

- Se le da click en el **botón** con la flecha (**verde**) significando que ya está lleno el campo, como se muestra en la siguiente *Figura 230: Botón (Fin de campo)*

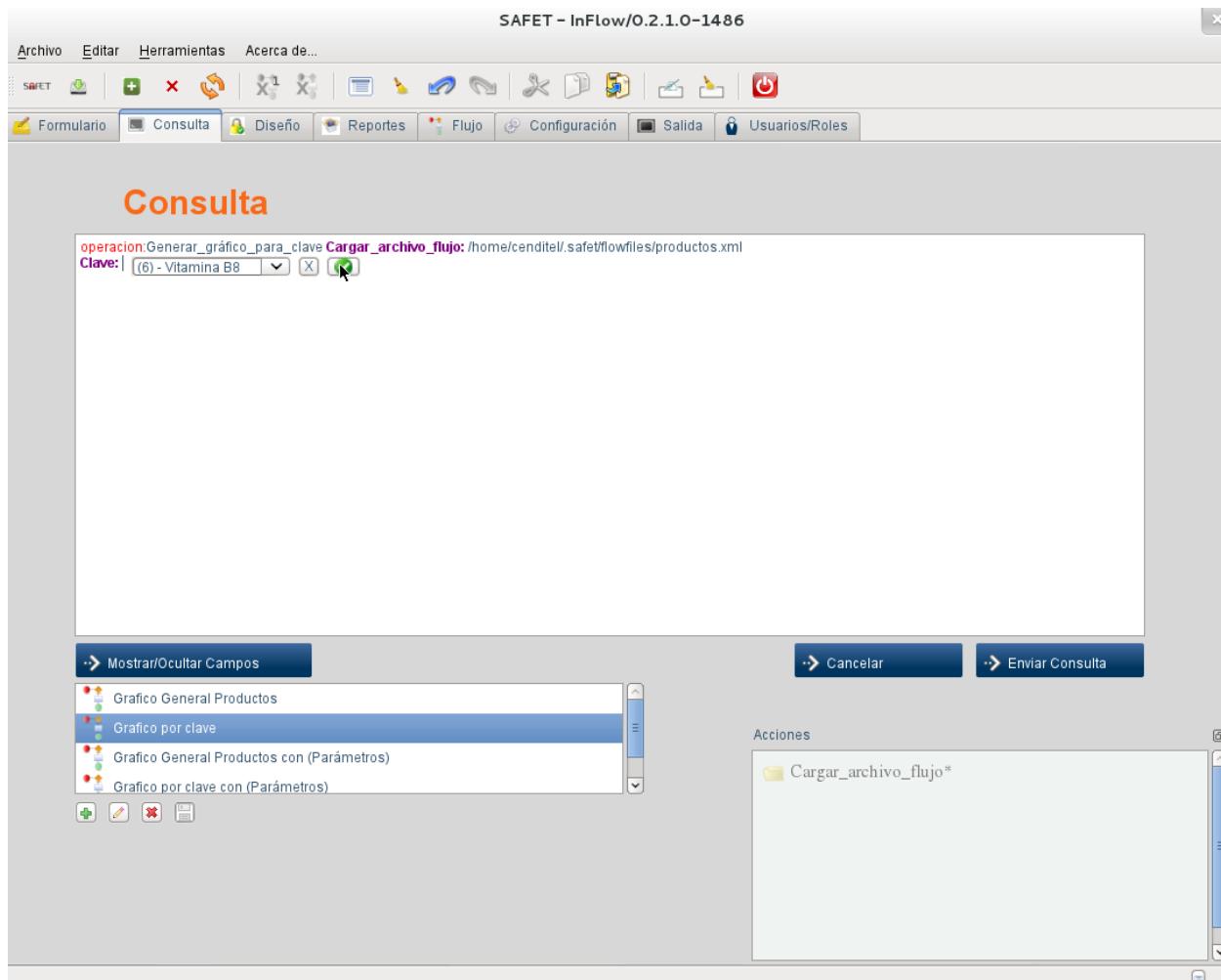


Figura 7.150: Figura 230: Botón (Fin de campo)

### 9° NOVENO PASO

- Pulsamos en el botón (**Enviar consulta**) para finalizar con la operación, se nos mostrara como resultado (**Consulta fue exitosa....ok!** (Ver reporte) ), como se muestra en la siguiente *Figura 231: Botón (Enviar consulta)*

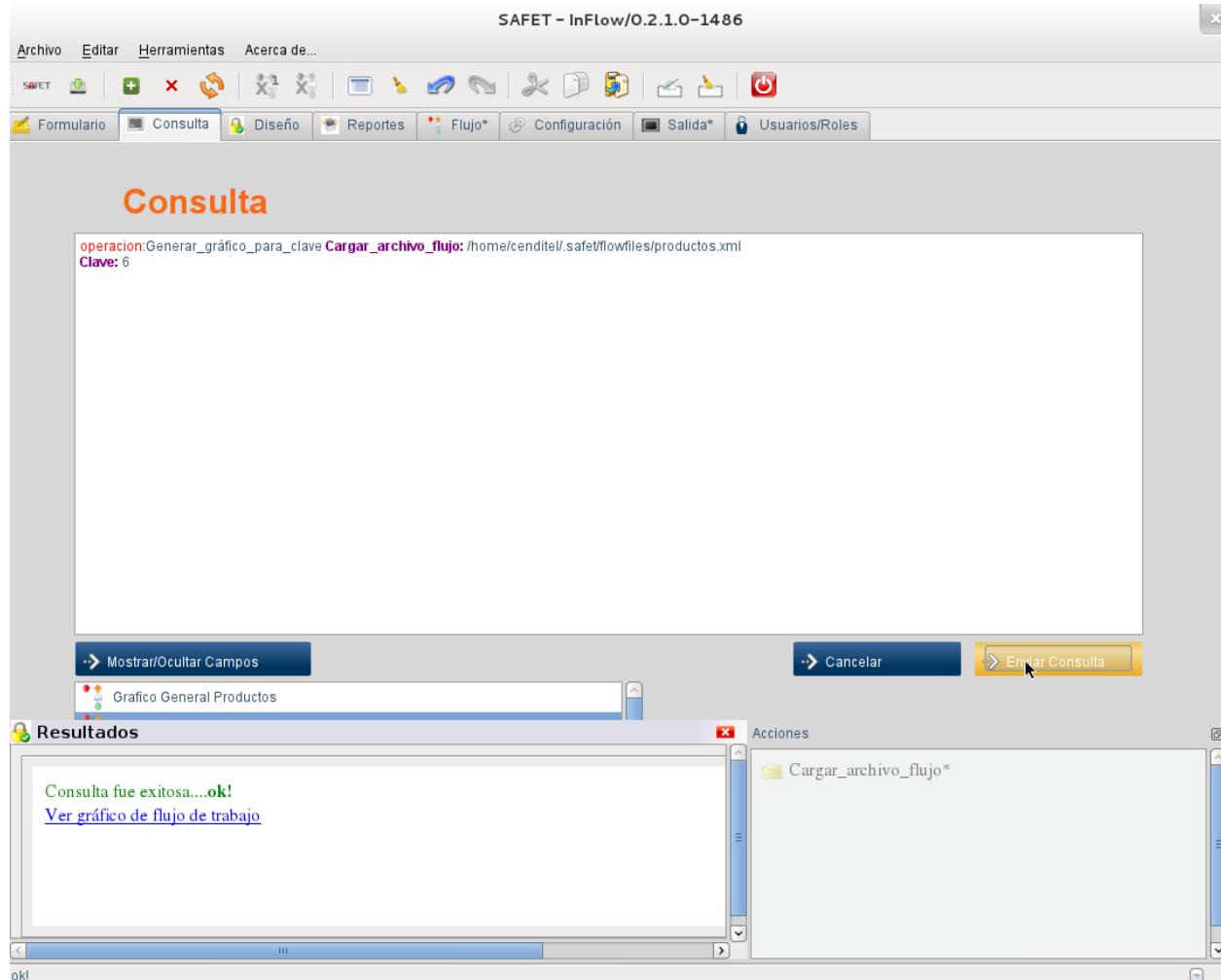


Figura 7.151: **Figura 231: Botón (Enviar consulta)**

### 10° DECIMO PASO

- Se le da click al resultado (**Ver gráficos de flujo de trabajo**) para ver el gráfico, como se muestra en la siguiente *Figura 232: Resultado (Ver gráficos de flujo de trabajo)*

**Nota:** Se nos mostrara la siguiente gráfica *Figura 233: Gráfica del un producto en específico*

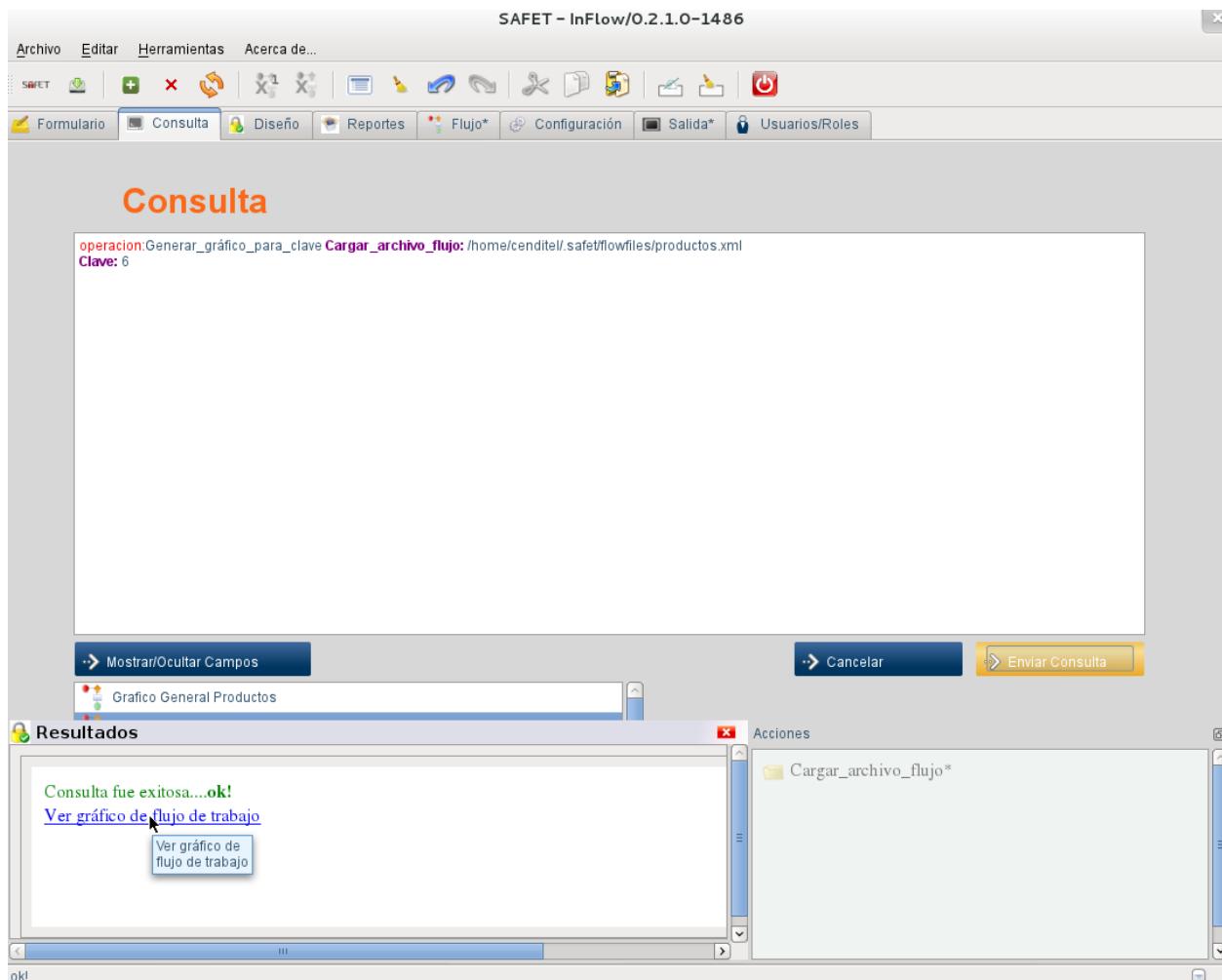


Figura 7.152: Figura 232: Resultado (Ver gráficos de flujo de trabajo)

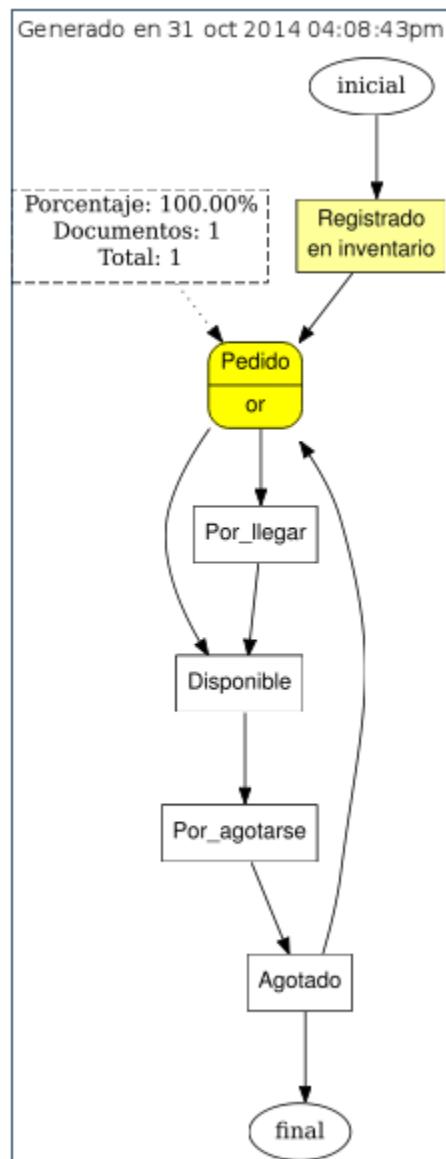


Figura 7.153: Figura 233: Gráfica del un producto en específico

## 7.7 Ejecución de gráficos usando flujos de trabajos con parámetros

En este tutorial explicaremos el **ejemplo de inventario**, la cual generaremos gráficos de flujo de trabajo de manera específico, para ello debemos tener instalado inflow, sino damos un click al siguiente enlace. Instalación de la interfaz gráfica (inflow).

A continuación seguimos los siguientes pasos la cual abriremos inflow y luego ejecutaremos la acciones:

### 7.7.1 A.- Editar el archivo (`productos.xml`)

En el paso anterior, insertamos los autofiltros la cual vamos a utilizar en este tutorial. Si no has insertado los autofiltros damos click al siguiente enlace. [A.- Editamos el archivo \(`productos.xml`\)](#)

### 7.7.2 B.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 234: Autenticación de Inflow*.

#### 3° TERCER PASO

- Damos click a la segunda opción (**Realizar consultas**), como se muestra en la siguiente *Figura 235: Realizar consultas*

### 7.7.3 C.- PRIMERA OPERACIÓN (Gráfico general productos con (Parámetros))

#### 1° PRIMER PASO

- Damos click a la operación (**Grafico general Productos con (Parámetros)**), como se muestra en la siguiente *Figura 236: Operación (Grafico general Productos con (Parámetros))*



Figura 7.154: Figura 234: Autenticación de Inflow.

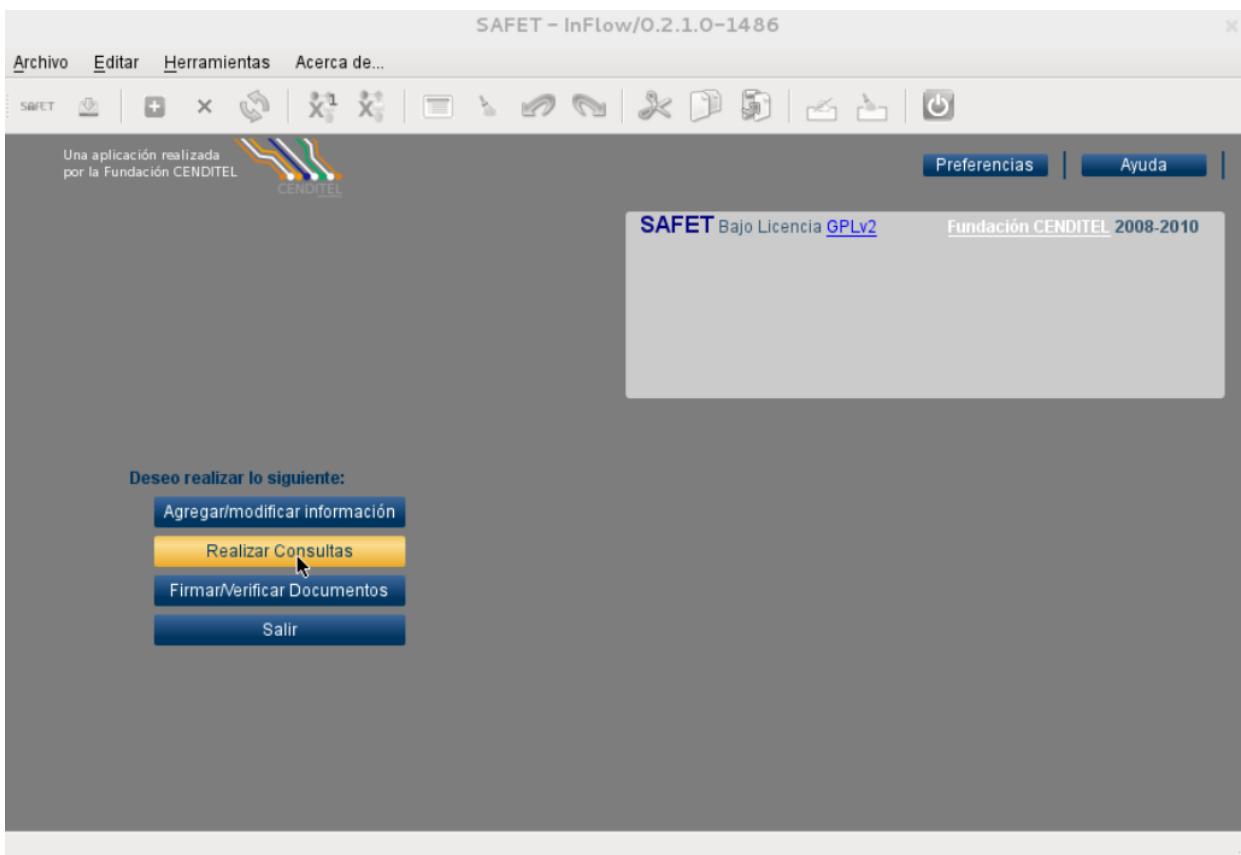


Figura 7.155: Figura 235: Realizar consultas

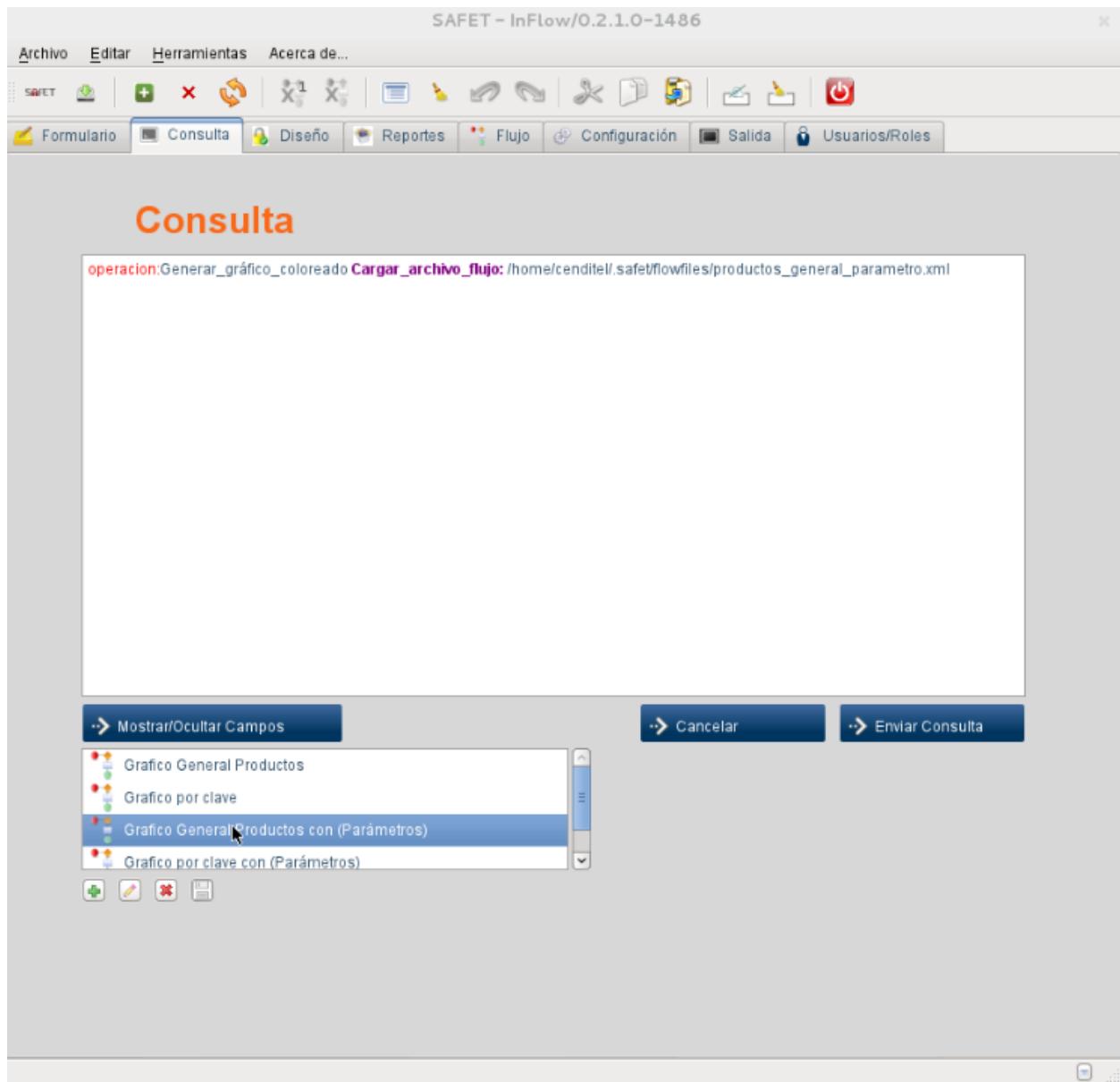


Figura 7.156: Figura 236: Operación (Grafico general Productos con (Parámetros))

## 2° SEGUNDO PASO

- Damos click al botón (**modificar campo**), como se muestra en la siguiente *Figura 237: Botón (modificar campo)*

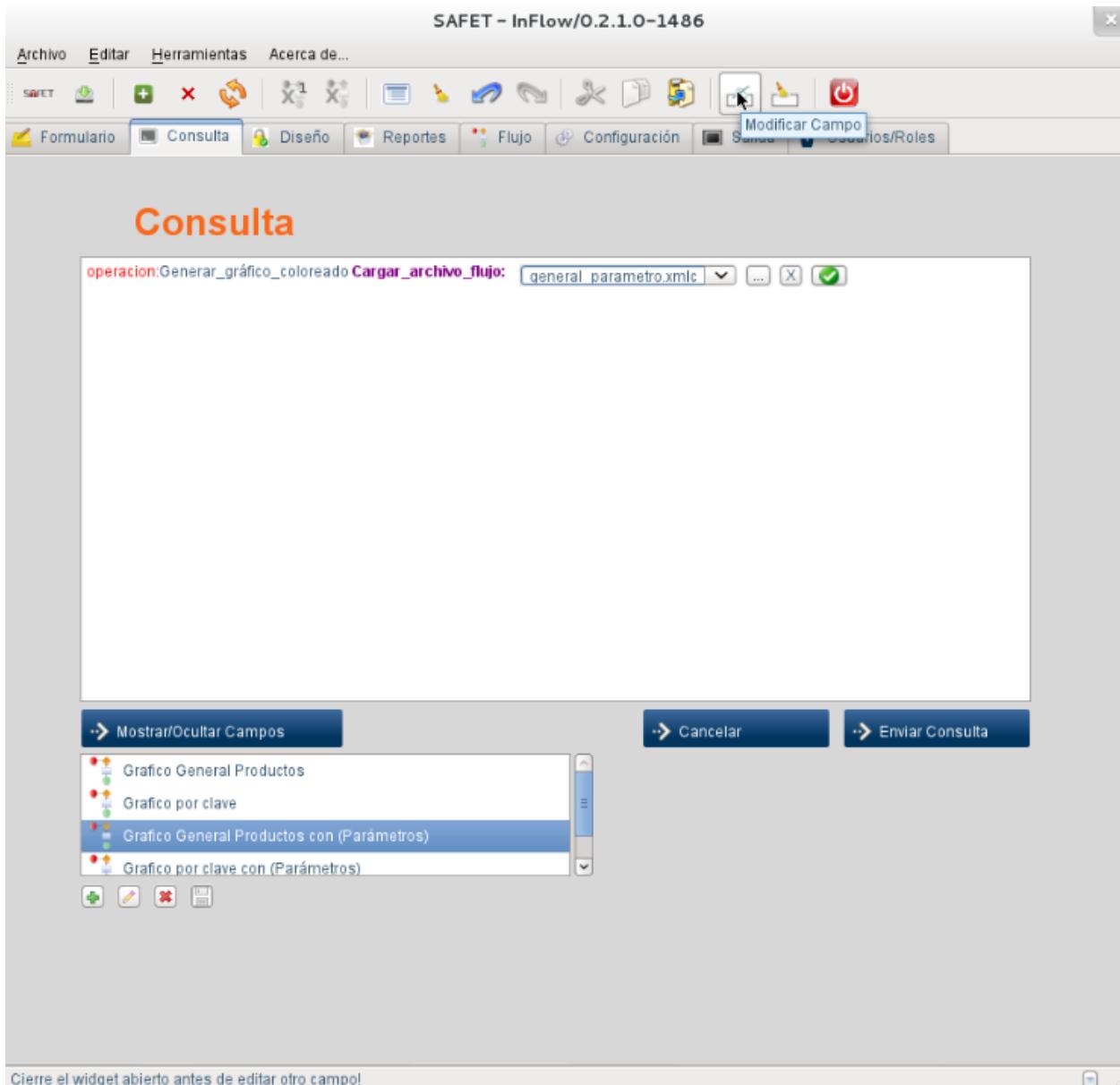


Figura 7.157: **Figura 237: Botón (modificar campo)**

### 3° TERCER PASO

- Damos click al botón (**buscar**) para buscar el archivo (**productos\_general\_parametro.xml**), como se muestra en la siguiente *Figura 238: Botón (buscar)*

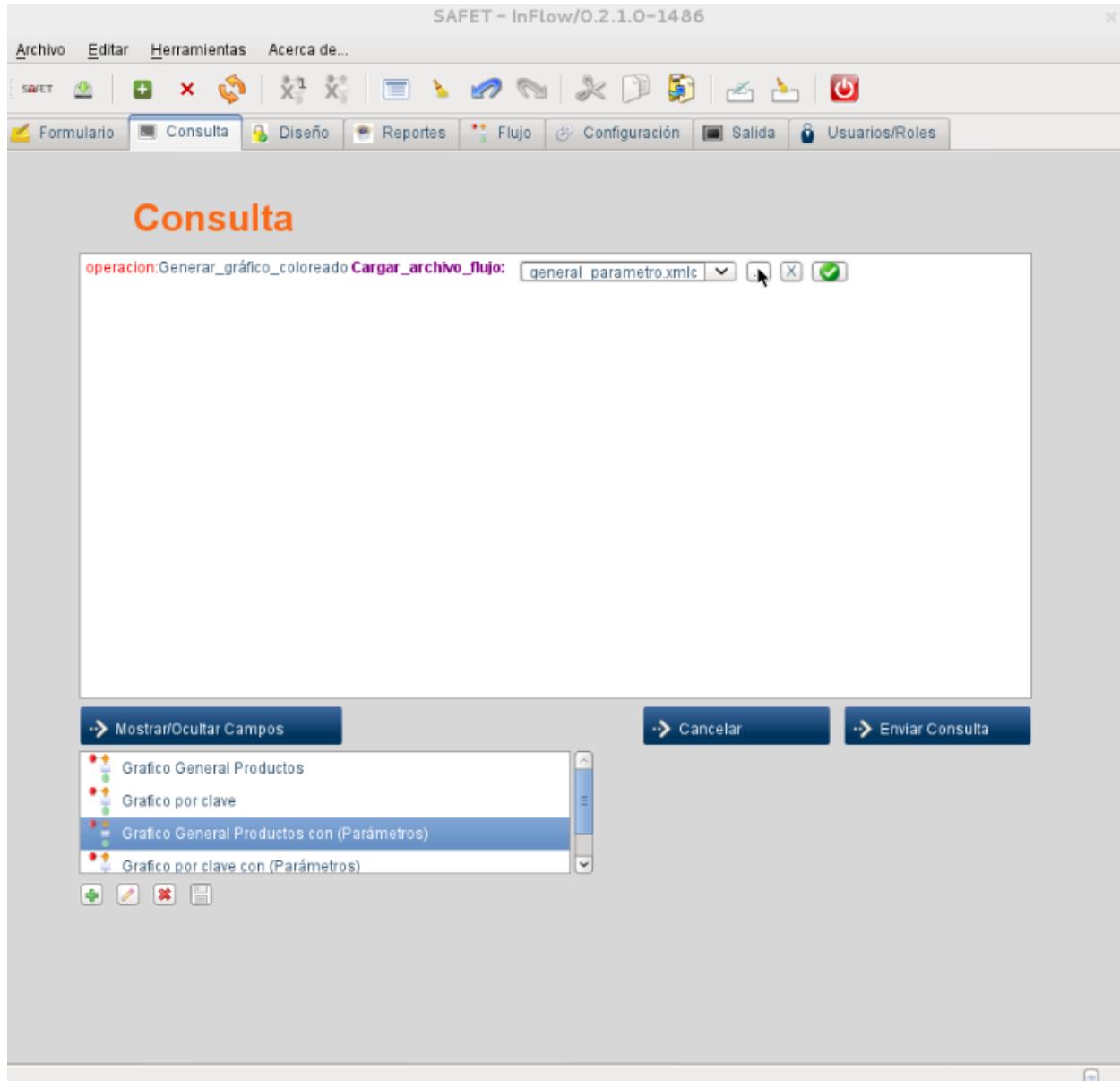


Figura 7.158: **Figura 238: Botón (buscar)**

### 4° CUARTO PASO

- Seleccionamos el archivo (**productos\_general\_parametro.xml**) del directorio (**/home/usuario/.safet/flowfiles**) y pulsamos en botón (**Abrir**), como se muestra en la siguiente

Figura 239: Seleccionar archivo (xml)

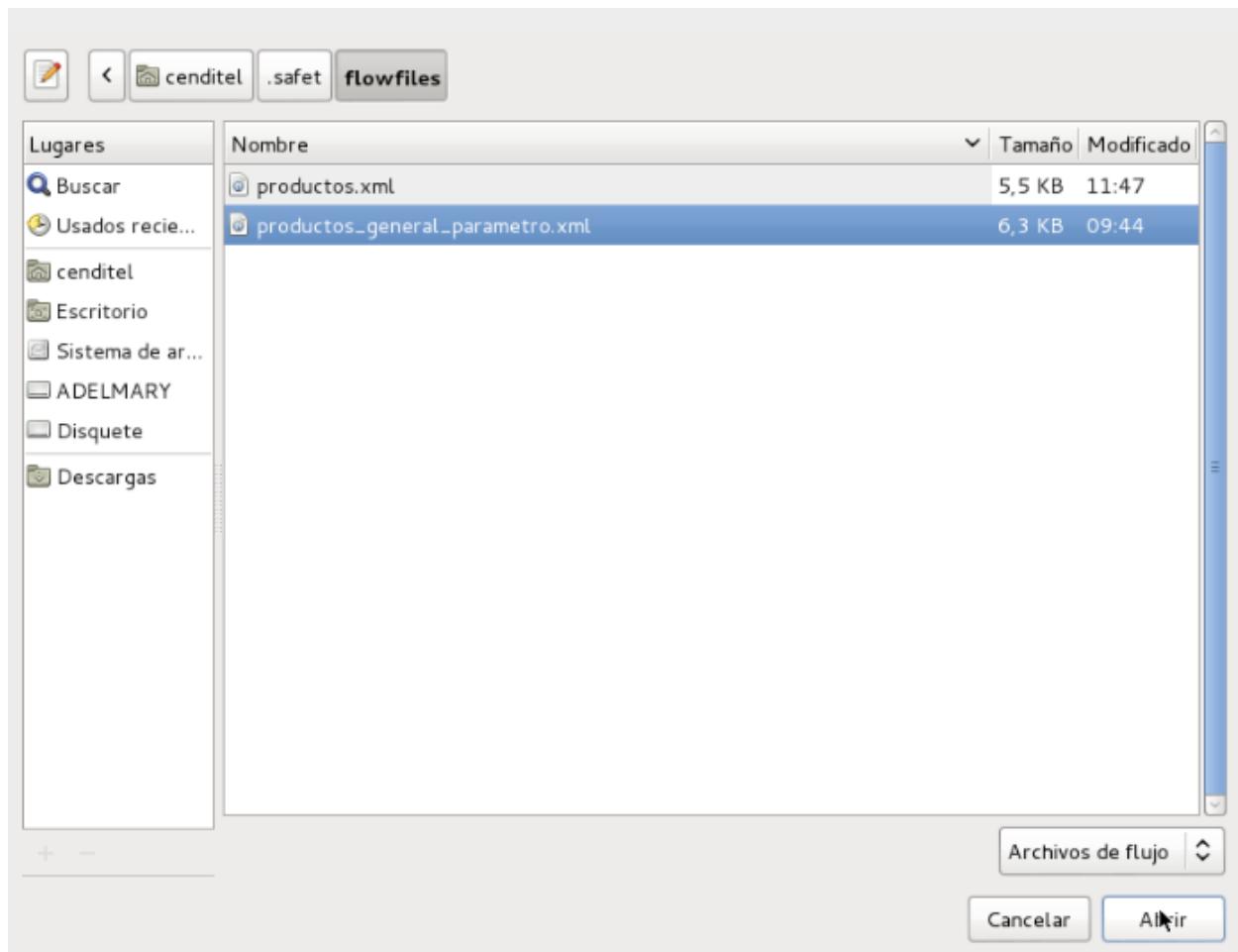


Figura 7.159: Figura 239: Seleccionar archivo (xml)

## 5° QUINTO PASO

- Damos un click al botón con la flecha (**verde**) significando que ya está lleno el campo, como se muestra en la siguiente [Figura 240: Botón \(Fin de campo\)](#)

## 6° SEXTO PASO

- Pulsamos en el botón (**Enviar consulta**) para finalizar con la operación, como se muestra en la siguiente [Figura 241: Botón \(Enviar consulta\)](#)

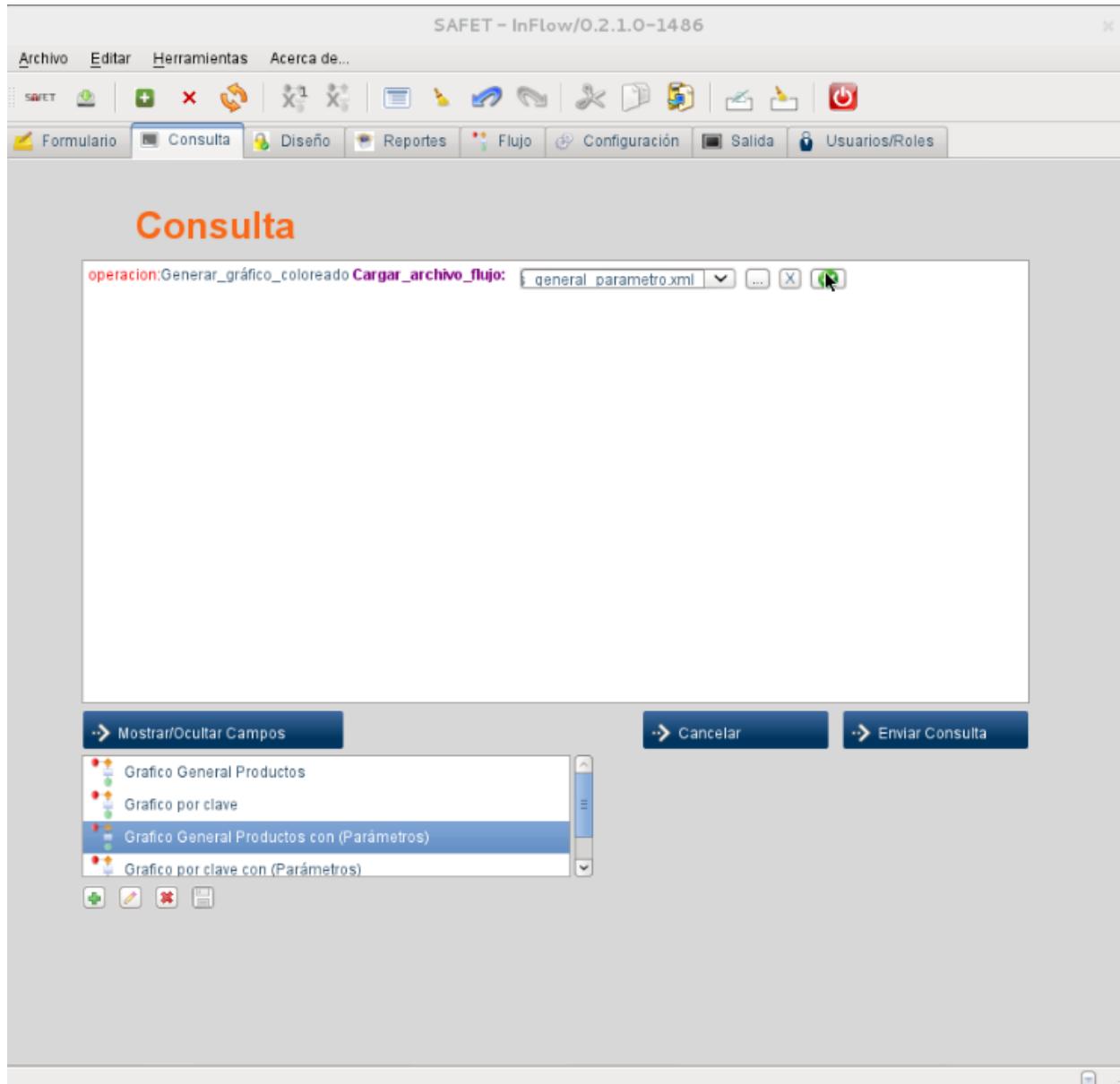


Figura 7.160: Figura 240: Botón (Fin de campo)

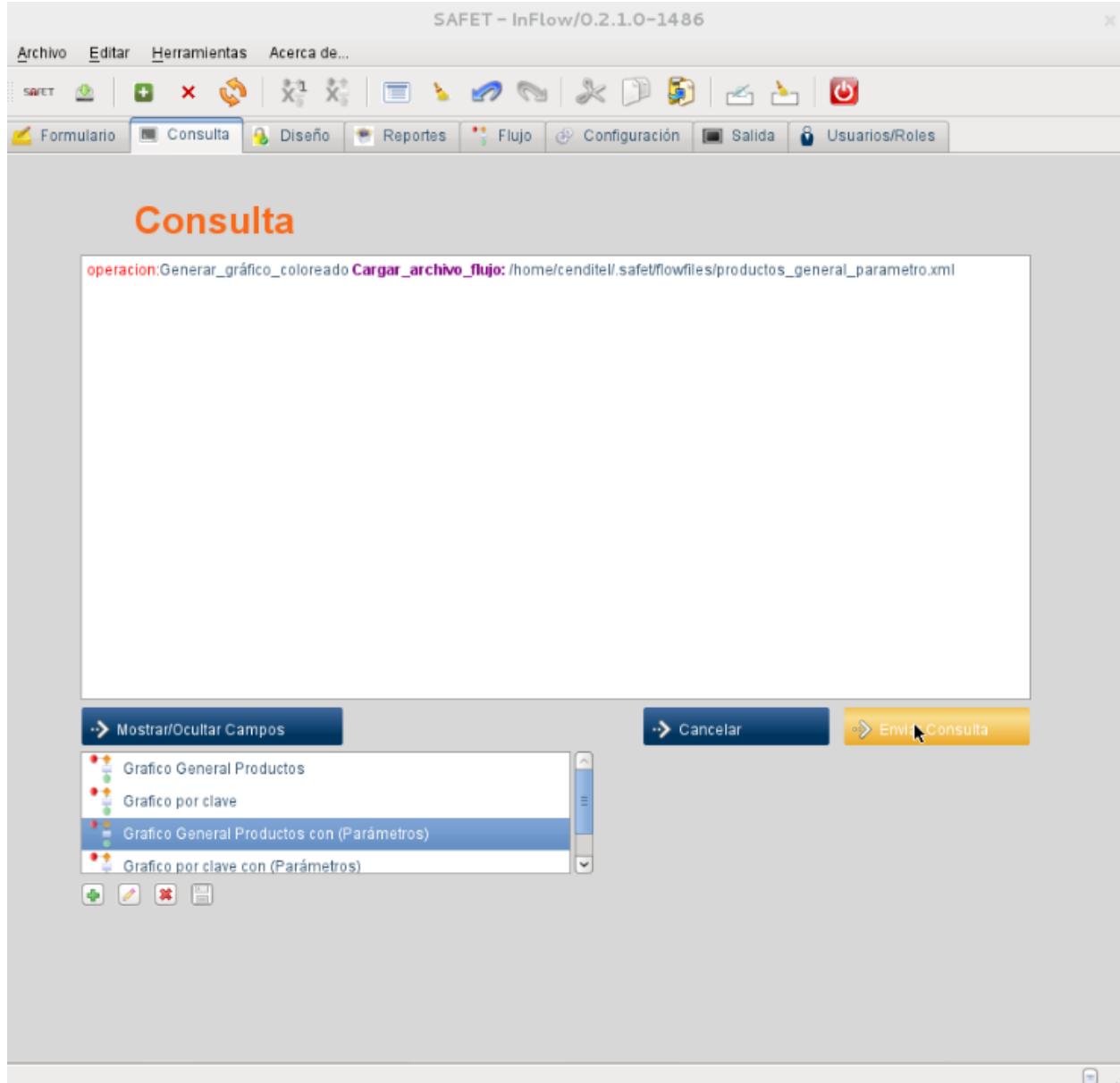


Figura 7.161: Figura 241: Botón (Enviar consulta)

## 7° SEPTIMO PASO

- Seleccionamos el la **Dirección de flujo de trabajo** y colocamos la palabra a buscar en la opción (**Empezar\_por**) y pulsamos el botón (**OK**), como se muestra en la siguiente *Figura 242: Parámetros*

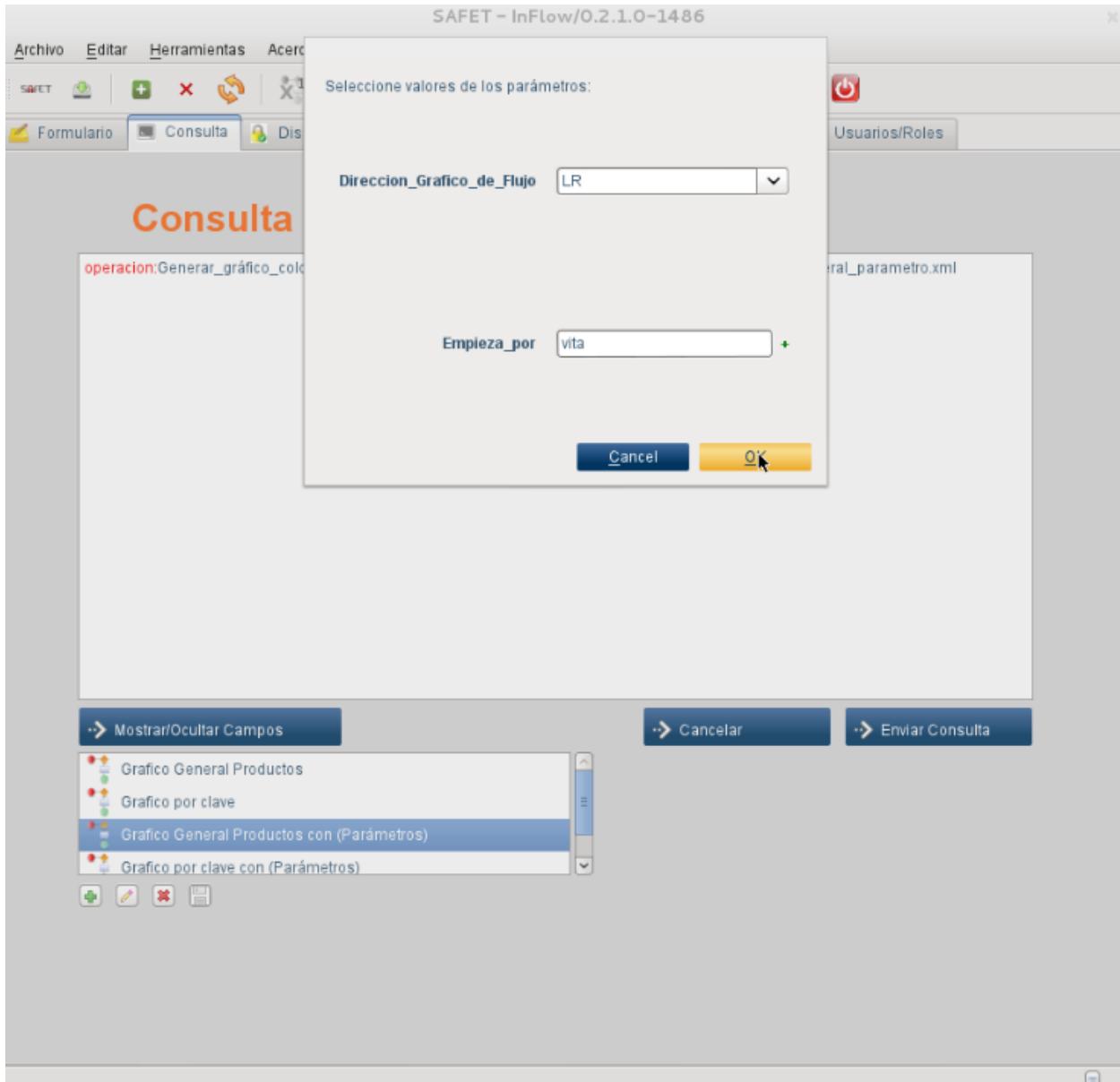


Figura 7.162: **Figura 242: Parámetros**

## 8° OCTAVO PASO

- Damos click al resultado (**Ver gráficos de flujo de trabajo**) para ver el gráfico, como se muestra en la siguiente *Figura 243: Resultado (Ver gráficos de flujo de trabajo)*

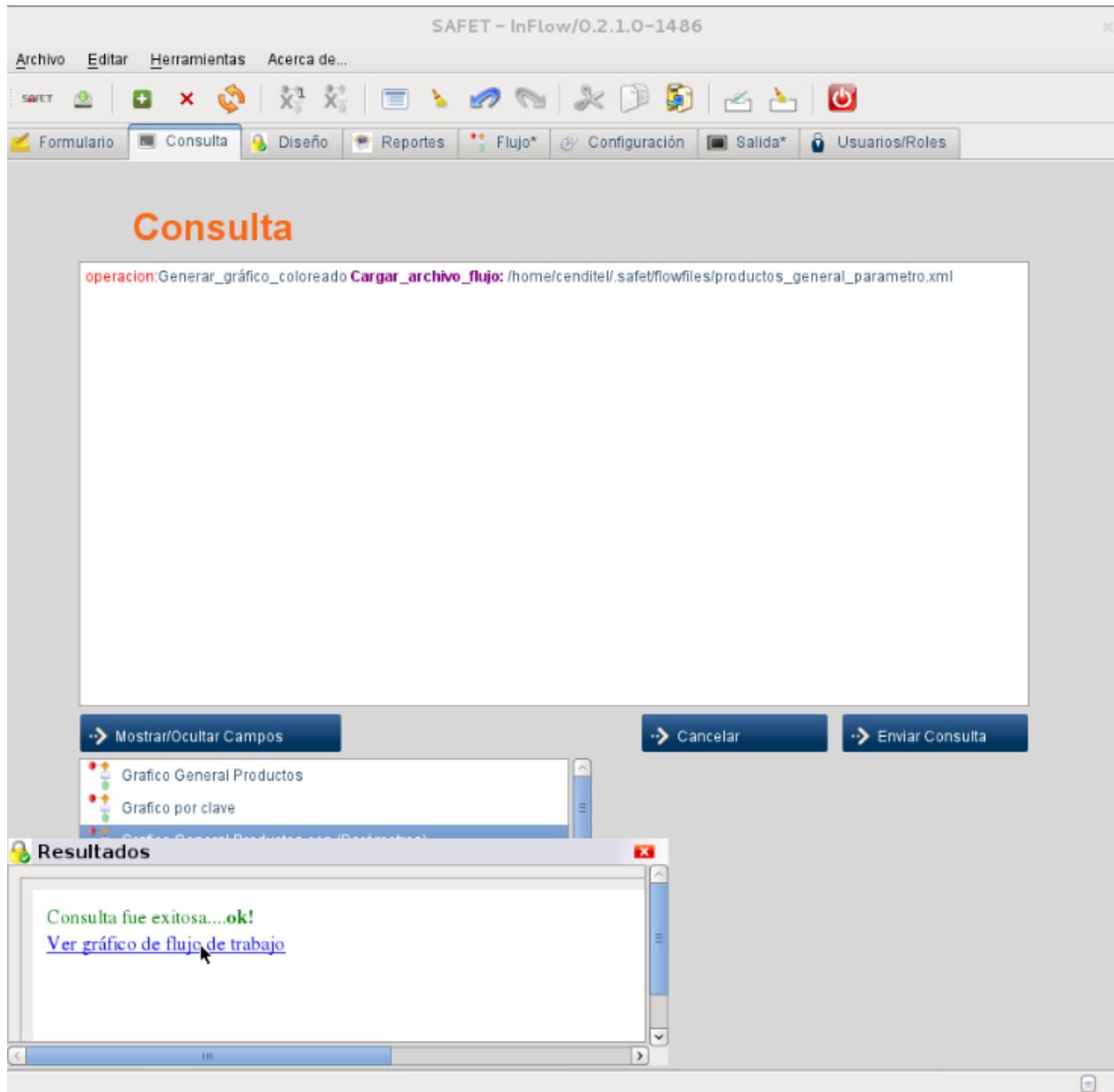


Figura 7.163: Figura 243: Resultado (Ver gráficos de flujo de trabajo)

**Nota:** Se nos mostrara la siguiente gráfica *Figura 244: Gráfica del un producto en específico*

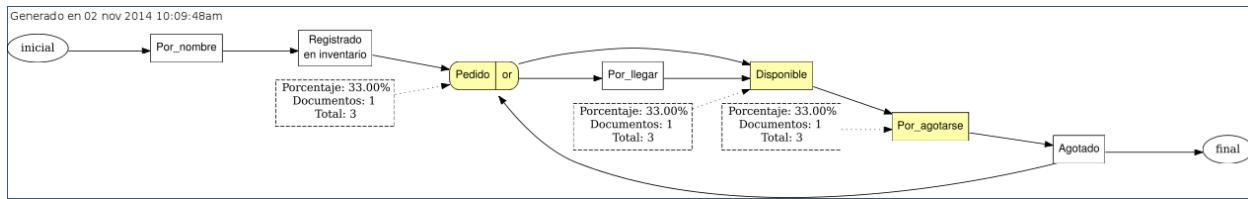


Figura 7.164: **Figura 244: Gráfica del un producto en específico**

#### 7.7.4 D.- SEGUNDA OPERACIÓN (Gráfico de un producto en específico con parámetro)

##### 1° PRIMER PASO

- Damos click a la operación (**Grafico por clave con (Parámetros)**), como se muestra en la siguiente *Figura 245: Operación (Grafico por clave con (Parámetros))*

##### 2° SEGUNDO PASO

- Damos click al botón (**modificar campo**), como se muestra en la siguiente *Figura 246: Botón (modificar campo)*

##### 3° TERCER PASO

- Seleccionamos el productos a buscar, por ejemplo ((5)-Vitamina B12), como se muestra en la siguiente *Figura 247: Selección del producto*

##### 4° CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 248: Botón (Fin de campo)*

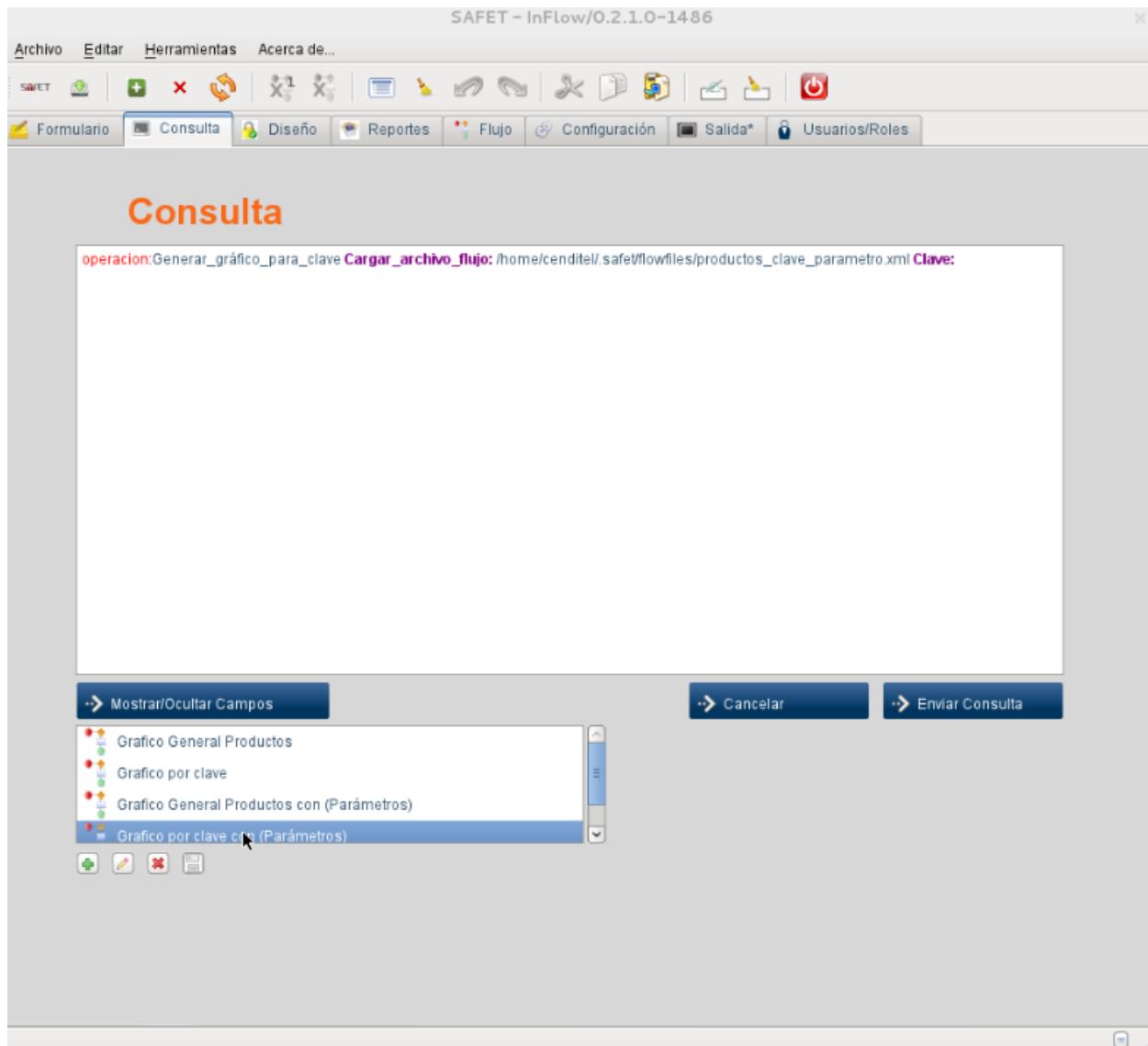


Figura 7.165: Figura 245: Operación (Grafico por clave con (Parámetros))

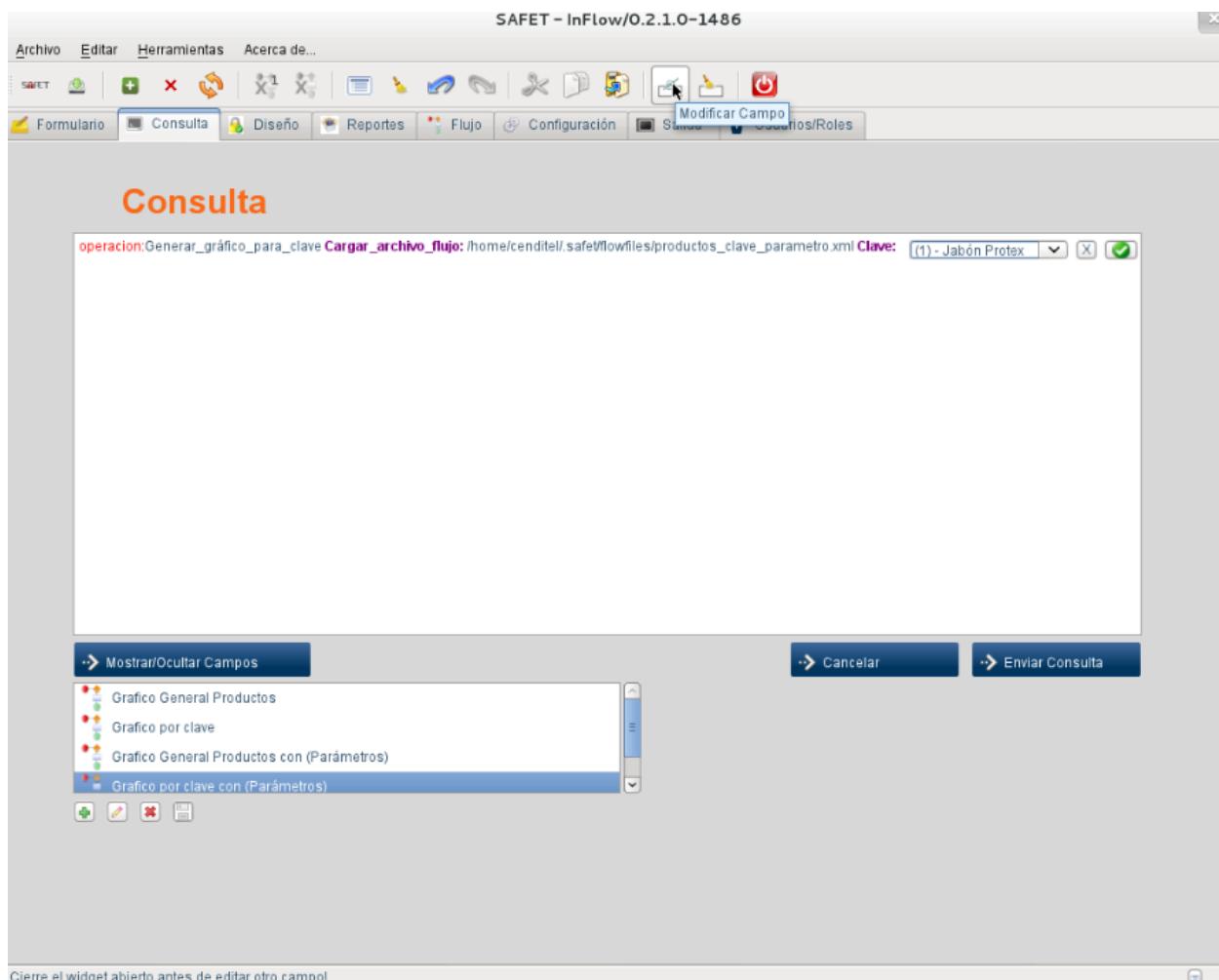


Figura 7.166: **Figura 246: Botón (modificar campo)**

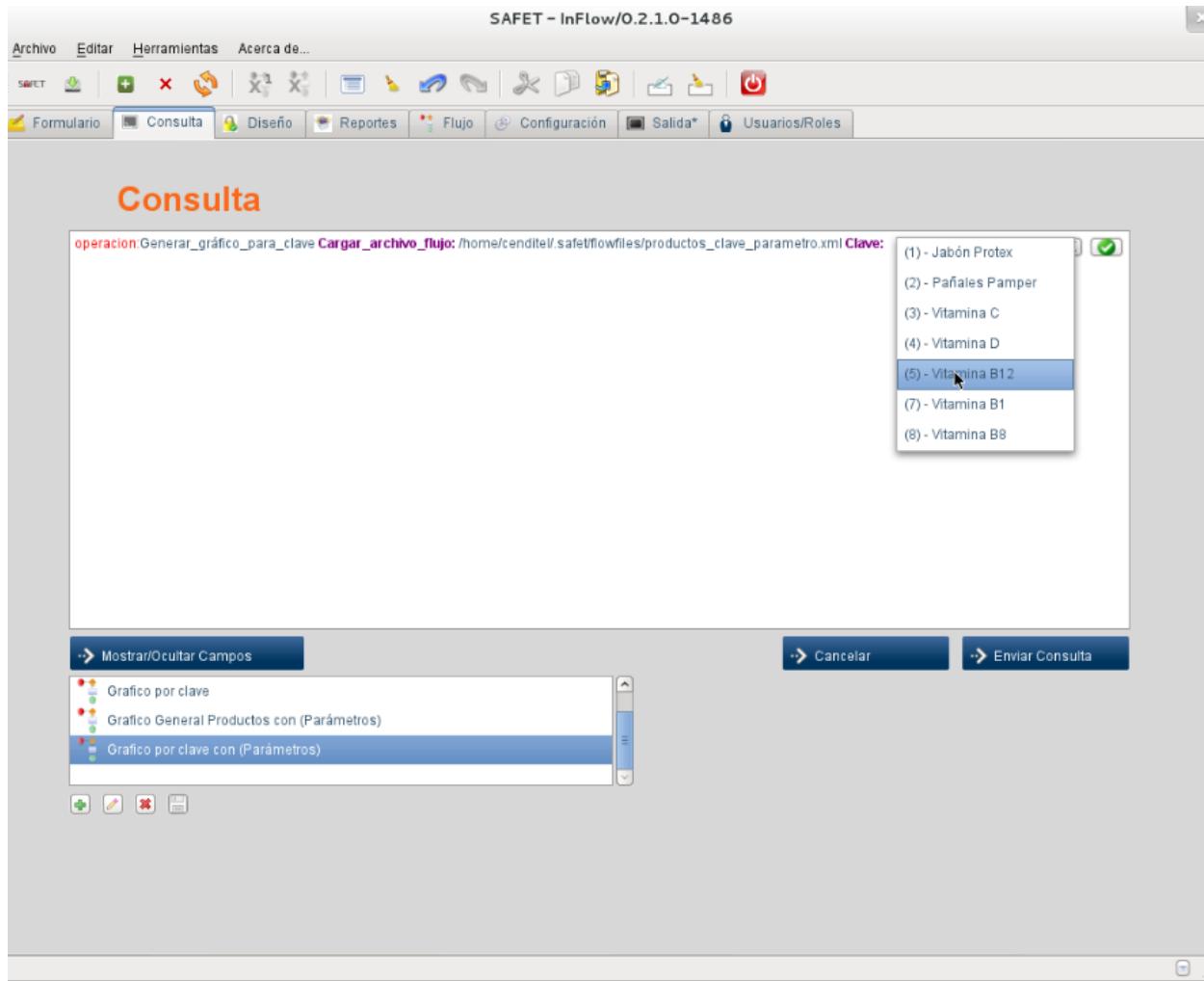


Figura 7.167: Figura 247: Selección del producto

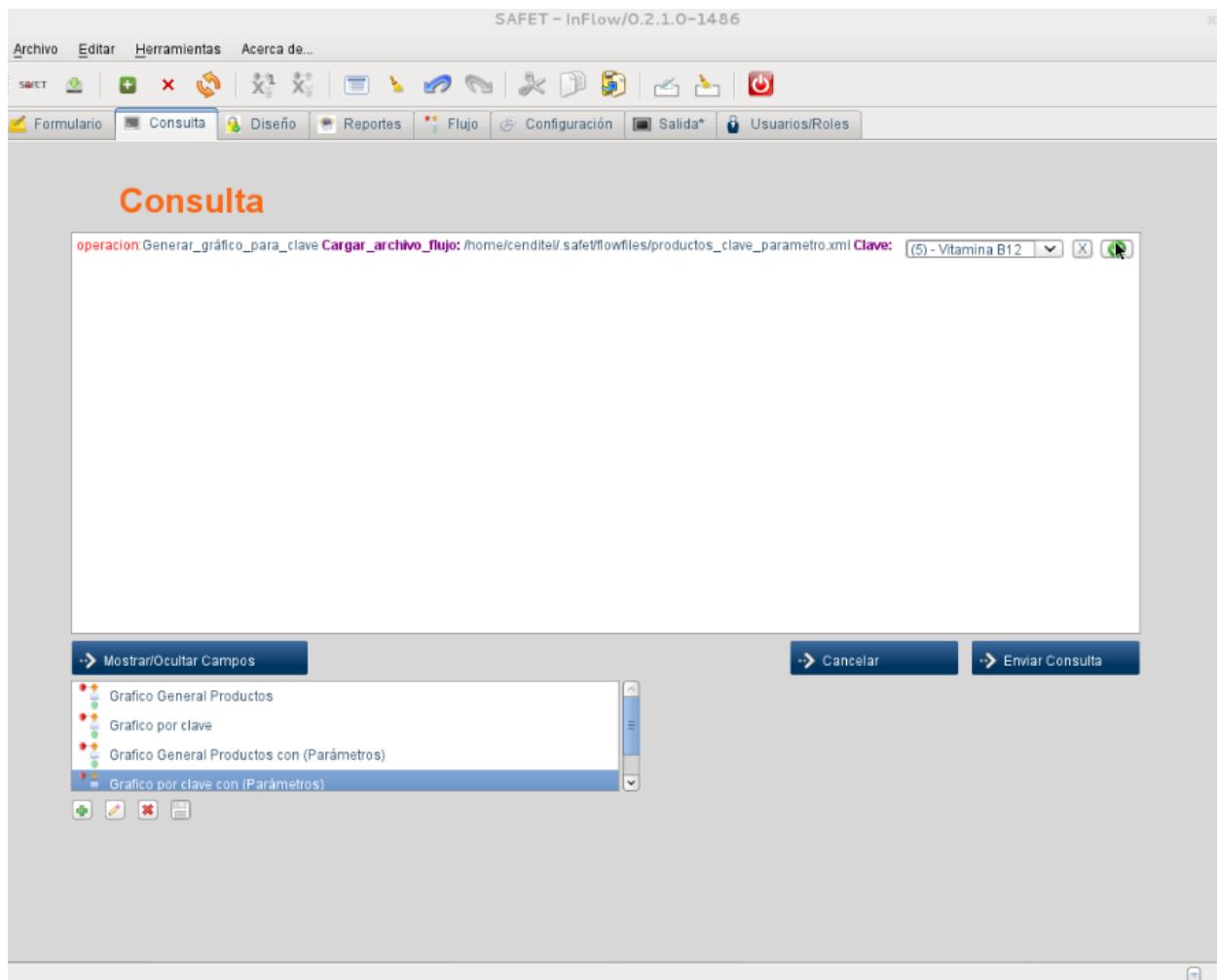


Figura 7.168: Figura 248: Botón (Fin de campo)

### 5° QUINTO PASO

- Pulsamos en el botón (**Enviar consulta**) para finalizar con la operación, como se muestra en la siguiente *Figura 249: Botón (Enviar consulta)*

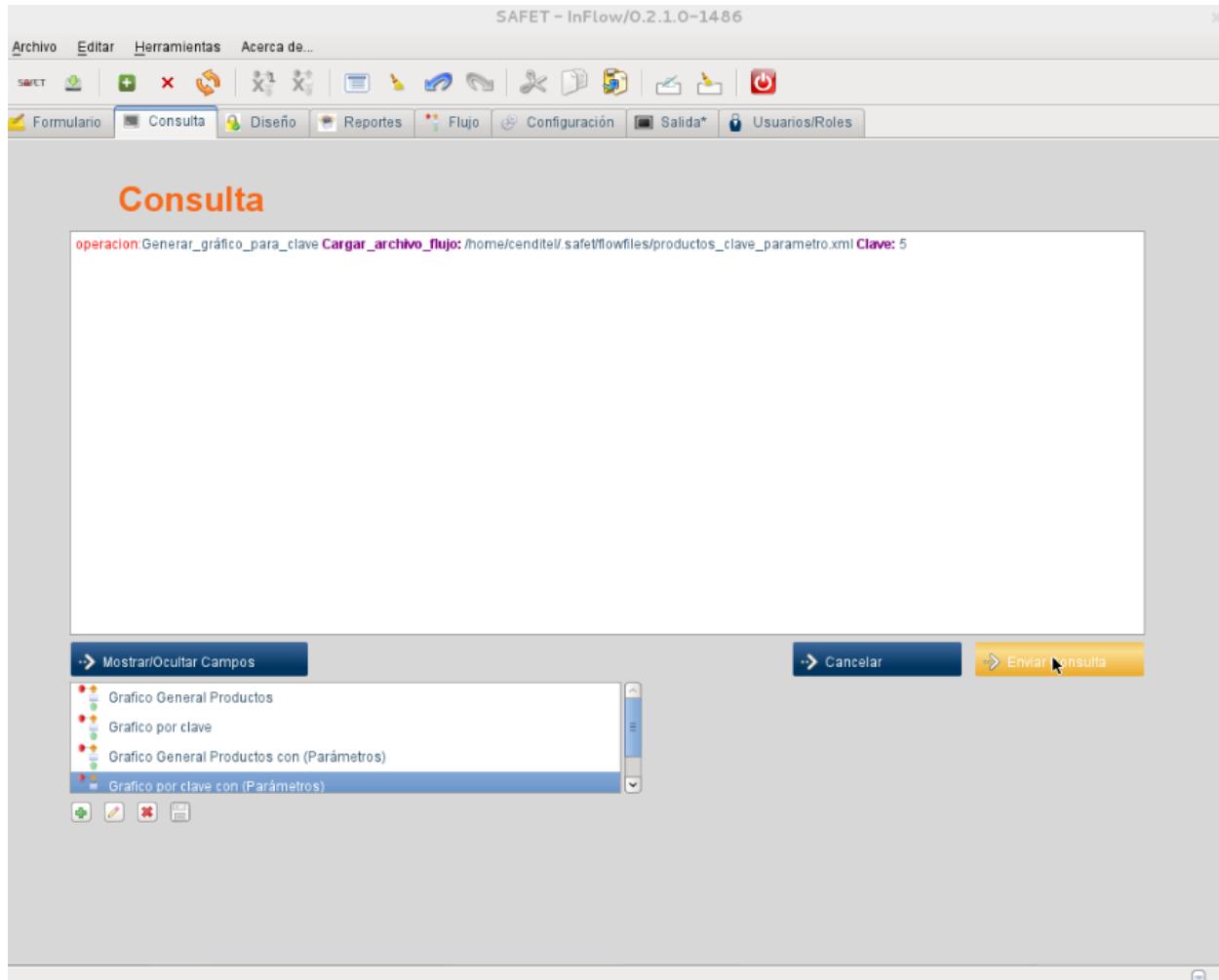


Figura 7.169: **Figura 249: Botón (Enviar consulta)**

### 6° SEXTO PASO

- Agregamos el parámetro de texto, por ejemplo (**Blue**) y pulsamos el botón (**Ok**), como se muestra en la siguiente *Figura 250: Parámetro de texto*

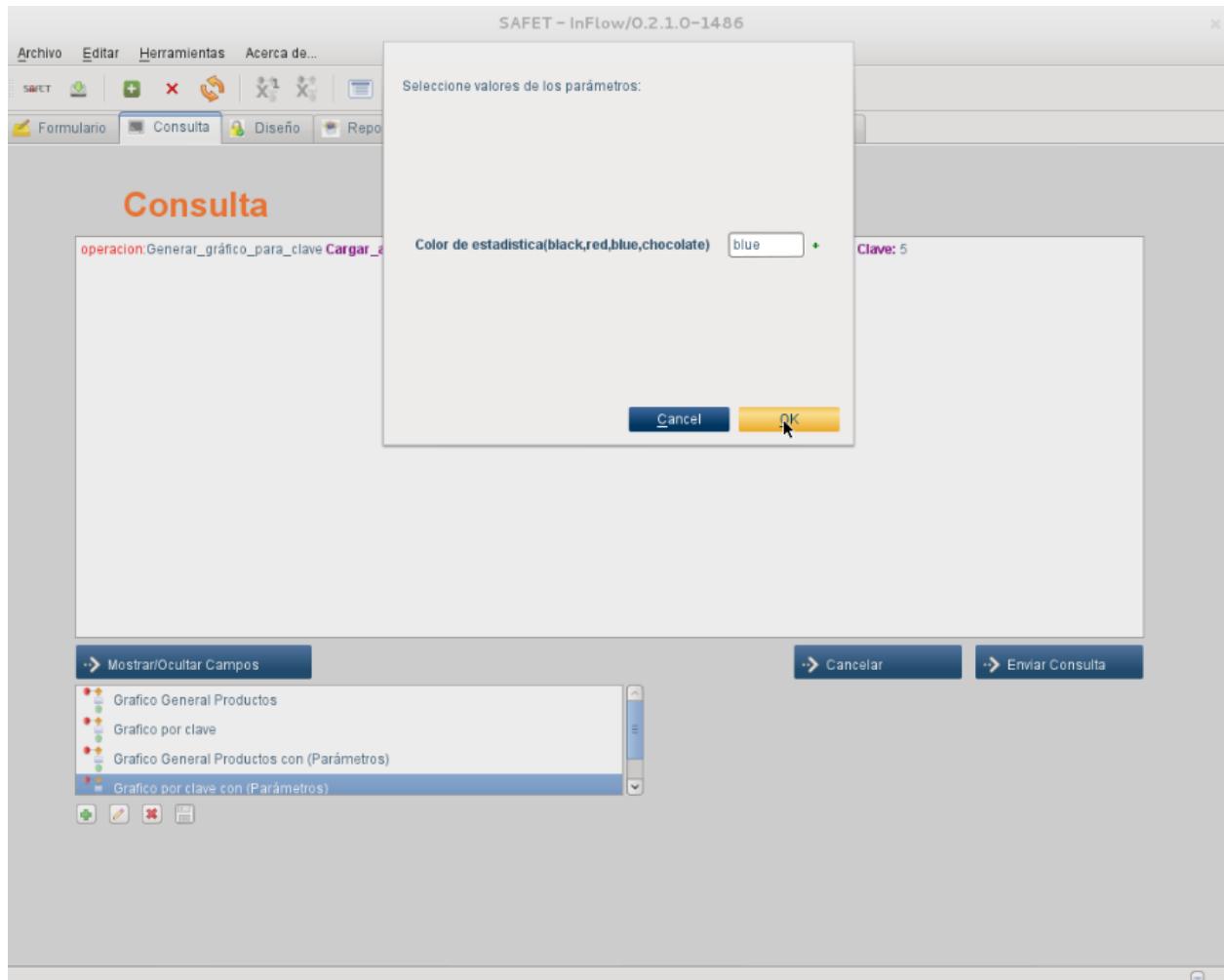


Figura 7.170: Figura 250: Parámetro de texto

### 7° SEPTIMO PASO

- Damos click al resultado (**Ver gráficos de flujo de trabajo**) para ver el gráfico, como se muestra en la siguiente *Figura 251: Resultado (Ver gráficos de flujo de trabajo)*

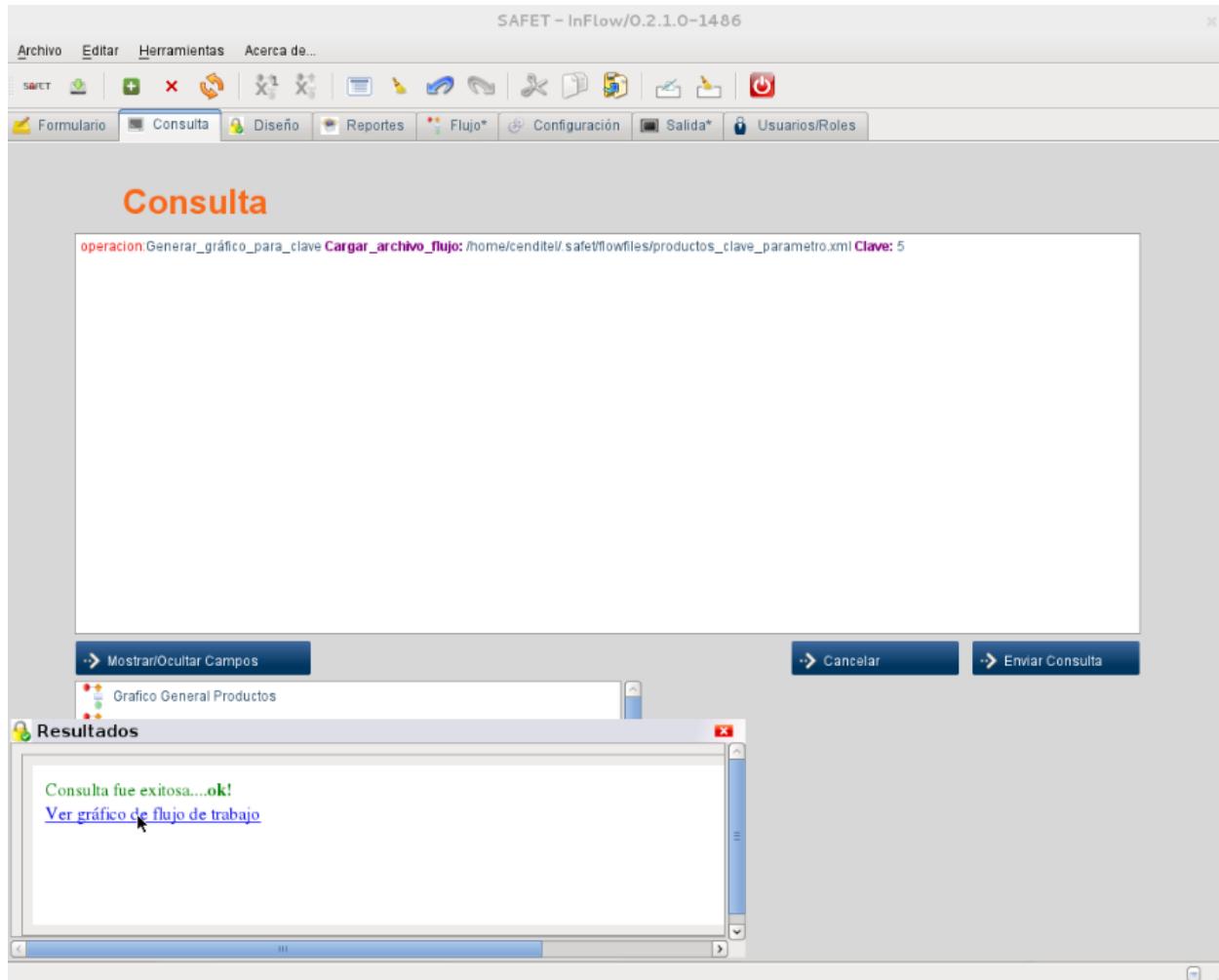


Figura 7.171: Figura 251: Resultado (Ver gráficos de flujo de trabajo)

---

**Nota:** Se nos mostrara la siguiente gráfica *Figura 252: Gráfica del un producto en específico*

- Se nos muestra las tareas en color **amarillo**, que significa por donde ah pasado el **producto**.

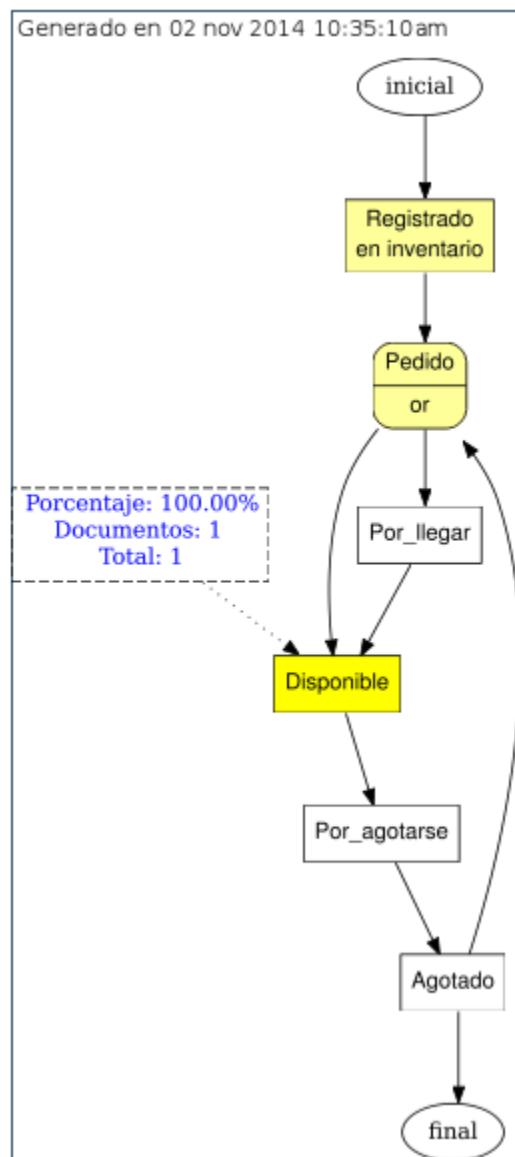


Figura 7.172: Figura 252: Gráfica del un producto en específico

## 7.8 Ejecución de gráficos usando flujos de trabajos con autofiltros

### 7.8.1 A).- Editar el archivo (`productos.xml`)

En el paso anterior, insertamos los autofiltros la cual vamos a utilizar en este tutorial. Si no has insertado los autofiltros damos click al siguiente enlace. [A\).- Editar el archivo \(`productos.xml`\)](#)

### 7.8.2 B).- Agregar campo en la tabla productos

En el paso anterior, ingresamos a la base de datos un campo (**categoria**) a la tabla **productos** la cual vamos a utilizar en este tutorial. Si no has ingresado el campo en la base de datos damos click al siguiente enlace. [B\).- Agregar campo en la tabla `productos`](#)

### 7.8.3 C).- Autofiltros con la interfaz gráfica inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente [Figura 253: Autenticación de Inflow](#).

#### 3° TERCER PASO

- Damos click a la segunda opción (**Realizar consultas**), como se muestra en la siguiente [Figura 254: Realizar consultas](#)

#### 4° CUARTO PASO

- Damos click a la opción (**Mostrar/Ocultar Campos**), la cual se nos mostrara las acciones la **Opciones de listados**, como se muestra en la siguiente [Figura 255: Acciones de reportes](#)



Figura 7.173: Figura 253: Autenticación de Inflow.

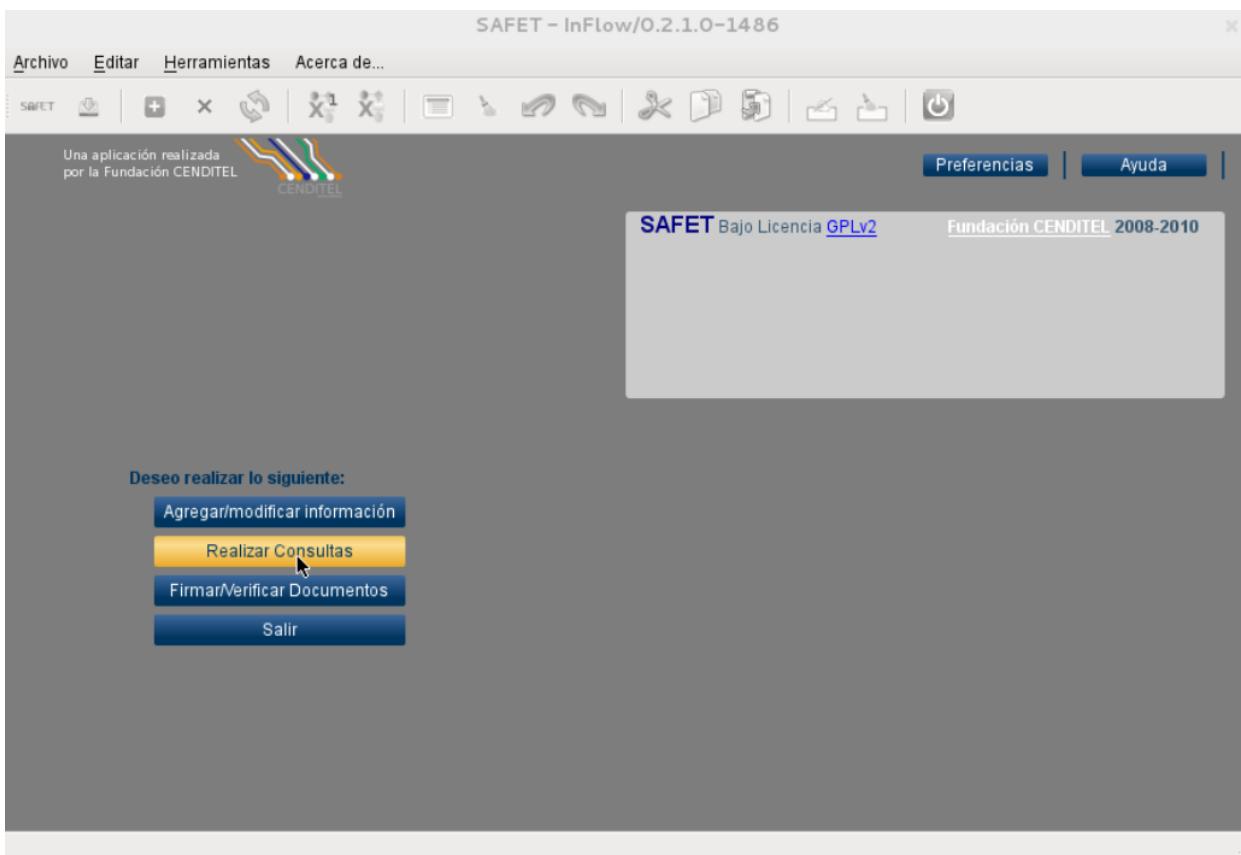


Figura 7.174: Figura 254: Realizar consultas

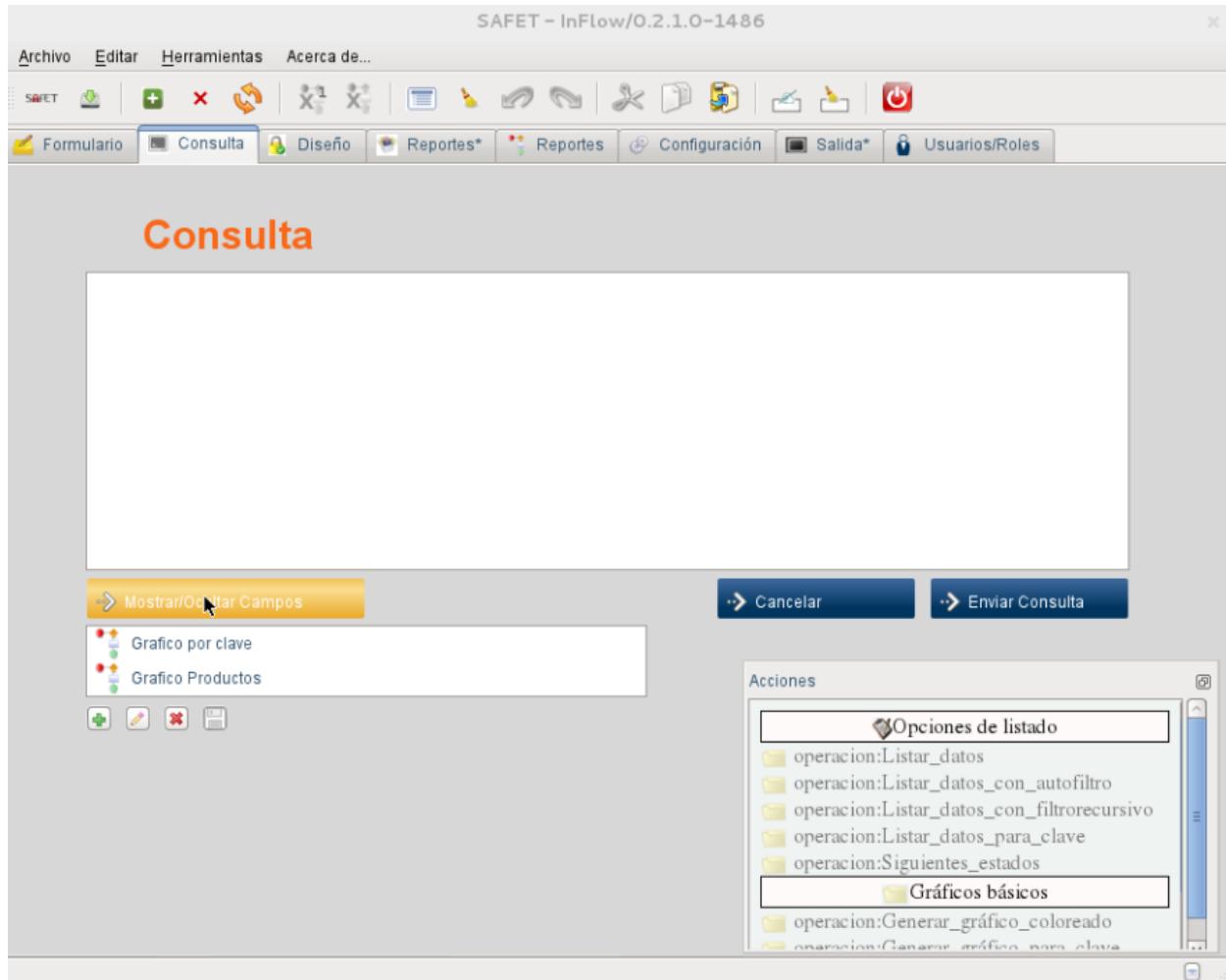


Figura 7.175: Figura 255: Acciones de reportes

## 5° QUINTO PASO

- Damos un click a la opción (**operacion:Generar\_gráfico\_con\_autofiltro**), como se muestra en la siguiente *Figura 256: operacion:Generar\_gráfico\_con\_autofiltro*

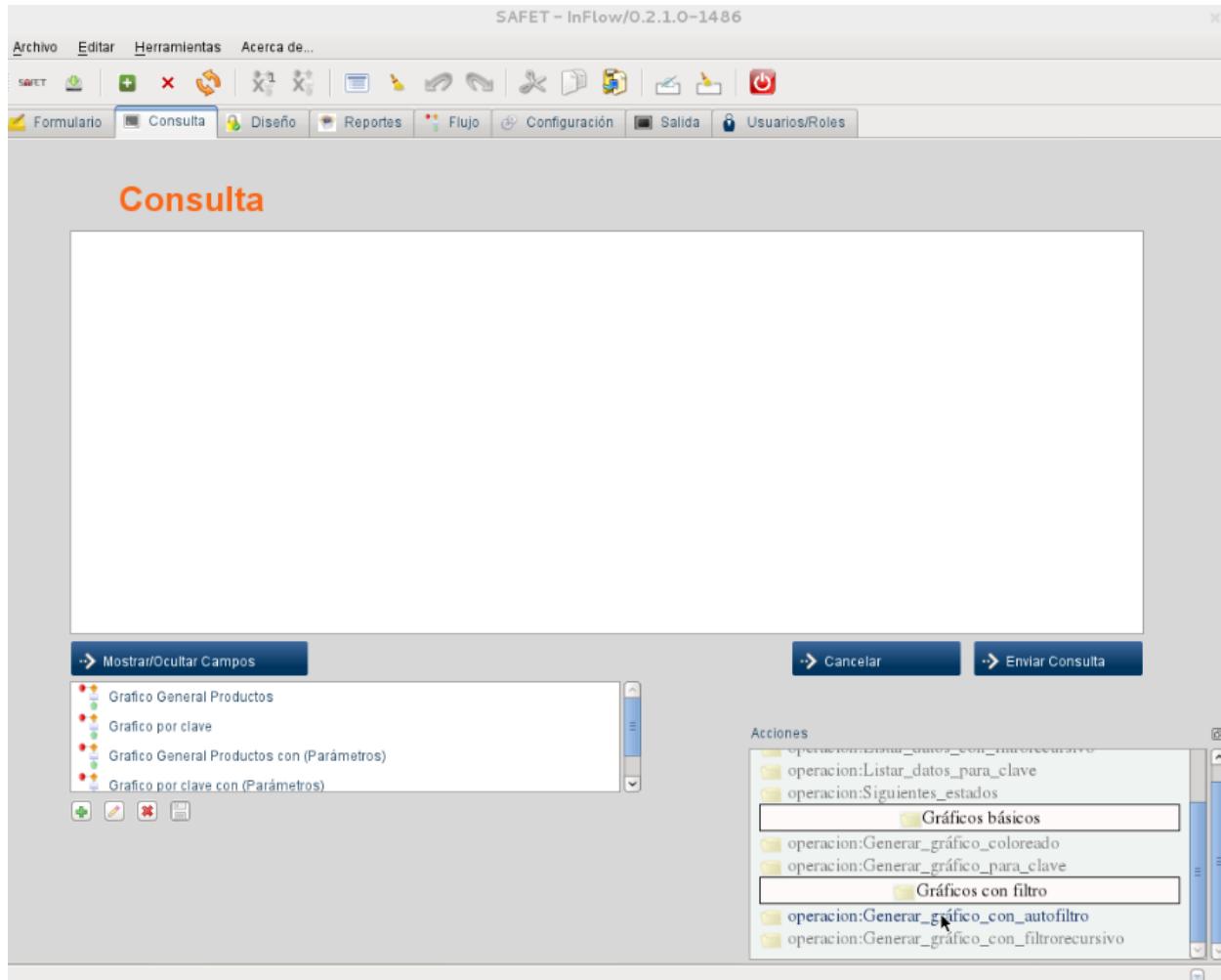


Figura 7.176: Figura 256: **operacion:Generar\_gráfico\_con\_autofiltro**

## 6° SEXTO PASO

- Damos un click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 257: Campo (Cargar\_archivo\_flujo\*)*

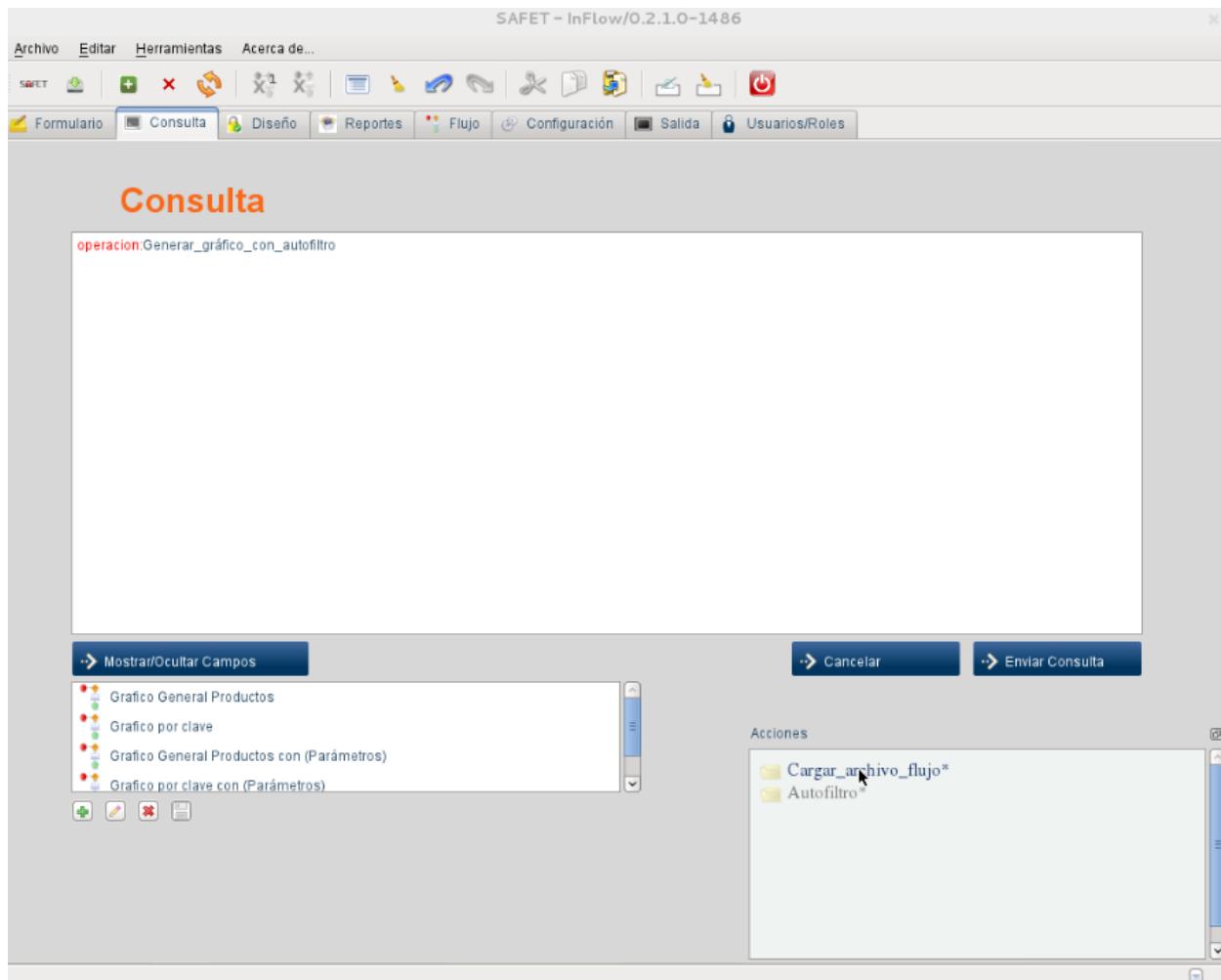


Figura 7.177: **Figura 257: Campo (Cargar\_archivo\_flujo\*)**

## 7° SEPTIMO PASO

- Seleccionamos el archivo **productos.xml**, como se muestra en la siguiente *Figura 258: Selección de Archivo (XML)*

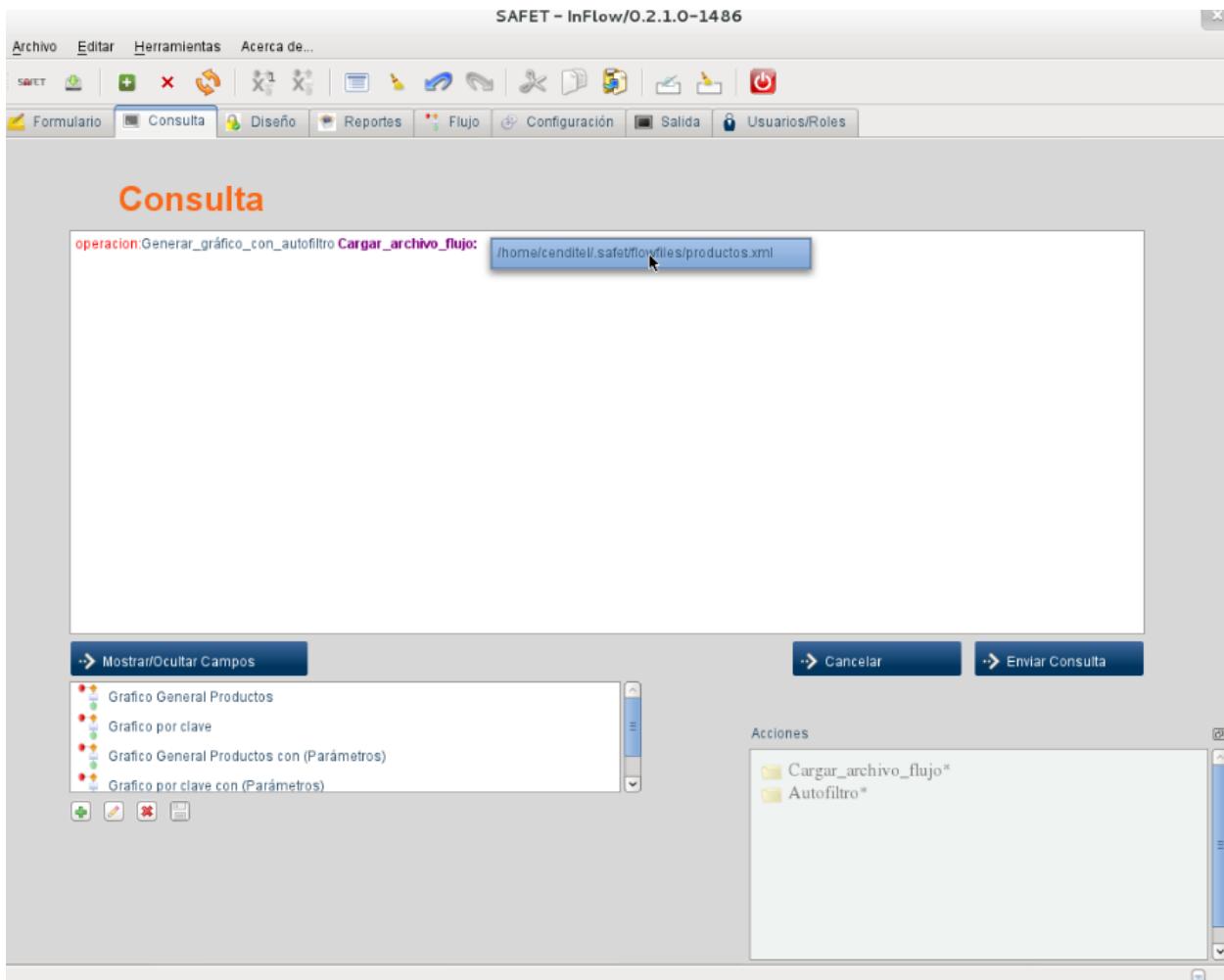


Figura 7.178: **Figura 258: Selección de Archivo (XML)**

## 8° OCTAVO PASO

- Damos un click al botón con la flecha verde, significando que ya está lleno el campo, como se muestra en la siguiente *Figura 259: Resultado de la operación*

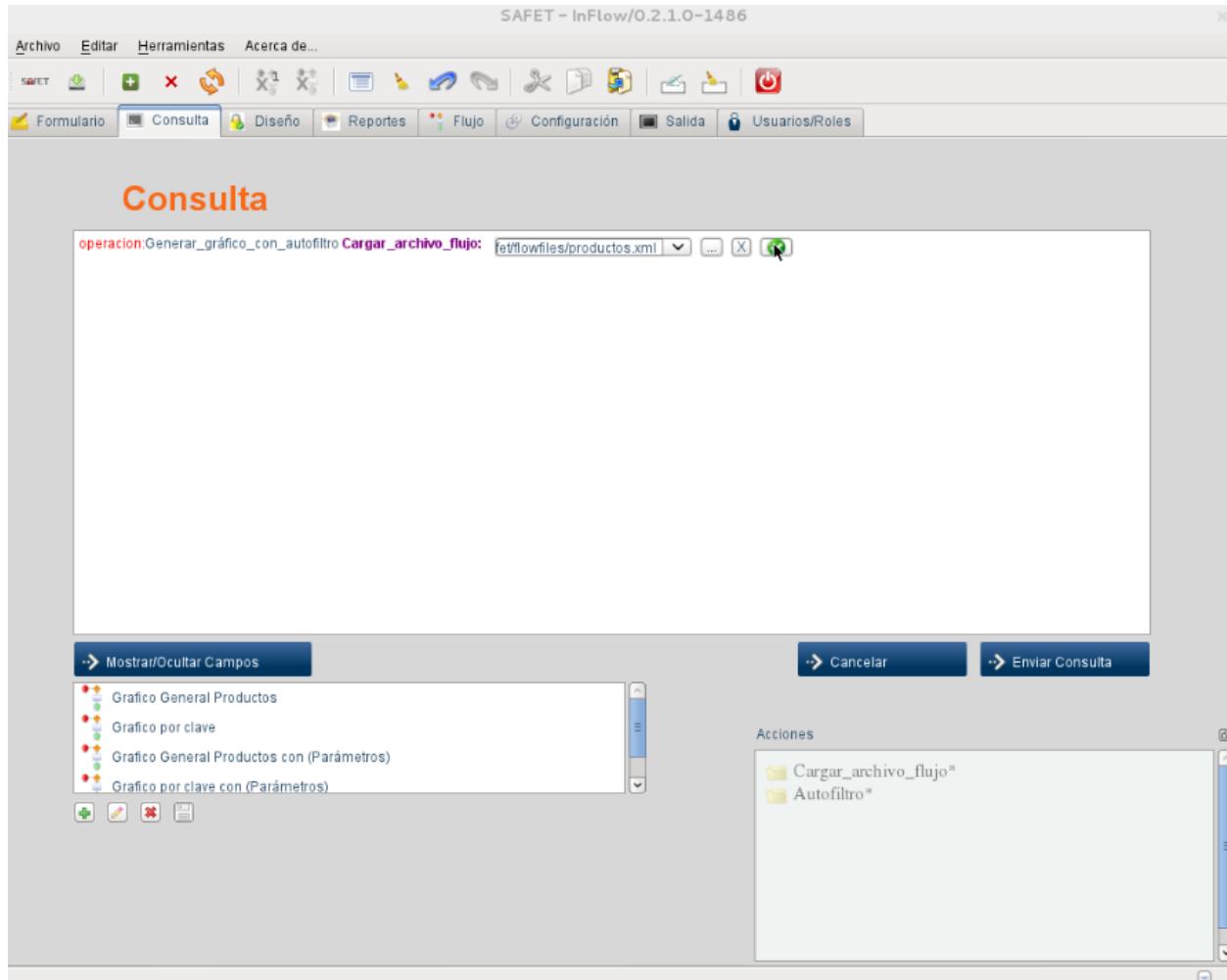


Figura 7.179: **Figura 259:** Resultado de la operación

## 9° NOVENO PASO

- Damos un click al campo obligatorio (**Autofiltro\***), como se muestra en la siguiente *Figura 260: Campo (Autofiltro\*)*

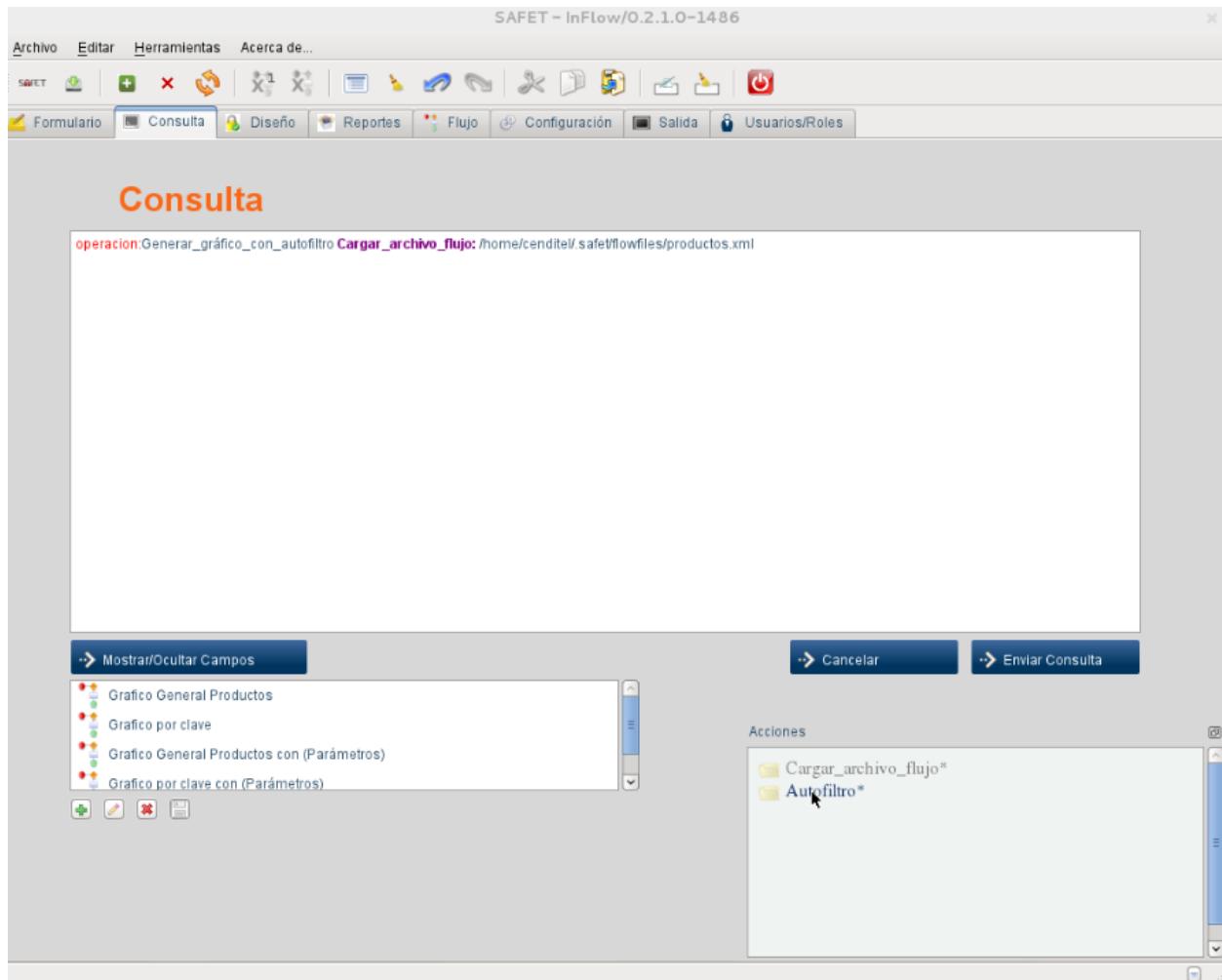


Figura 7.180: **Figura 260: Campo (Autofiltro\*)**

## 10° DECIMO PASO

- Seleccionamos la **categoria** a buscar, por ejemplo (**Categoría\_registrado**) como se muestra en la siguiente *Figura 261: Selección de (Categoría)*

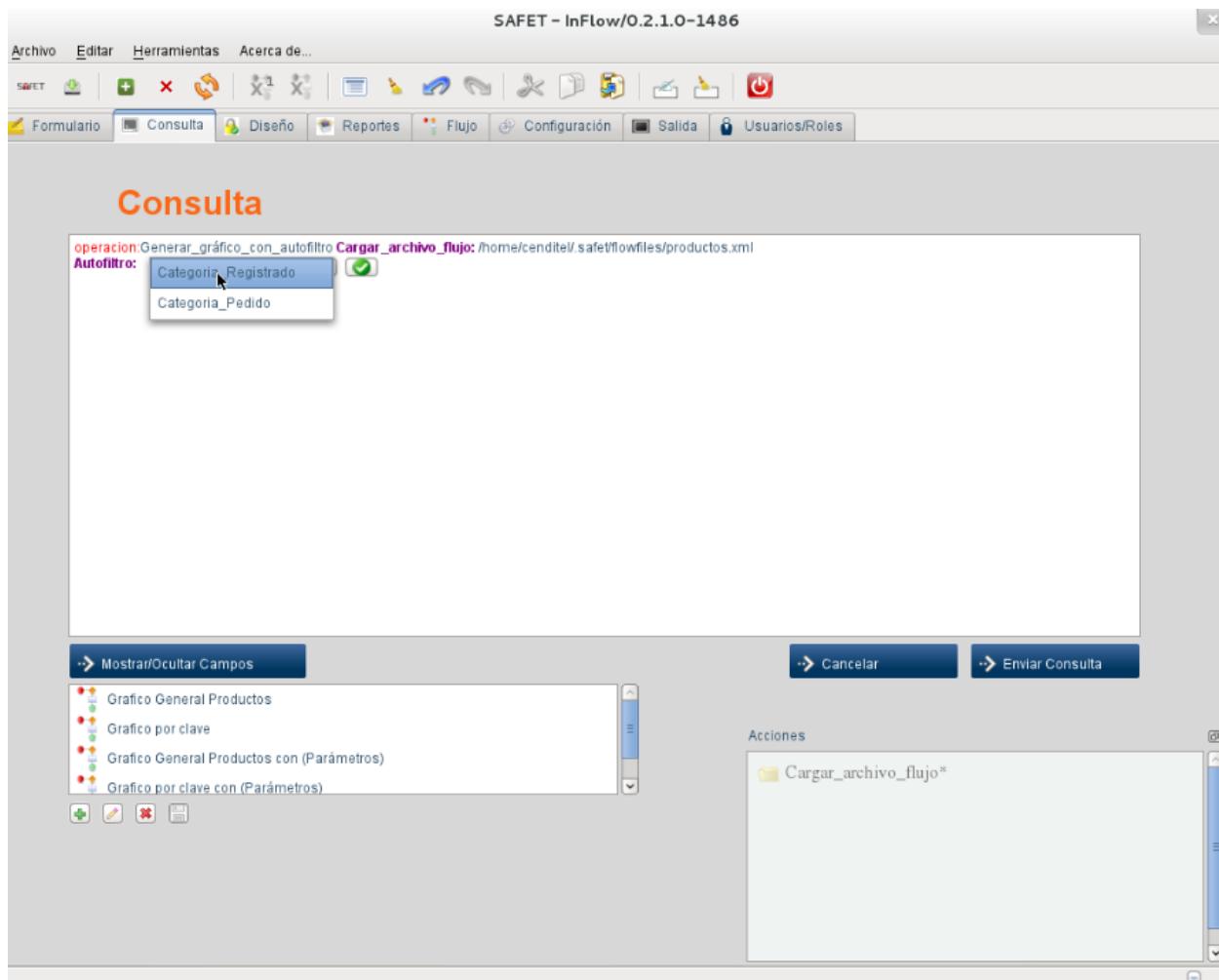


Figura 7.181: Figura 261: Selección de (Categoría)

## 11° DECIMO PRIMERO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 262: Botón (Fin de campo)*

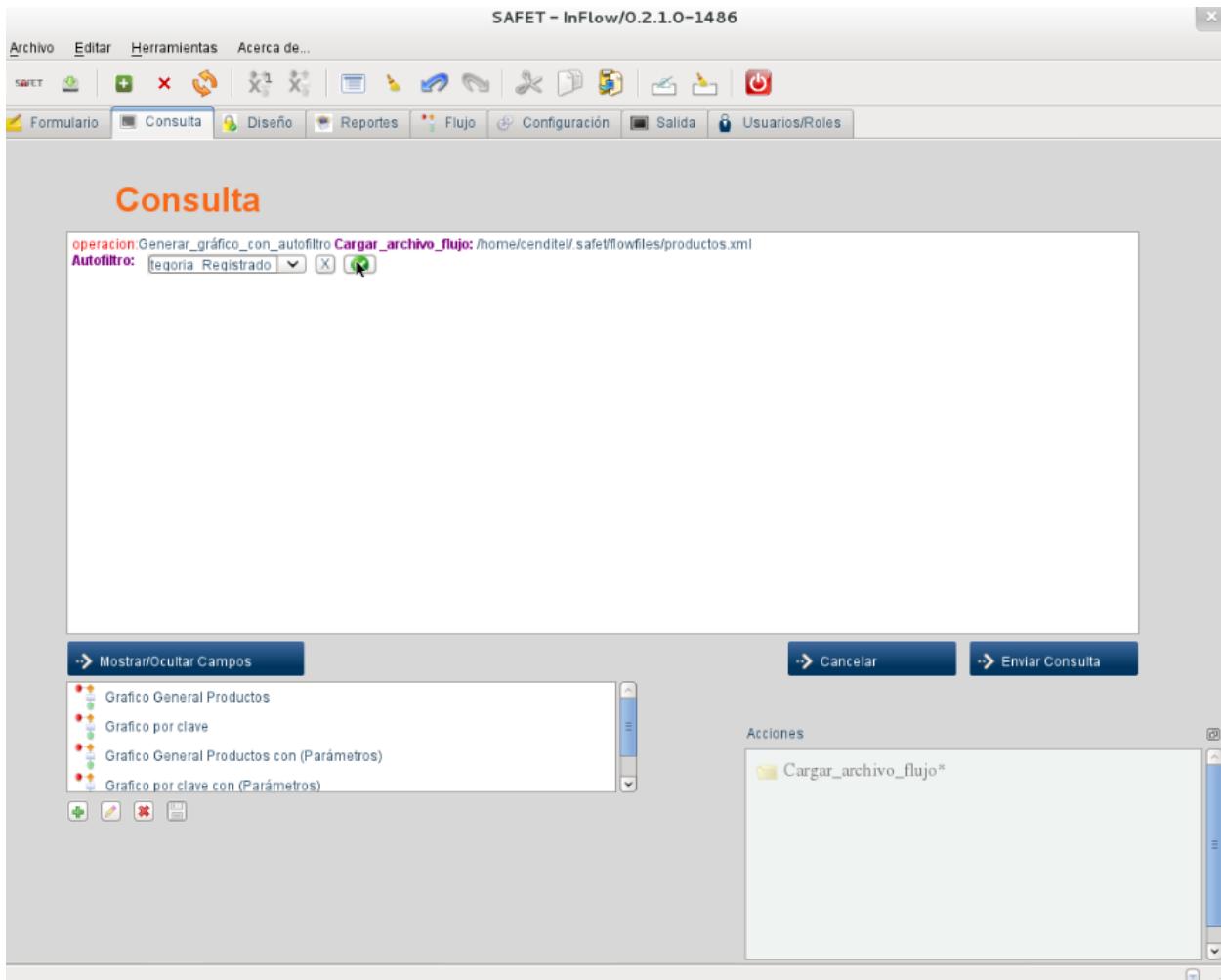


Figura 7.182: **Figura 262: Botón (Fin de campo)**

## 12° DECIMO SEGUNDO PASO

- Para terminar con la operación damos un click al botón (**Enviar formulario**) y nos mostrara como resultado (**Consulta fue exitosa....ok! (Ver reporte)** ), como se muestra en la siguiente *Figura 263: Botón (Fin de campo)*

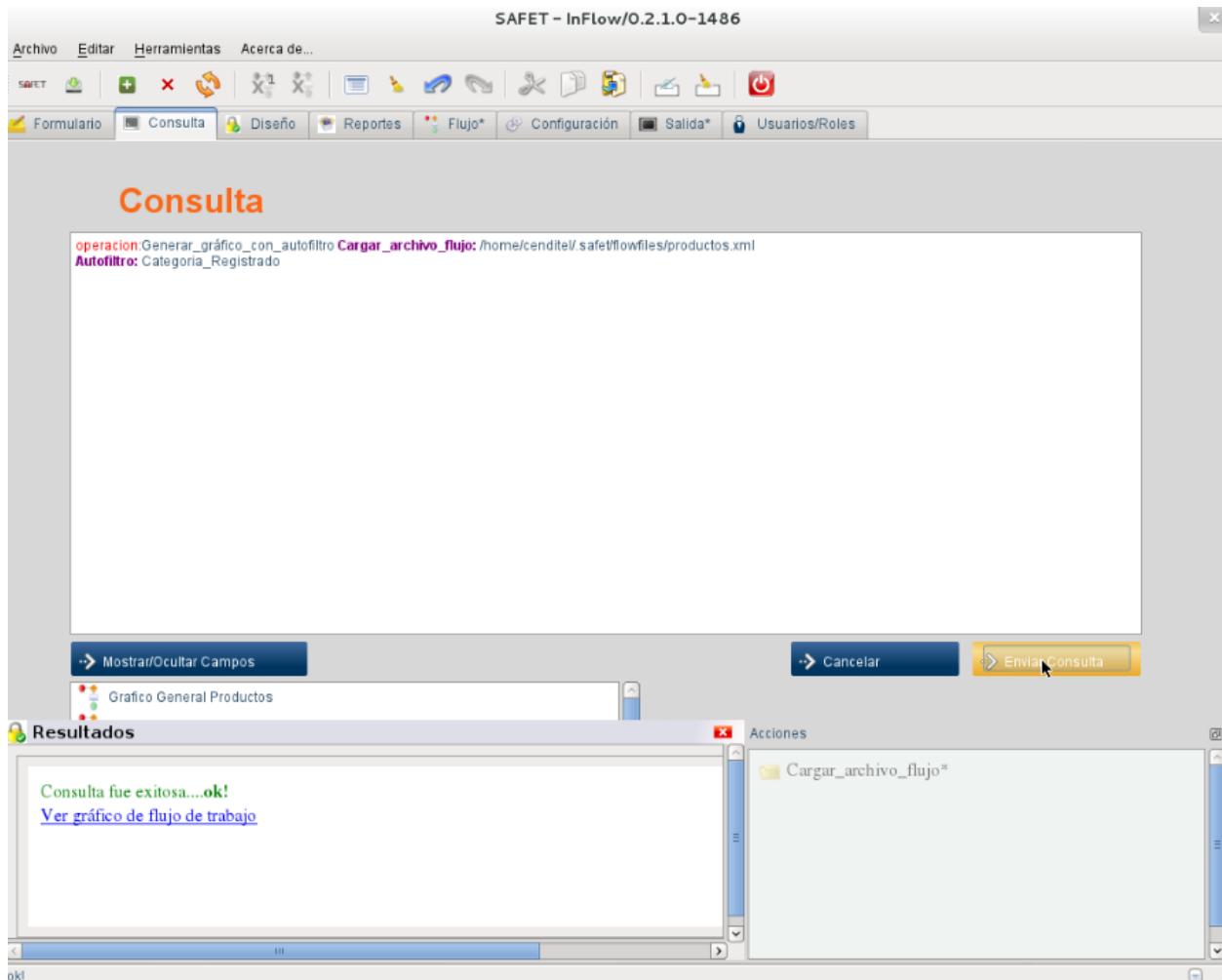


Figura 7.183: Figura 263: Botón (Fin de campo)

### 13° DECIMO TERCERO PASO

- Damos click al resultado (**Ver gráficos de flujo de trabajo**) para ver el gráfico, como se muestra en la siguiente *Figura 264: Resultado (Ver gráficos de flujo de trabajo)*

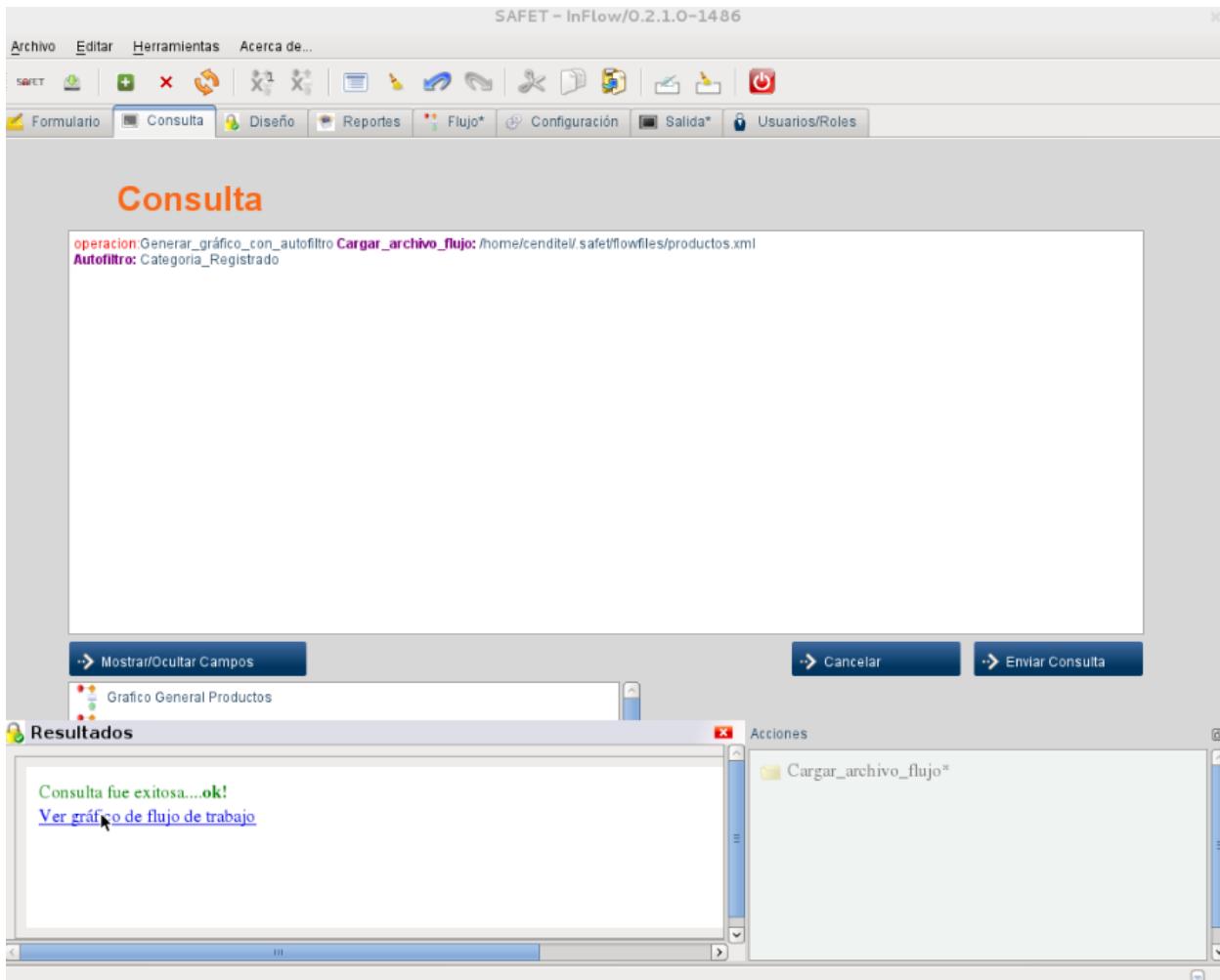


Figura 7.184: Figura 264: Resultado (Ver gráficos de flujo de trabajo)

---

**Nota:** Se nos mostrara la siguiente gráfica *Figura 265: Gráfico con autofiltro*

---

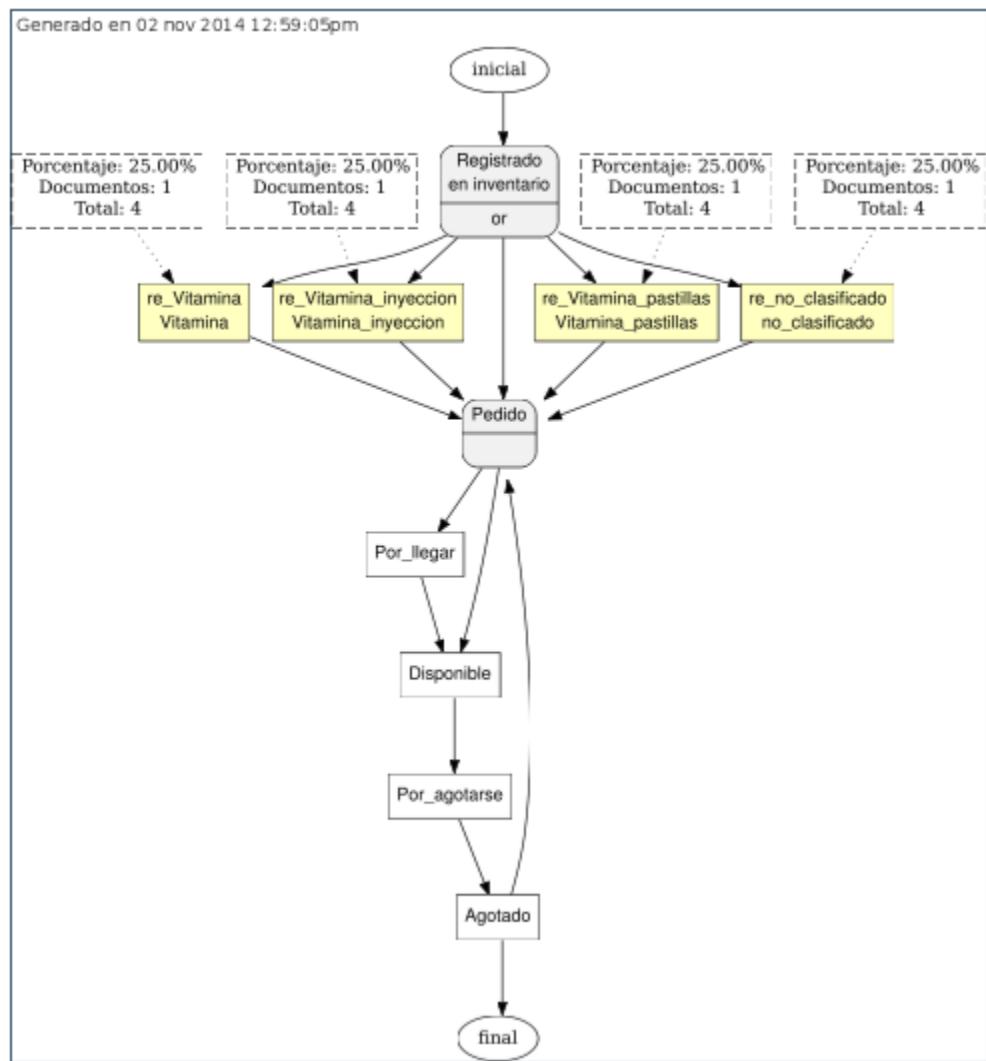


Figura 7.185: Figura 265: Gráfico con autofiltro

## 7.9 Diseño de creación de un nuevo gráfico de flujo de trabajo (.xml)

### 7.9.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 266: Autenticación de Inflow*.



Figura 7.186: **Figura 266: Autenticación de Inflow.**

#### 3° TERCER PASO

- Damos click en la segunda opción (**Firmar/verificar documento**), como se muestra en la siguiente *Figura 267: Opcion Diseño*

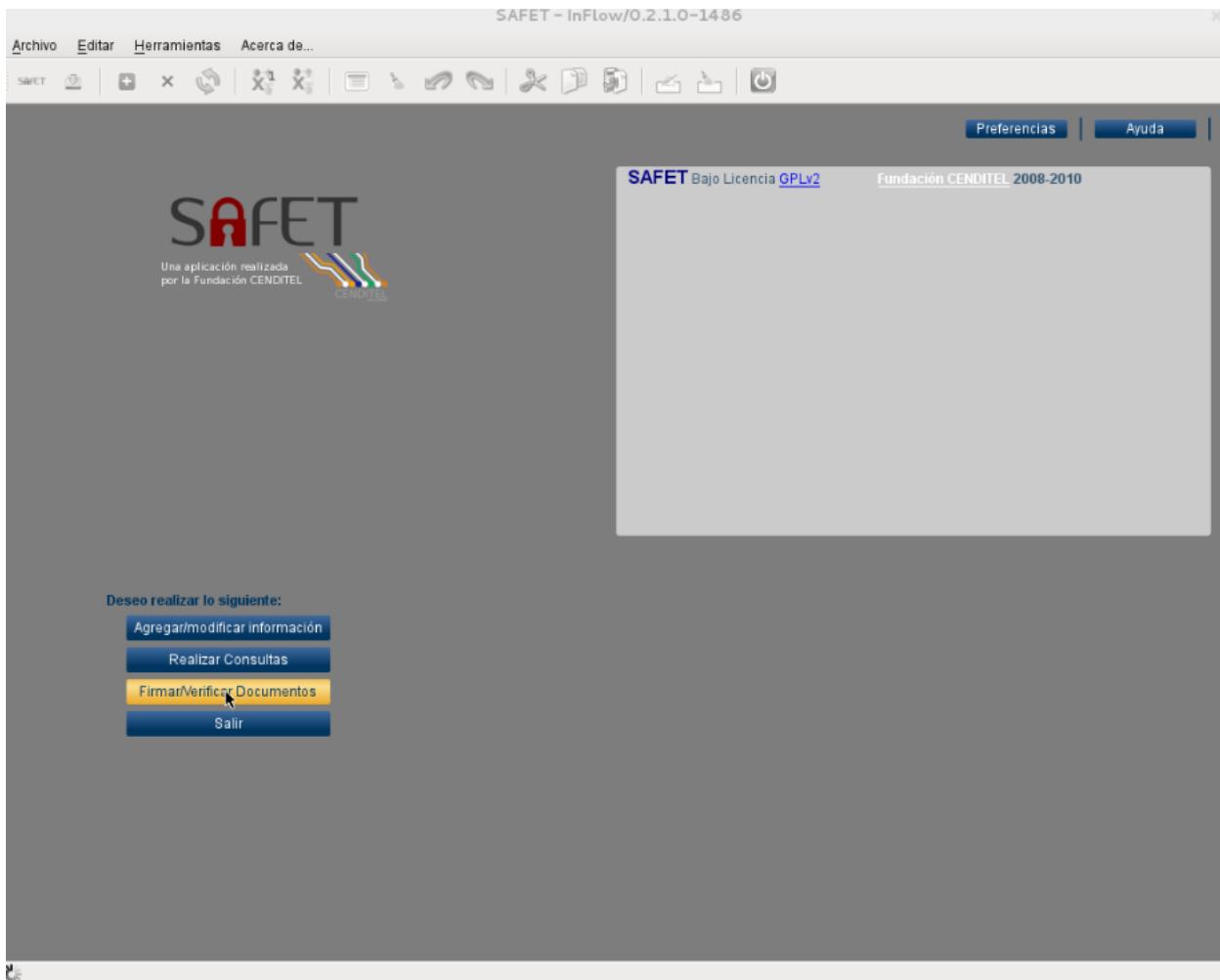


Figura 7.187: **Figura 267: Opcion Diseño**

#### 4° CUARTO PASO

- Damos click a la segunda opción (**Mostar/Ocultar Campos**), como se muestra en la siguiente *Figura 268: Campos del Diseño*

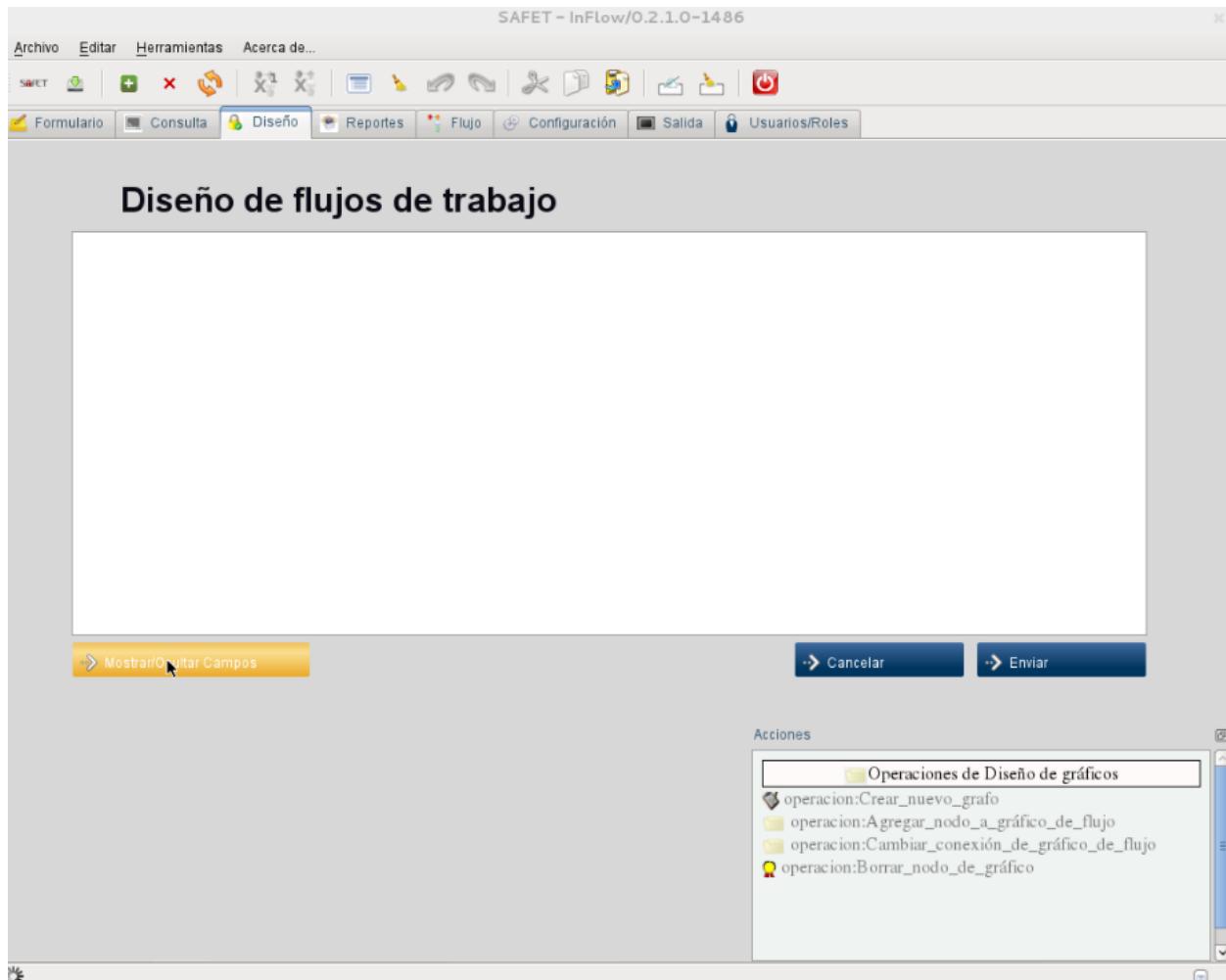


Figura 7.188: **Figura 268: Campos del Diseño**

#### 7.9.2 B.- Creación del flujo de trabajo (.xml)

##### 1° PRIMER PASO

- Damos click a la operación (**Crear\_nuevo\_grafo**), como se muestra en la siguiente *Figura 269: Crear\_nuevo\_grafo*

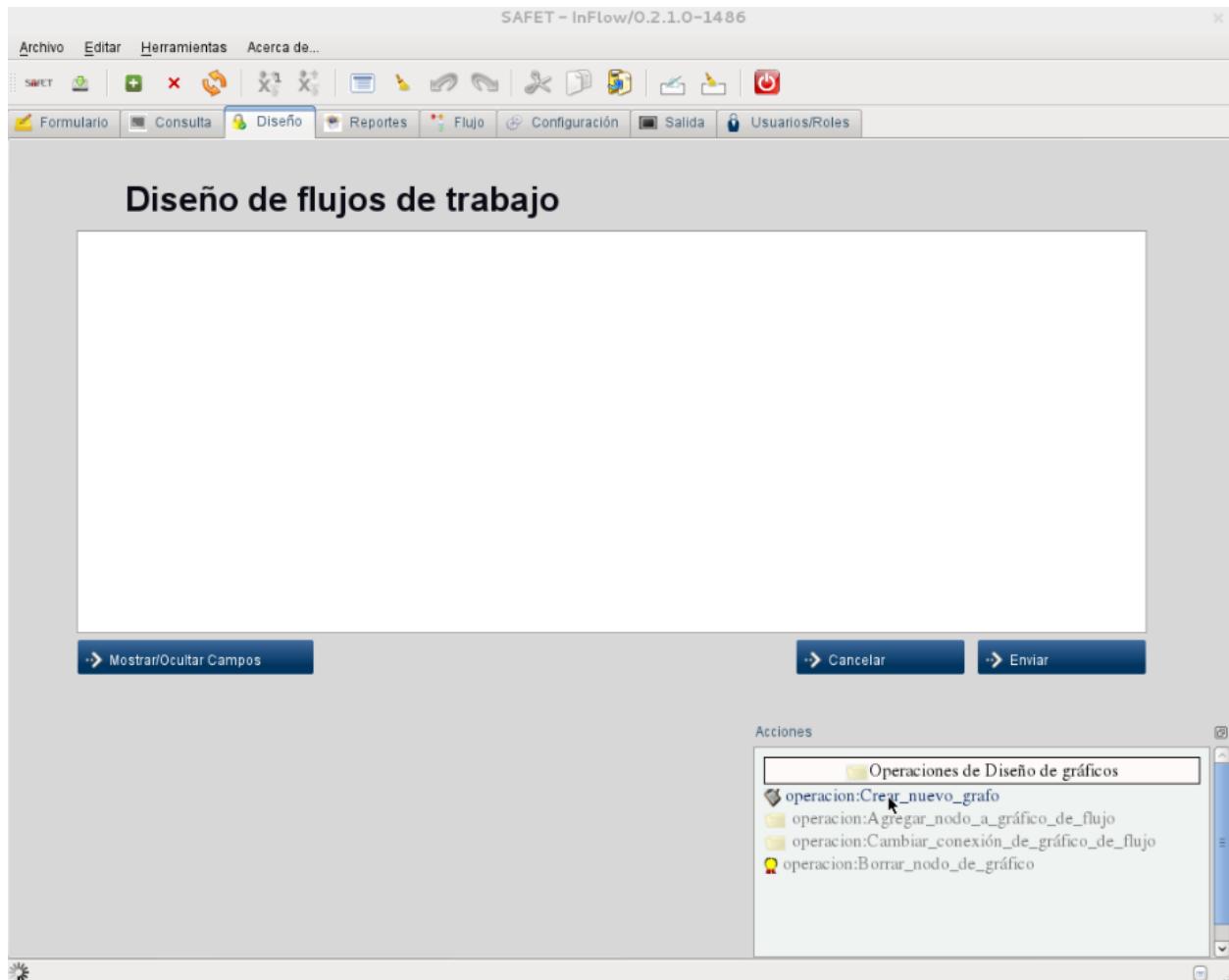


Figura 7.189: Figura 269: Crear\_nuevo\_grafo

## 2° SEGUNDO PASO

- Damos click al campo obligatorio (**Nombre\_del\_grafo\***), como se muestra en la siguiente *Figura 270: Campo (Nombre\_del\_grafo\*)*

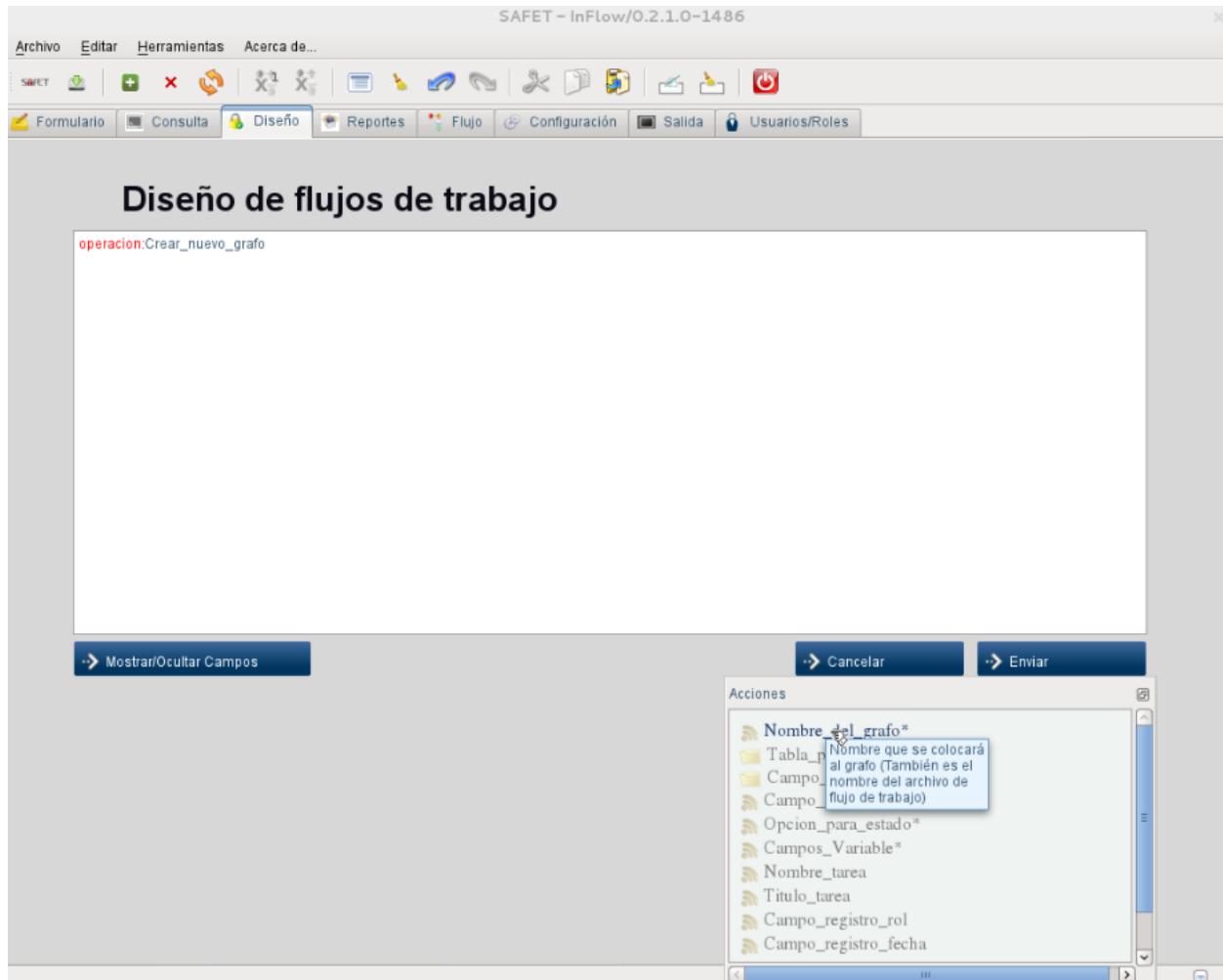


Figura 7.190: **Figura 270: Campo (Nombre\_del\_grafo\*)**

## 3° TERCER PASO

- Agregamos el nombre del archivo de **xml**, por ejemplo (**Ejemplo1**) y pulsamos el **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 271: Nombre del (xml)*

---

**Nota:** Se creara el archivo en el directorio (/home/usuario/.safef/flowfiles/) con el nombre (**Ejemplo1.xml**)

---

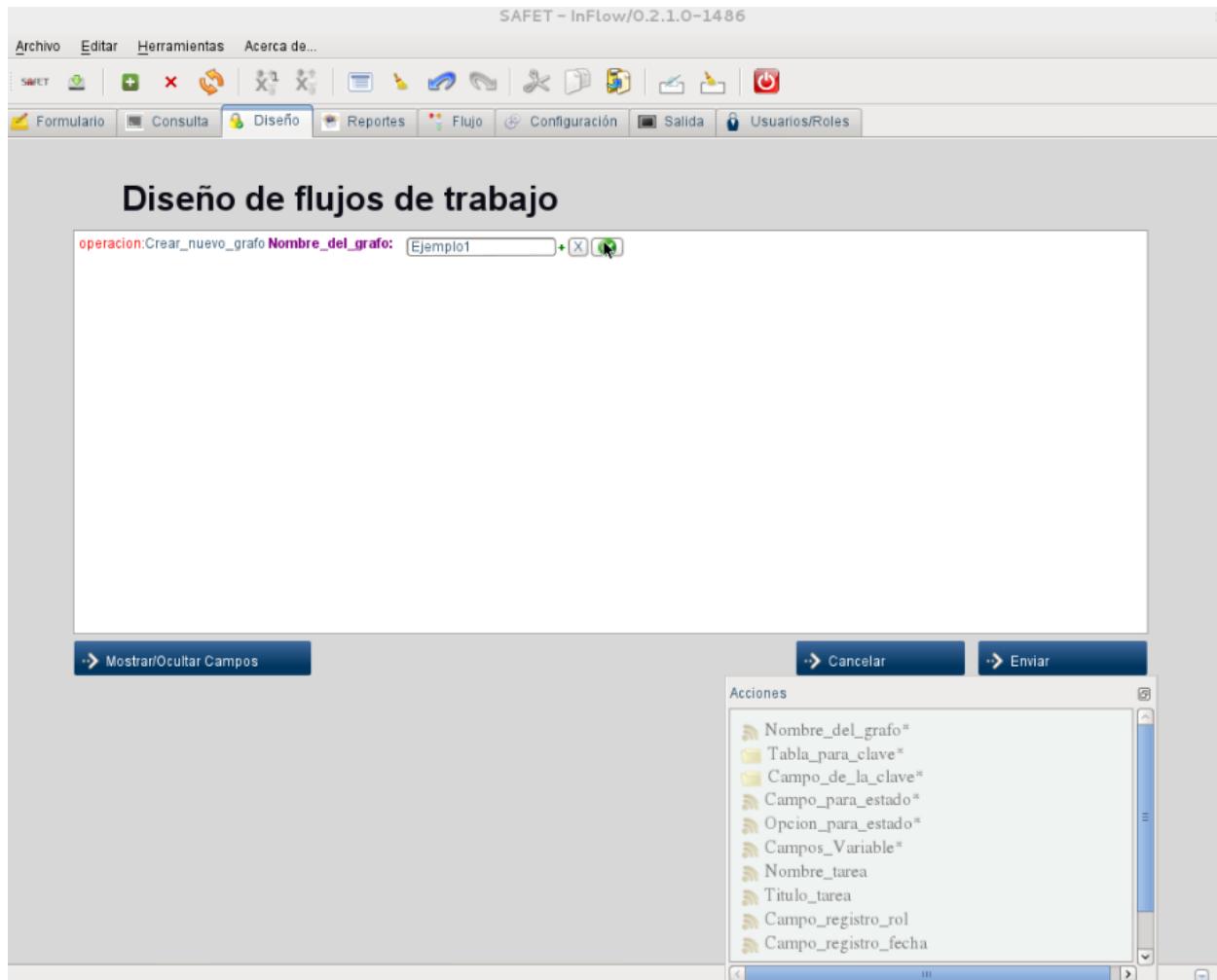


Figura 7.191: **Figura 271: Nombre del (xml)**

## 4° CUARTO PASO

- Damos click al siguiente campo obligatorio (**Tabla\_para\_clave\***), como se muestra en la siguiente *Figura 272: Campo (Tabla\_para\_clave\*)*

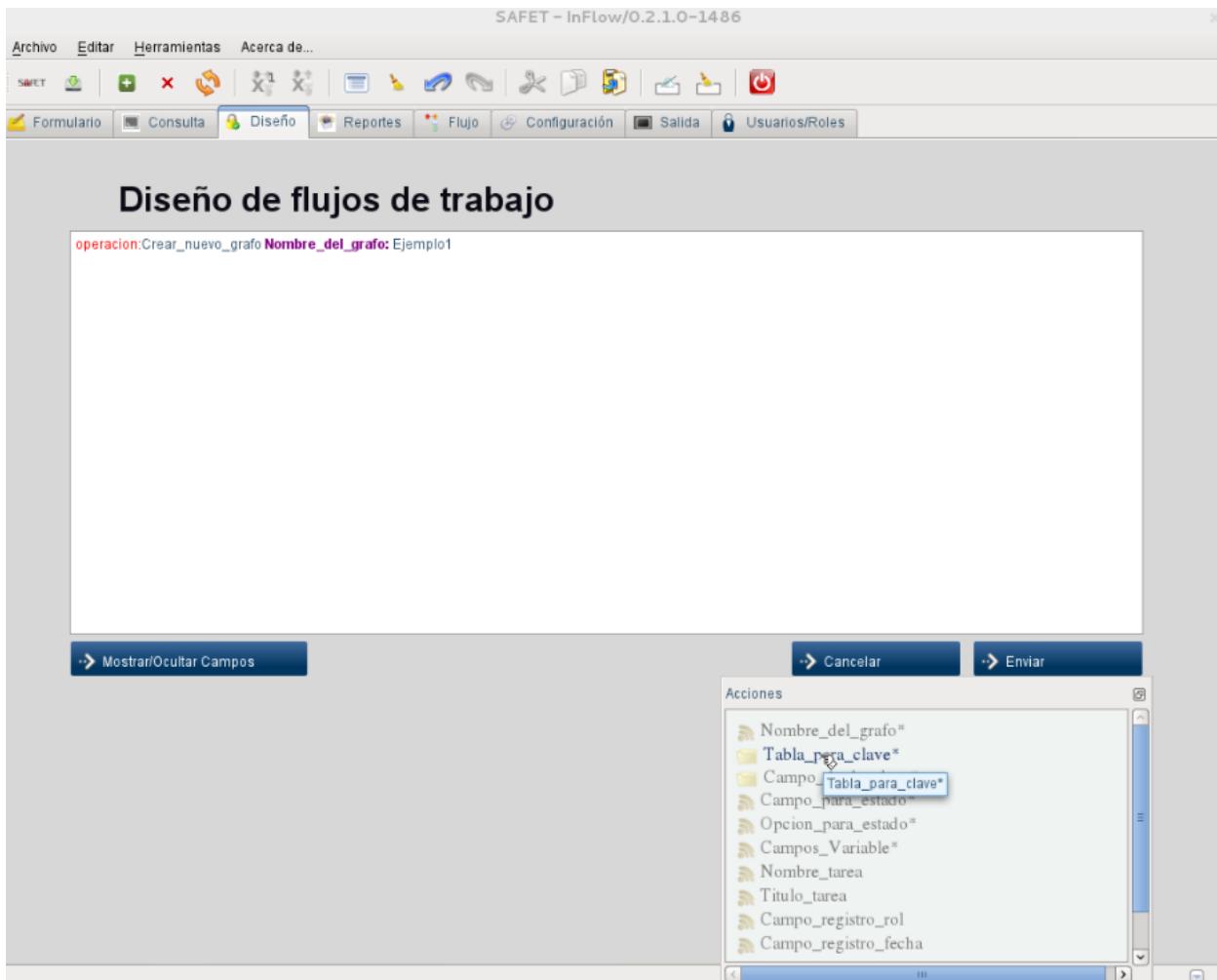


Figura 7.192: **Figura 272: Campo (Tabla\_para\_clave\*)**

## 5° QUINTO PASO

- Seleccionamos la tabla **productos**, como se muestra en la siguiente *Figura 273: Tabla productos*

## 6° SEXTO PASO

- Damos un click al botón con la flecha verde significando que ya está lleno el campo , como se muestra en la siguiente *Figura 274: Botón (Fin del campo)*

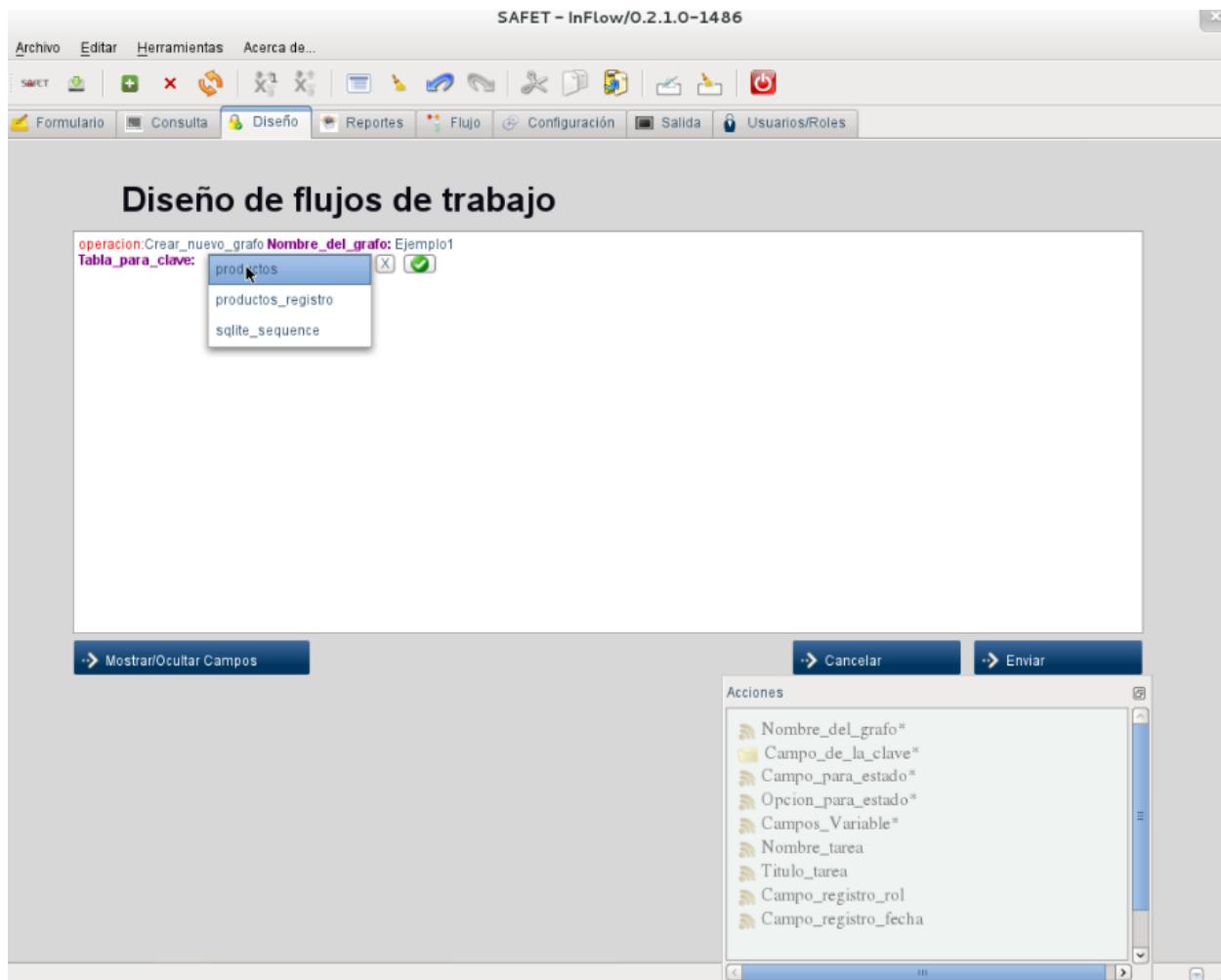


Figura 7.193: Figura 273: Tabla productos

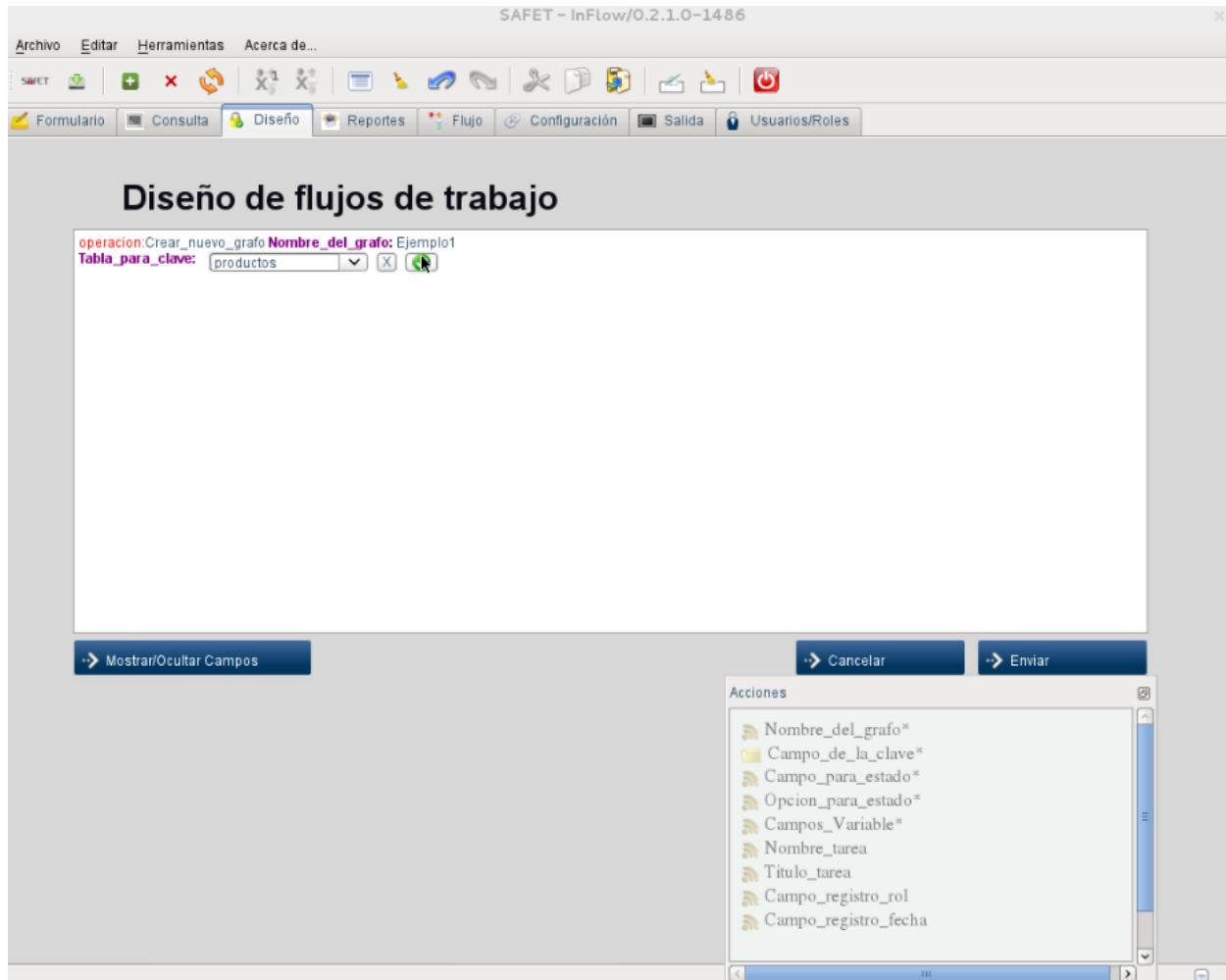


Figura 7.194: Figura 274: Botón (Fin del campo)

## 7° SEPTIMO PASO

- Damos click al siguiente campo obligatorio (**Campo\_de\_la\_clave\***), como se muestra en la siguiente *Figura 275: Campo (Campo\_de\_la\_clave\*)*

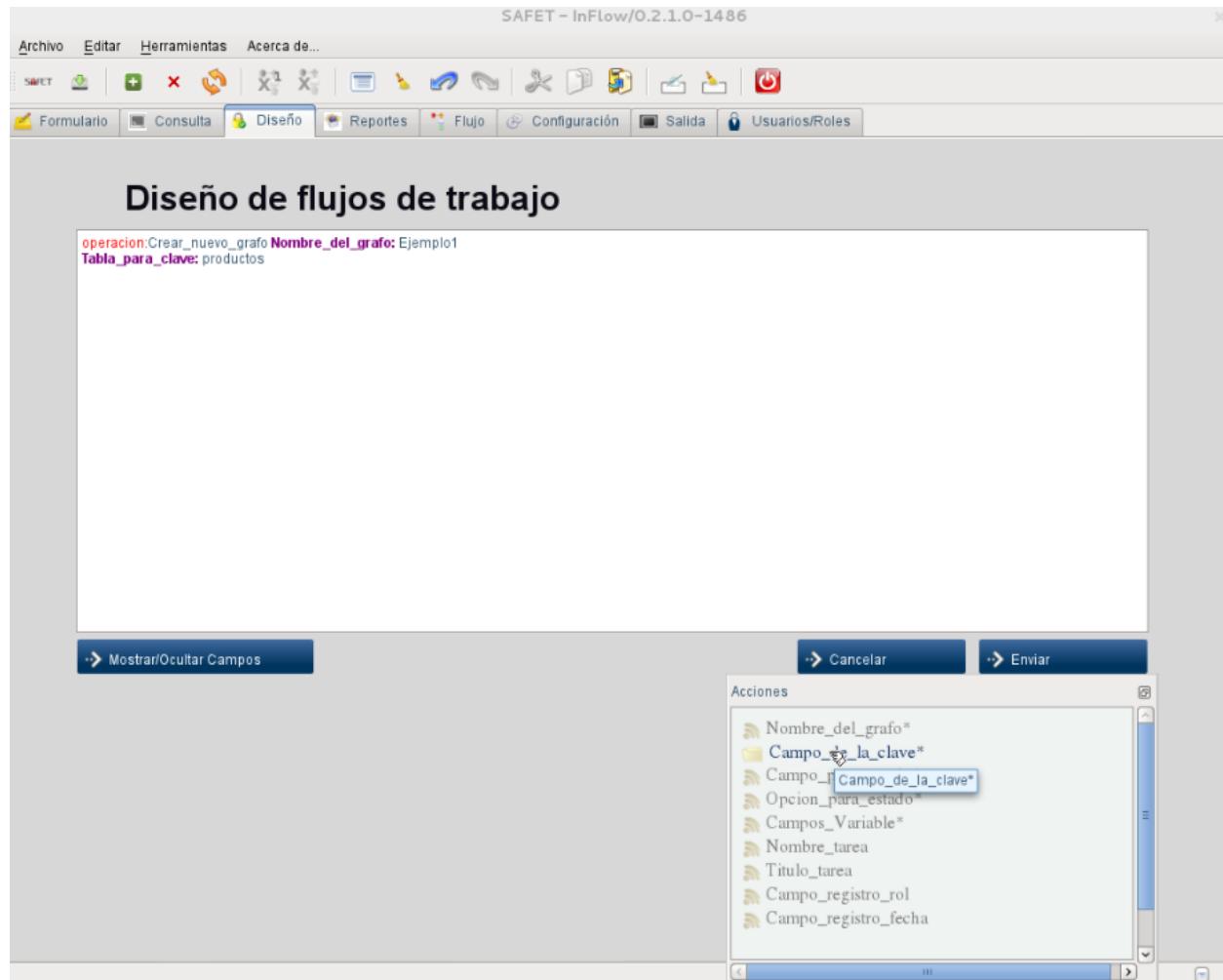


Figura 7.195: *Figura 275: Campo (Campo\_de\_la\_clave\*)*

## 8° OCTAVO PASO

- Seleccionamos el campo clave del productos (**id**), como se muestra en la siguiente *Figura 276: Campo (id)*

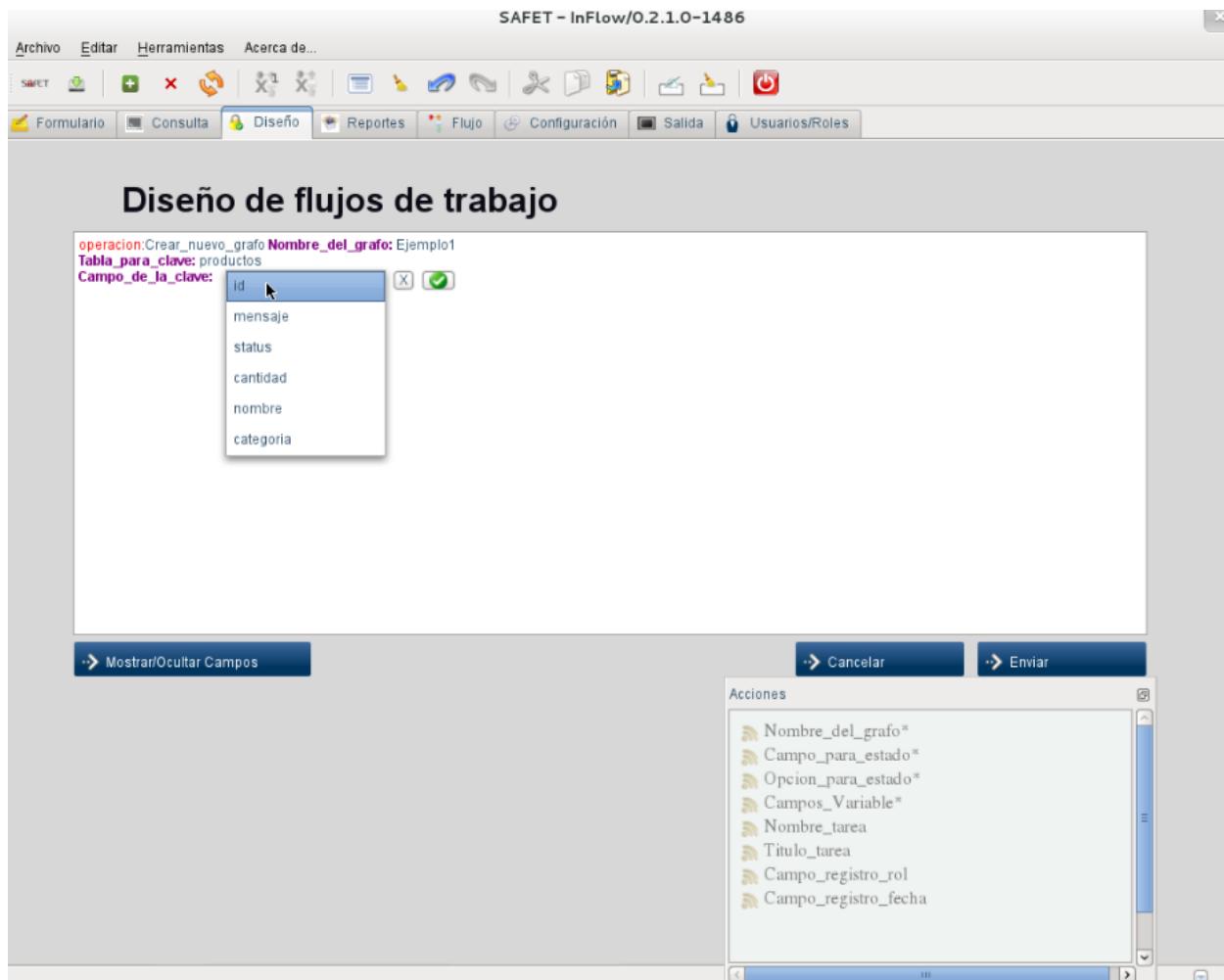


Figura 7.196: Figura 276: Campo (id)

### 9° NOVENO PASO

- Damos un click al botón con la flecha verde significando que ya está lleno el campo , como se muestra en la siguiente *Figura 277: Botón (Fin del campo)*

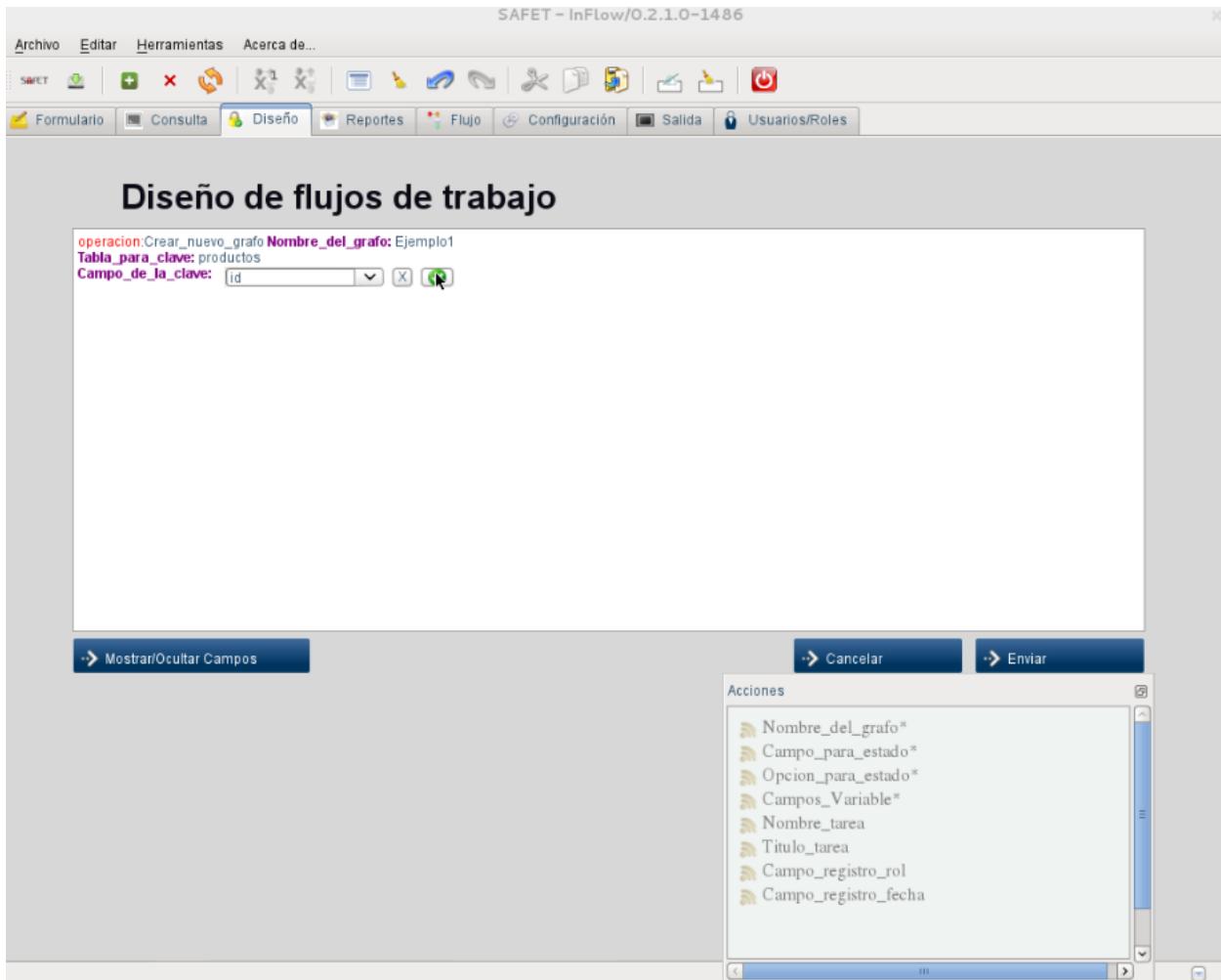


Figura 7.197: **Figura 277: Botón (Fin del campo)**

### 10° DECIMO PASO

- Damos click al siguiente campo obligatorio (**Campo\_para\_estado\***), como se muestra en la siguiente *Figura 278: Campo (Campo\_para\_estado\*)*

### 11° DECIMO PRIMERO PASO

- Seleccionamos el campo (**cantidad**), como se muestra en la siguiente *Figura 279: Campo (cantidad)*

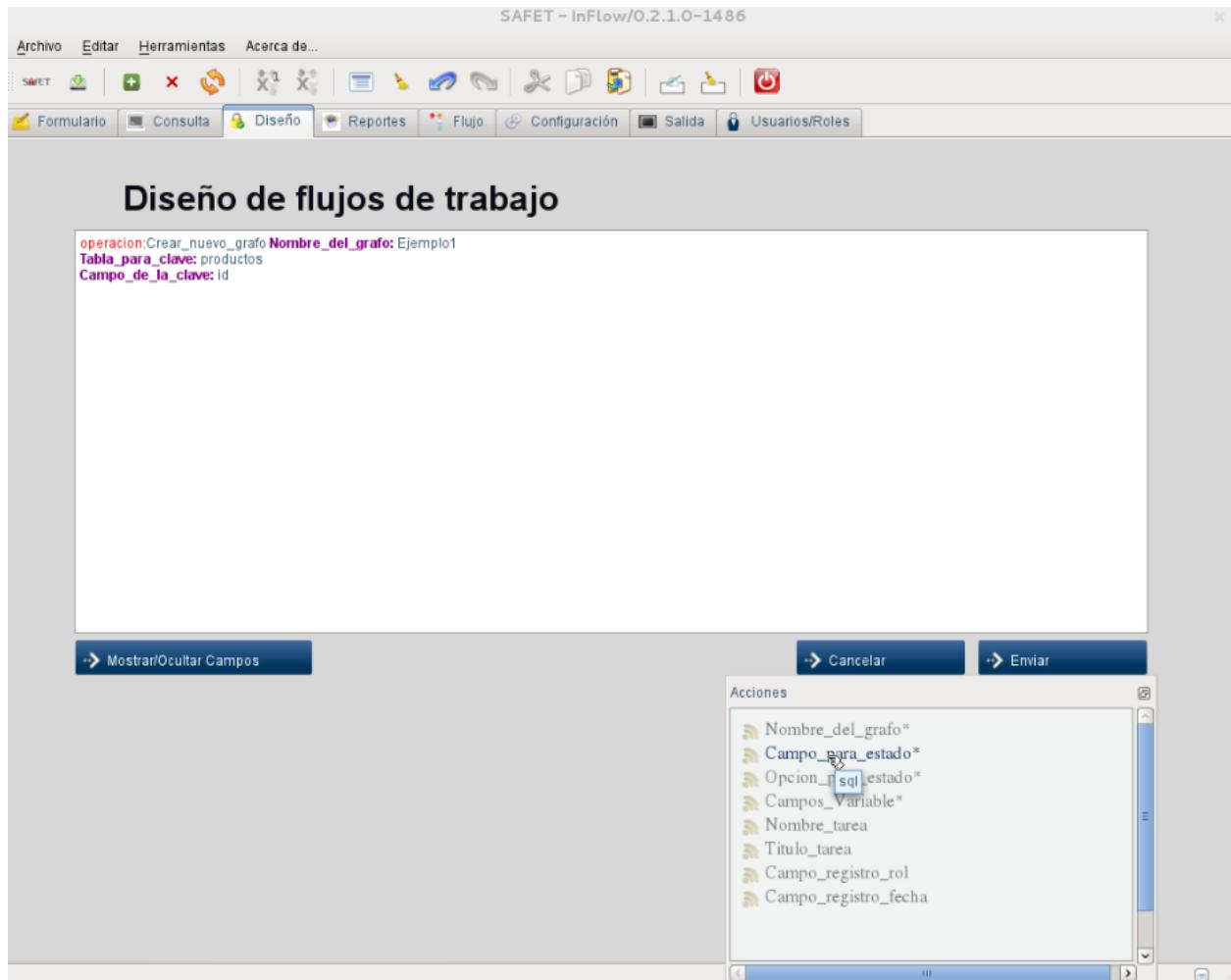


Figura 7.198: Figura 278: Campo (Campo\_para\_estado\*)

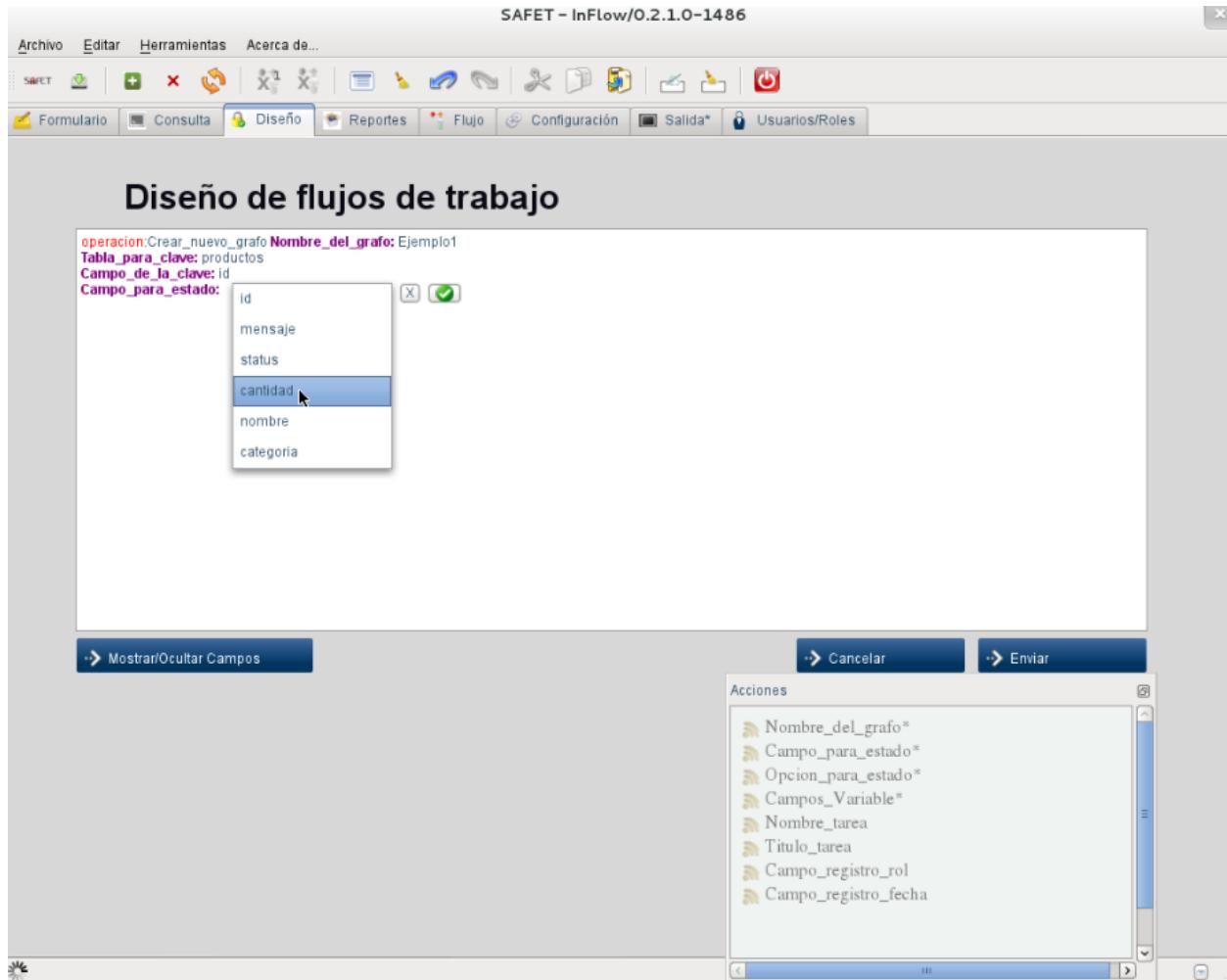


Figura 7.199: Figura 279: Campo (cantidad)

## 12° DECIMO SEGUNDO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo , como se muestra en la siguiente *Figura 280: Botón (Fin de campo)*

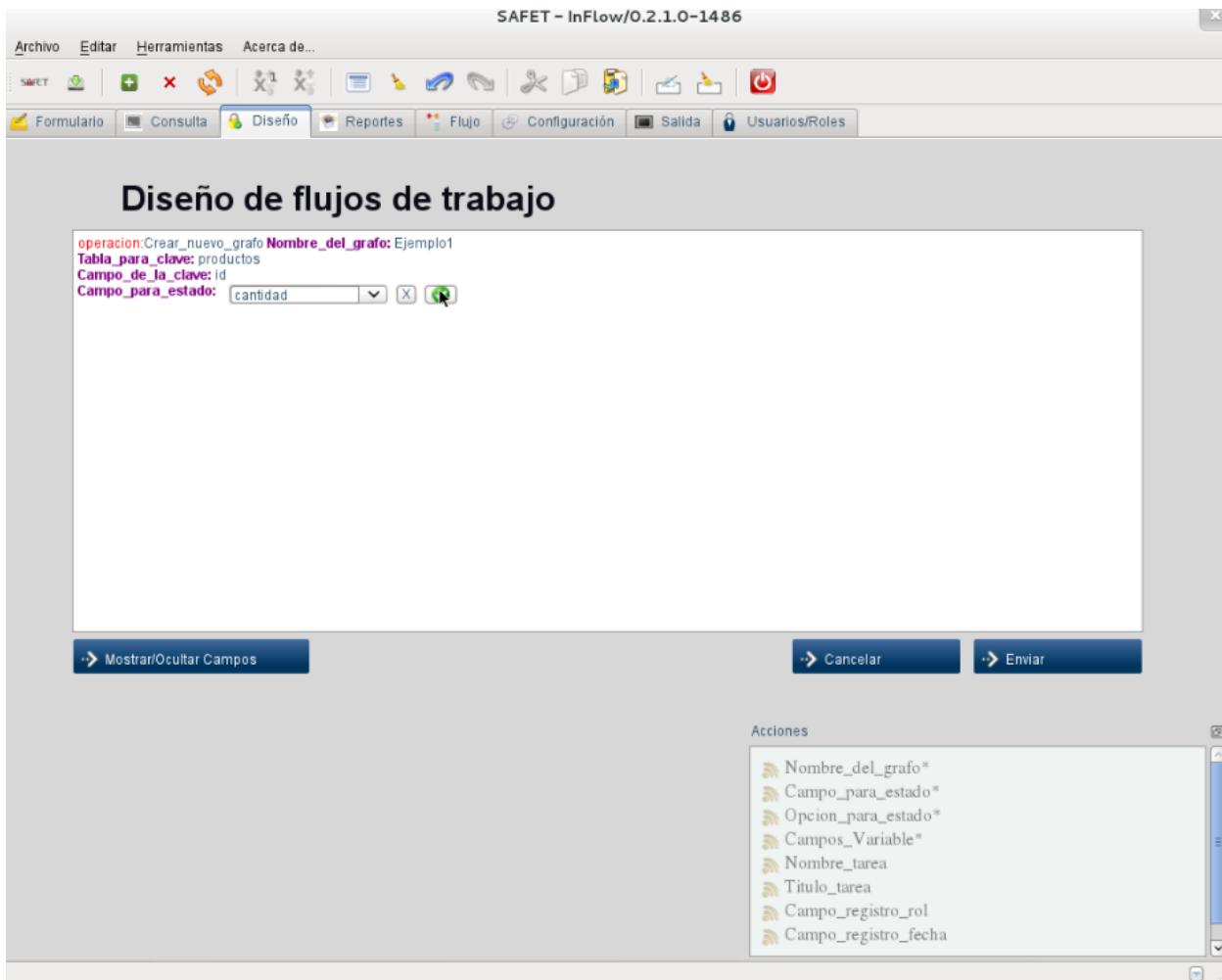


Figura 7.200: Figura 280: Botón (Fin de campo)

## 13° DECIMO TERCER PASO

- Damos click al siguiente campo obligatorio (**Opcion\_para\_estado\***), como se muestra en la siguiente *Figura 281: Campo (Opcion\_para\_estado\*)*

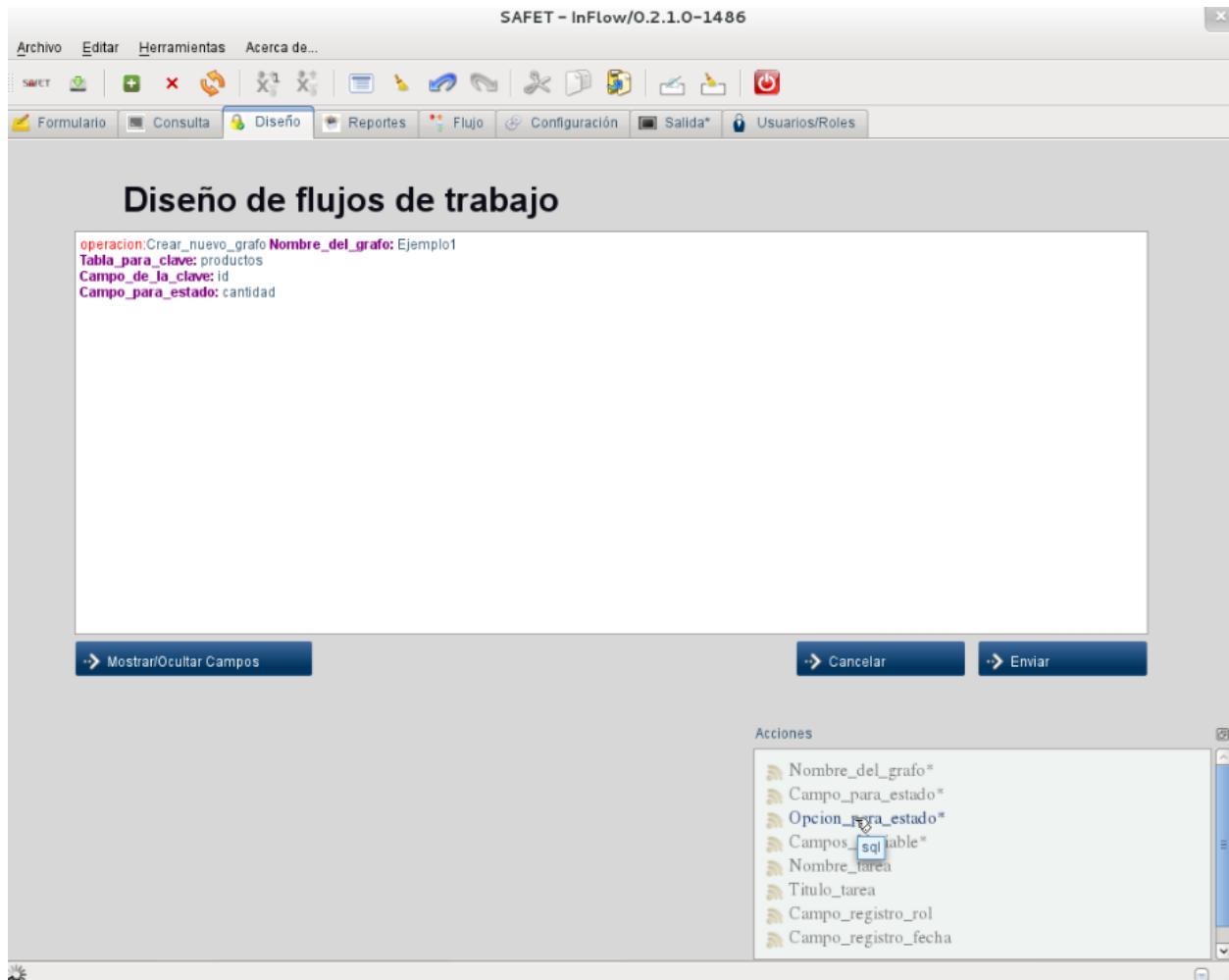


Figura 7.201: Figura 281: Campo (Opcion\_para\_estado\*)

## 14° DECIMO CUARTO PASO

- Agregamos la opción de búsqueda del datos, por ejemplo que muestre cuantos productos hay mayor a o igual a 10, colocamos la opción ( $>=10$ ) y pulsamos el botón con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 282: Opción (>=10)*

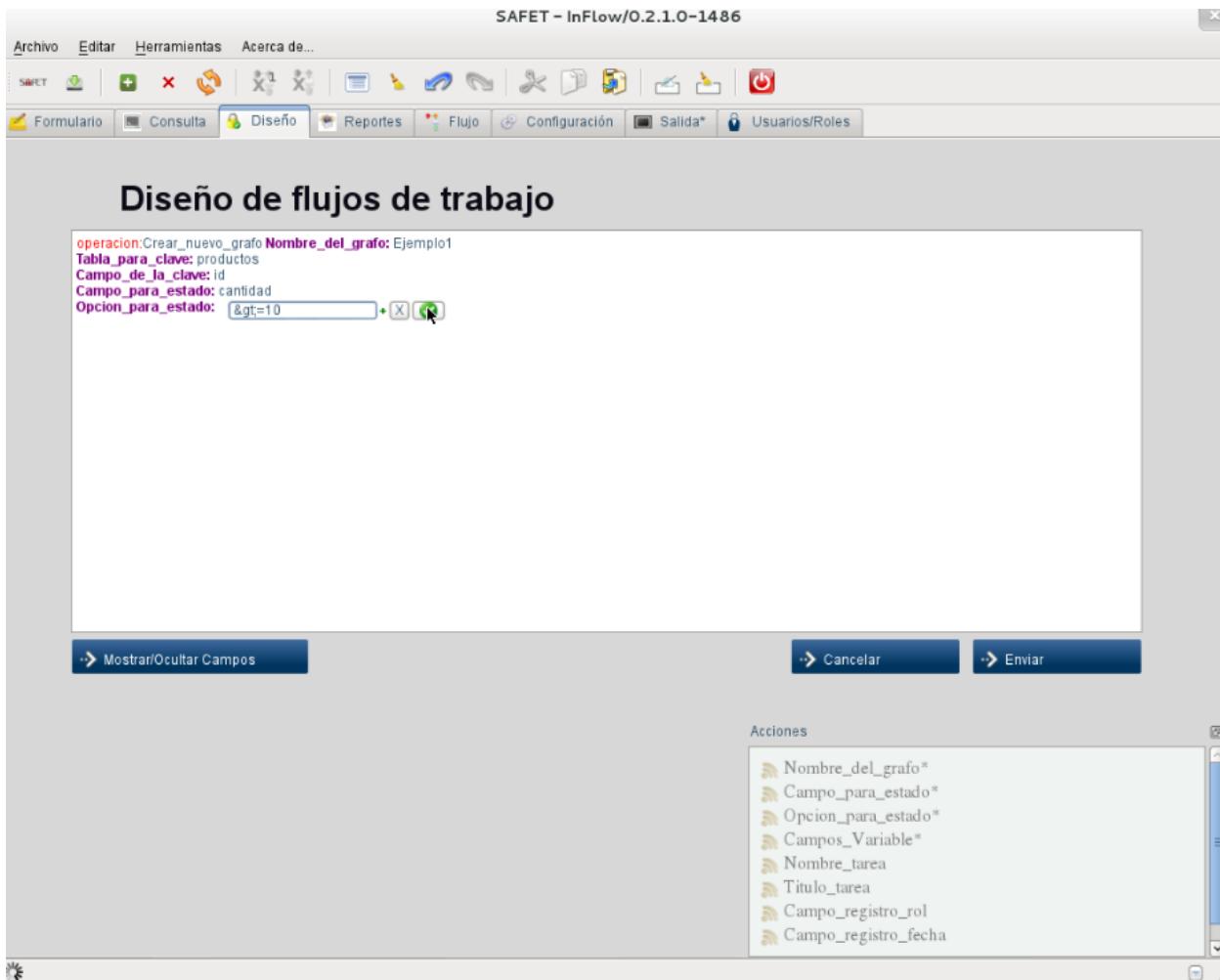


Figura 7.202: **Figura 282: Opción (>=10)**

## 15° DECIMO QUINTO PASO

- Damos click al siguiente campo obligatorio (**Campos\_Variable\***), como se muestra en la siguiente *Figura 283: Campo (Campos\_Variable\*)*

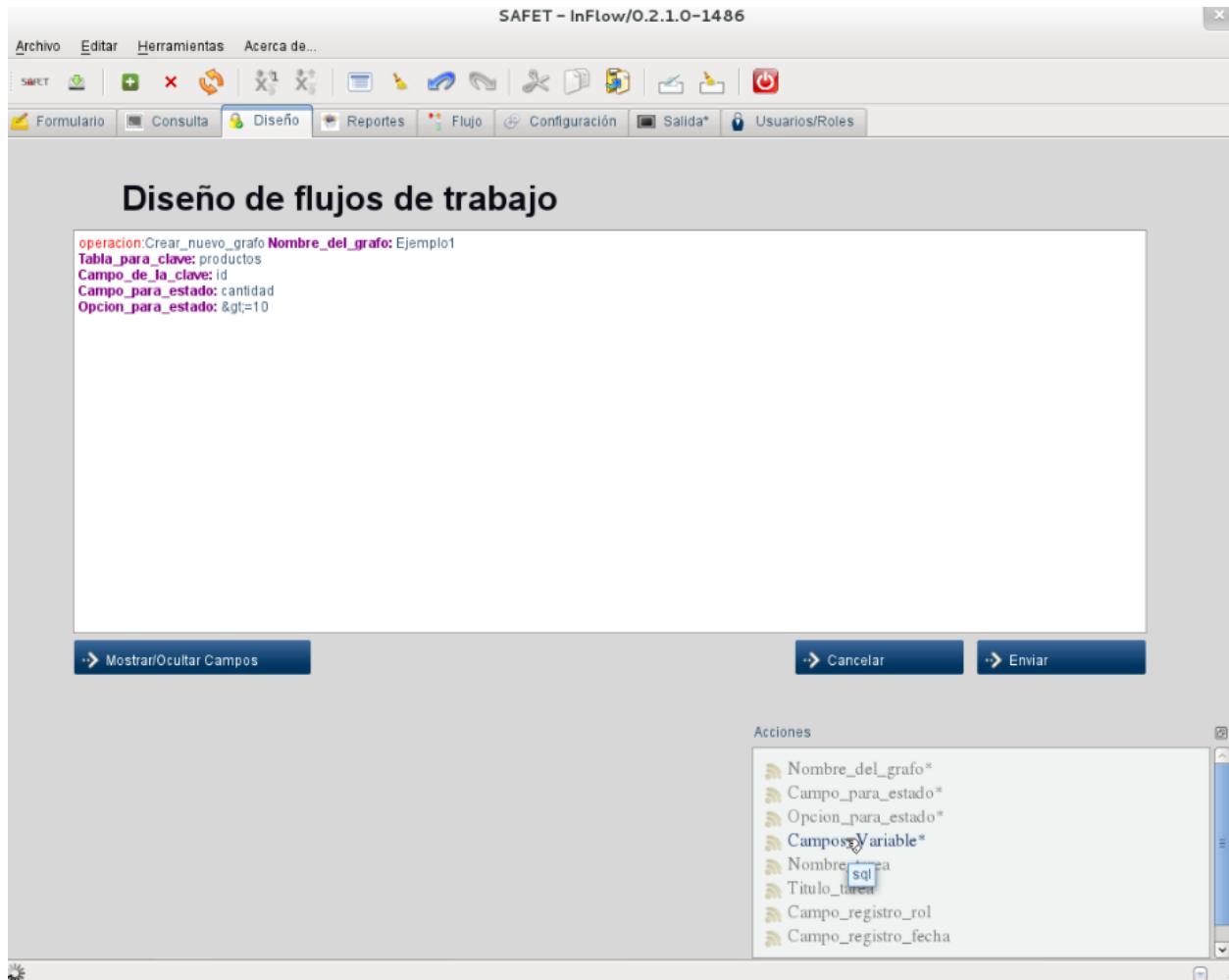


Figura 7.203: Figura 283: Campo (Campos\_Variable\*)

## 16° DECIMO SEXTO PASO

- Seleccionamos los campos (**mensaje,status,cantidad,nombre**) y pulsamos el **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 284: Selección de Campos*

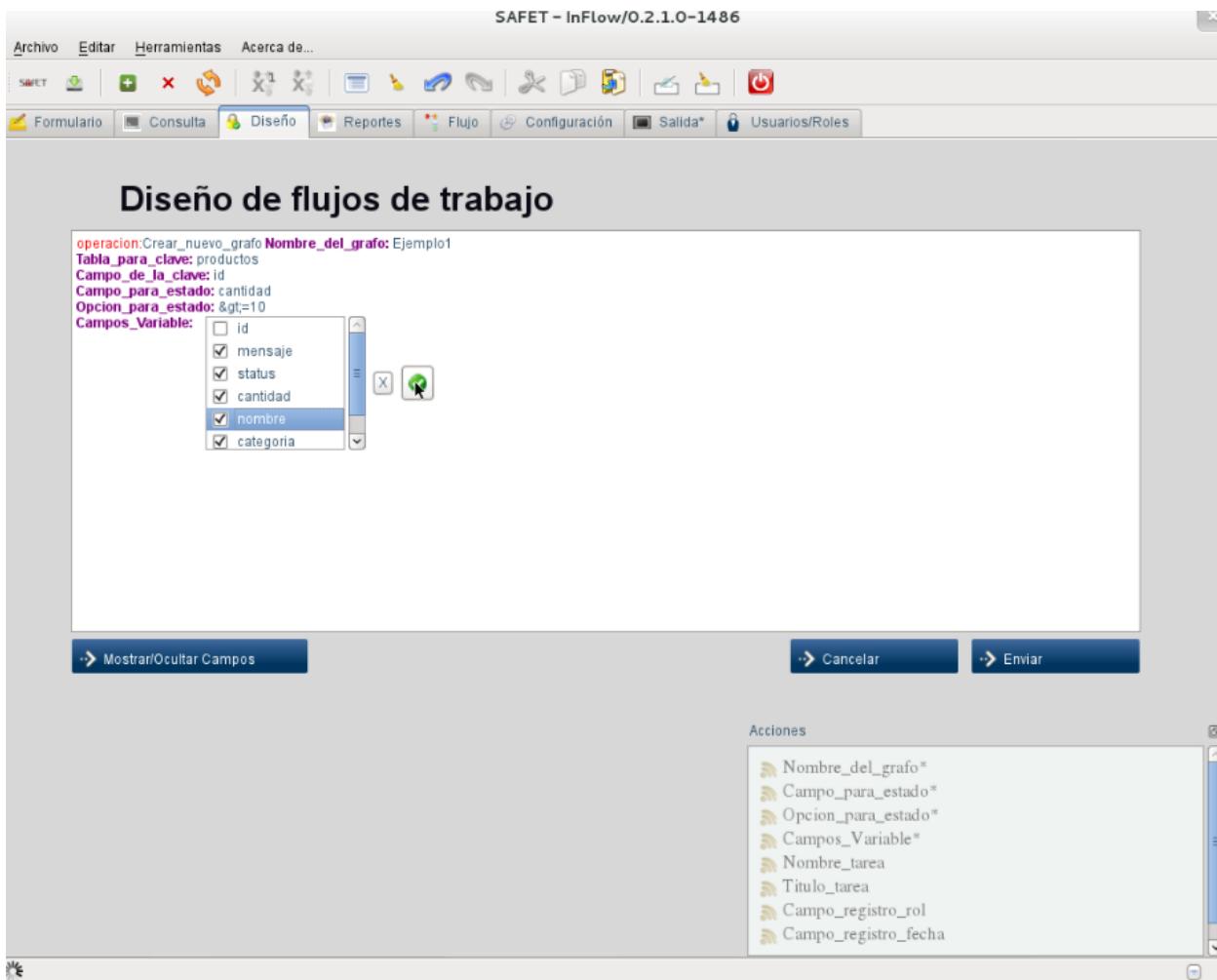


Figura 7.204: Figura 284: Selección de Campos

## 17° DECIMO SEPTIMO PASO

- Damos click al siguiente campo opcional (**Nombre\_tarea**), como se muestra en la siguiente *Figura 285: Campo (Nombre\_tarea)*

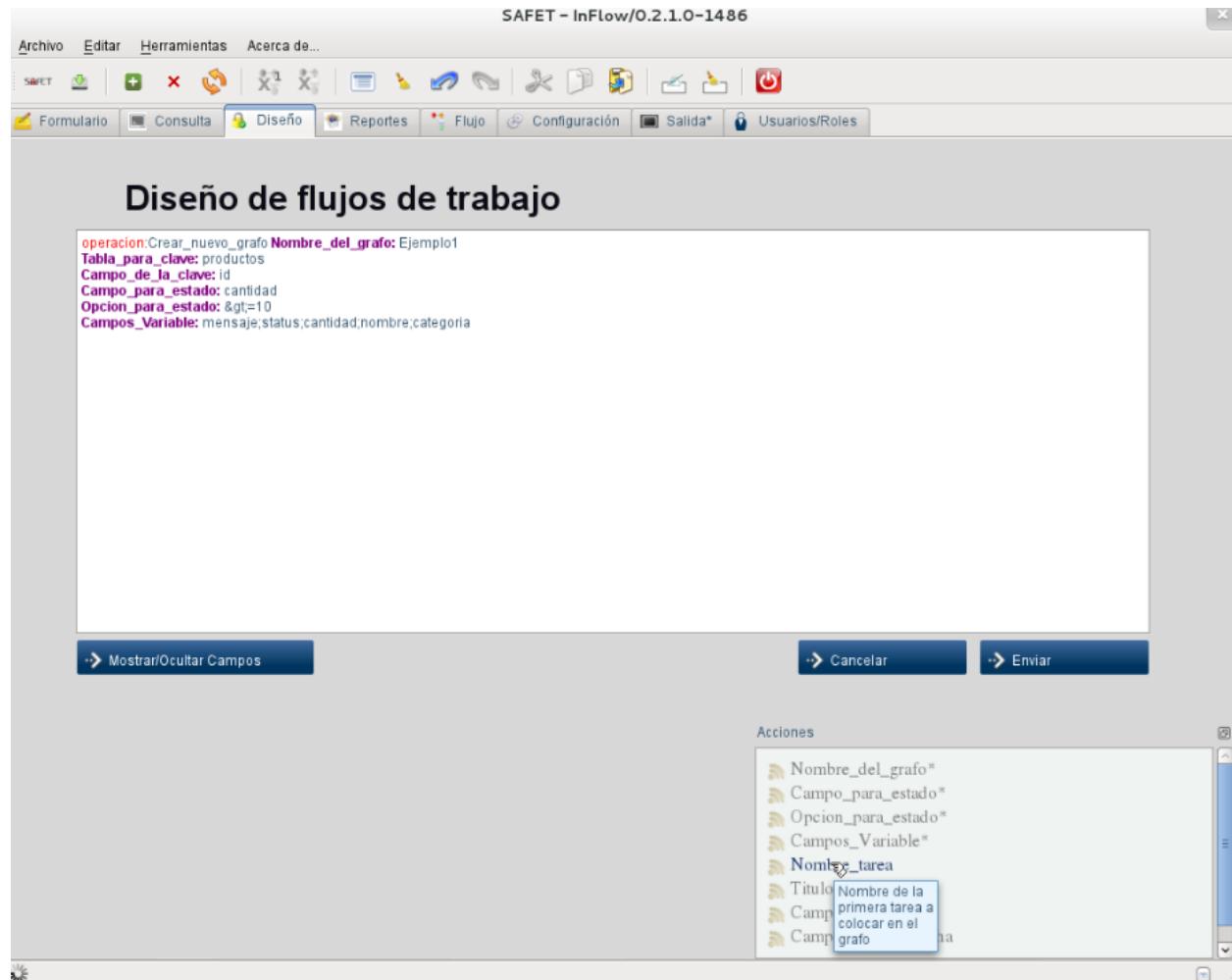


Figura 7.205: Figura 285: Campo (Nombre\_tarea)

## 18° DECIMO OCTAVO PASO

- Agregamos el nombre de la tarea por ejemplo (**Cantidad10**) y pulsamos el **botón con la flecha verde** significando que ya está lleno el campo, como se muestra en la siguiente *Figura 286: Nombre de la tarea*

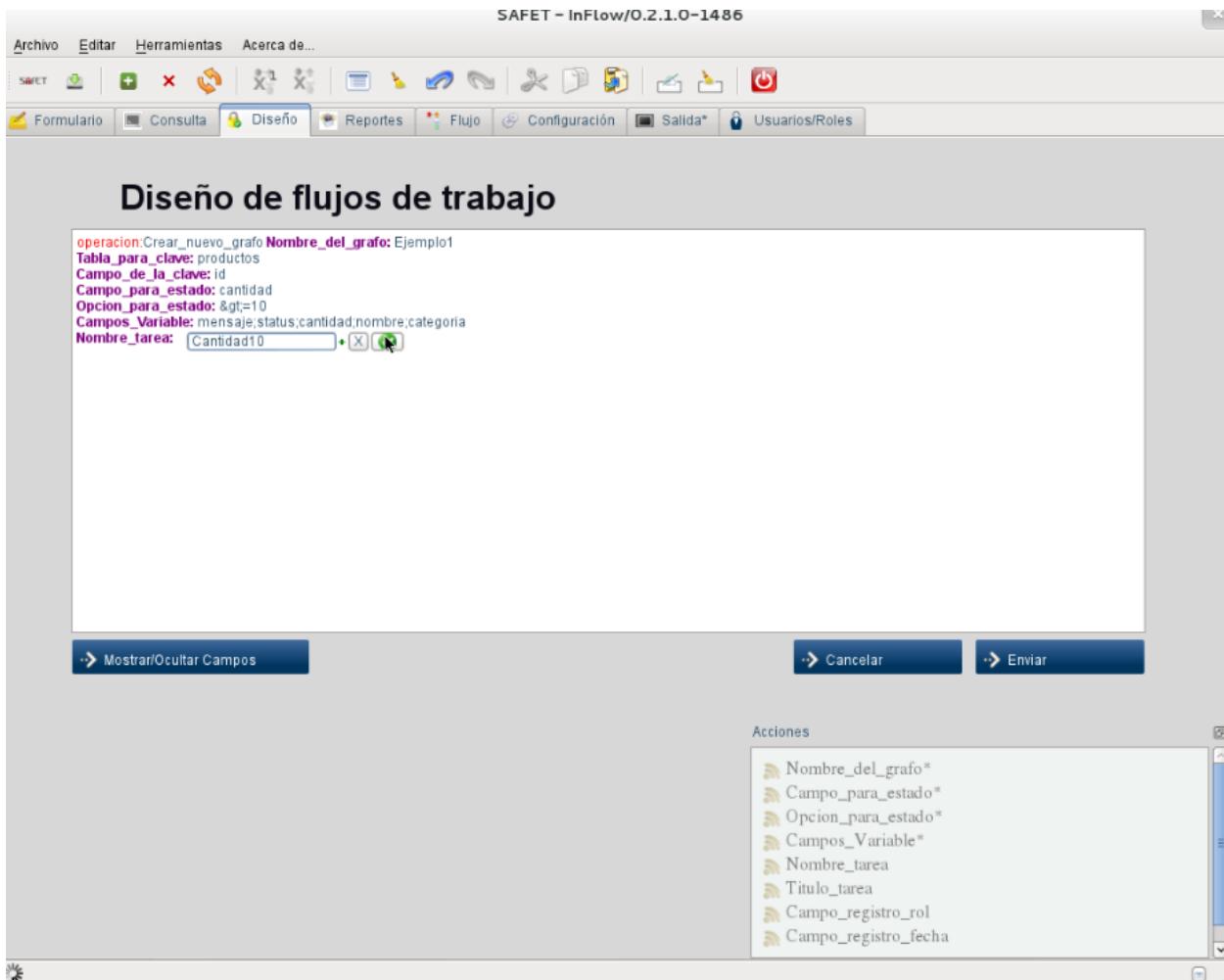


Figura 7.206: **Figura 286: Nombre de la tarea**

## 19° DECIMO NOVENO PASO

- Damos click al siguiente campo opcional (**Titulo\_tarea**), como se muestra en la siguiente *Figura 287: Campo (Titulo\_tarea)*

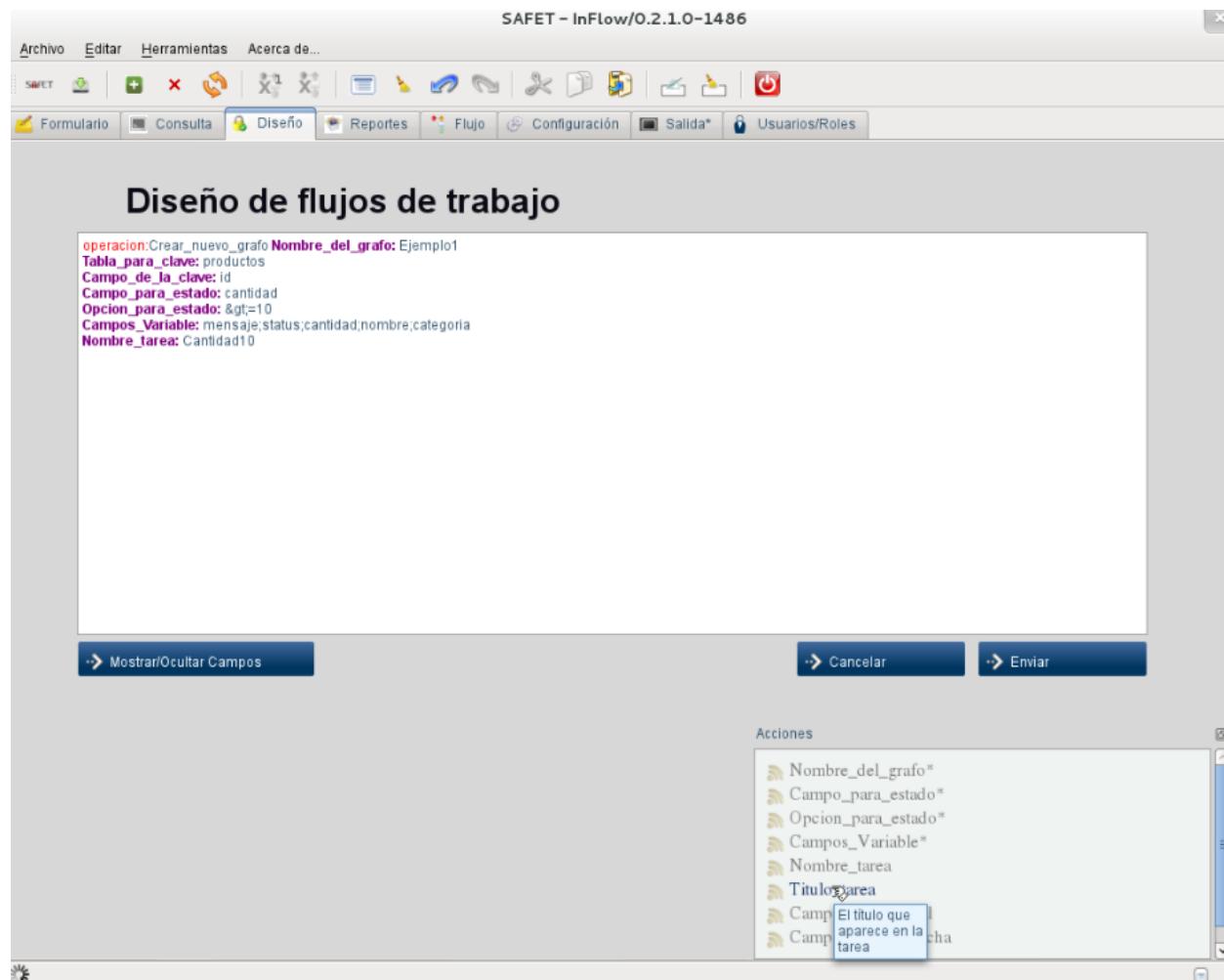


Figura 7.207: Figura 287: Campo (Titulo\_tarea)

## 20° VIGÉSIMO PASO

- Agregamos el título de la tarea por ejemplo (**Tarea1**) y pulsamos el botón con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 288: Titulo de la tarea*

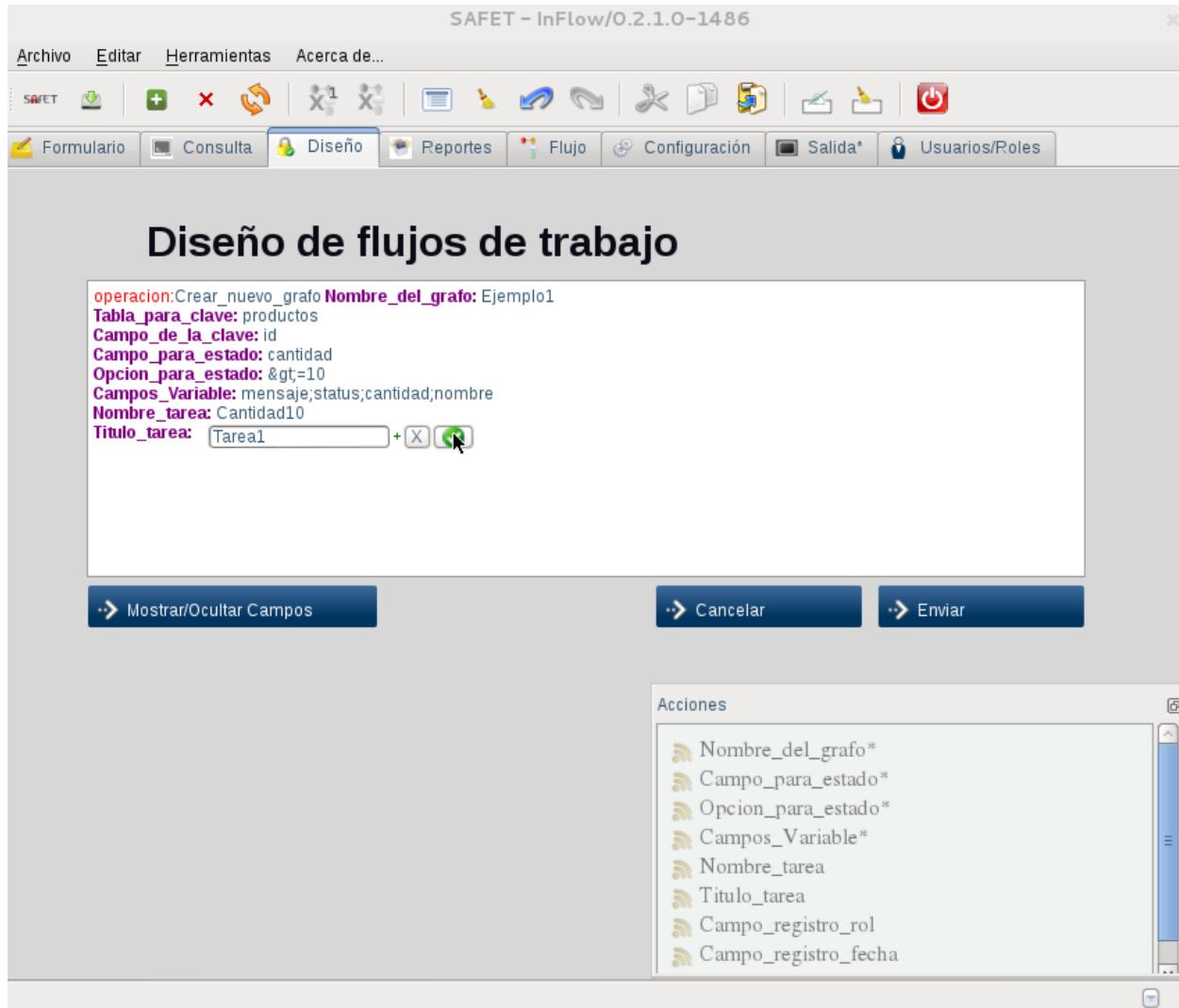


Figura 7.208: **Figura 288: Titulo de la tarea**

## 21° VIGÉSIMO PRIMERO PASO

- Damos un click al botón (**Enviar**) para finalizar con la operación la cual se nos mostrara como resultado (**La operación grafo fue exitosa y se donde se escribe el archivo creado**), como se muestra en la siguiente *Figura 289: Fin de la operación*

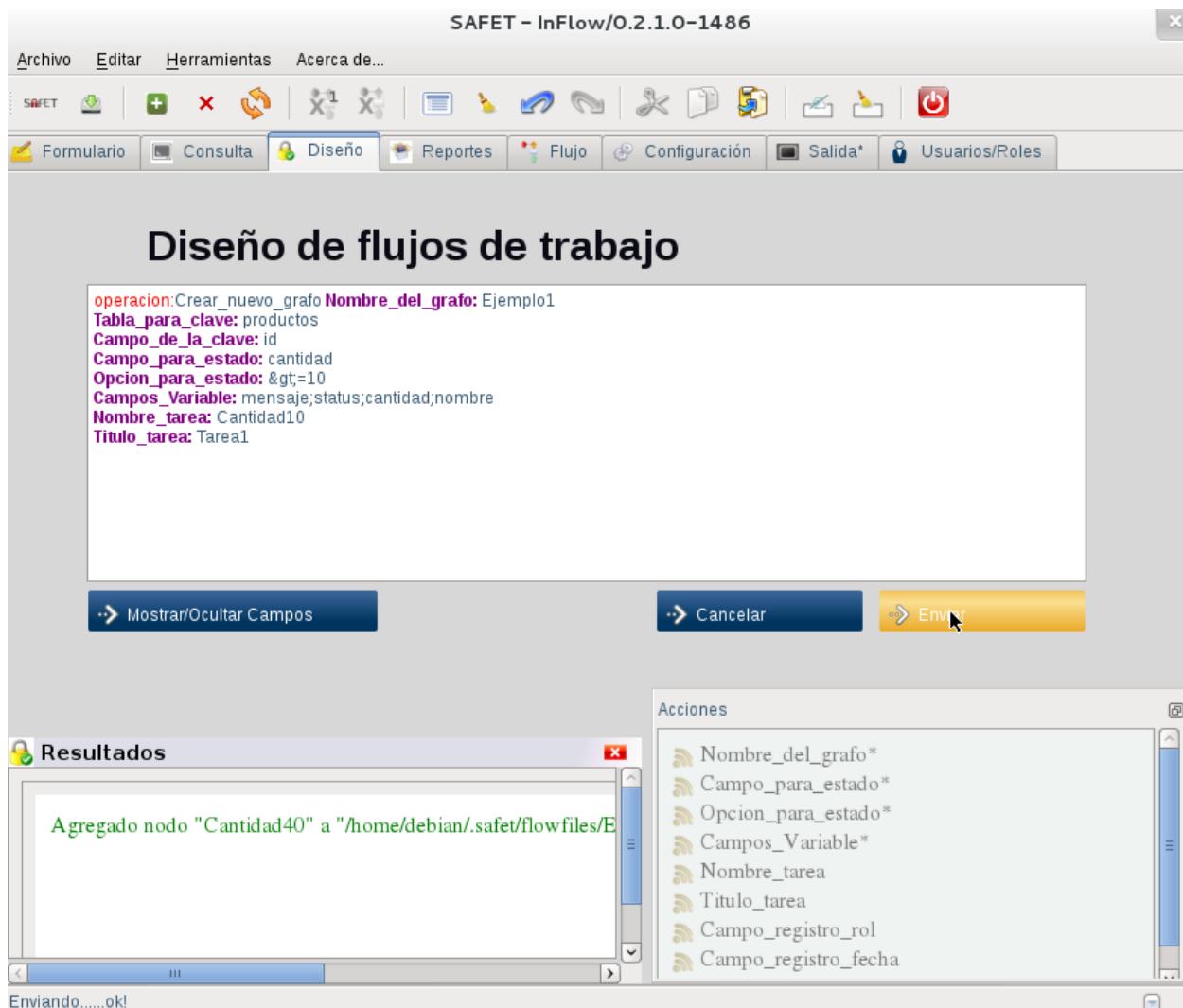


Figura 7.209: Figura 289: Fin de la operación

### 7.9.3 C.- Observemos el grafo creado en el flujo de trabajo (.xml)

Para observar la gráfica del archivo (**Ejemplo1.xml**) del directorio (**/home/usuario/.safet/flowfiles/**), para ello damos click al siguiente **link**. *B.- PRIMERA OPERACIÓN (Gráfico coloreado general)*

Nota: Se nos mostrara el reporte del gráfico de la siguiente manera *Figura 290: Gráfico*

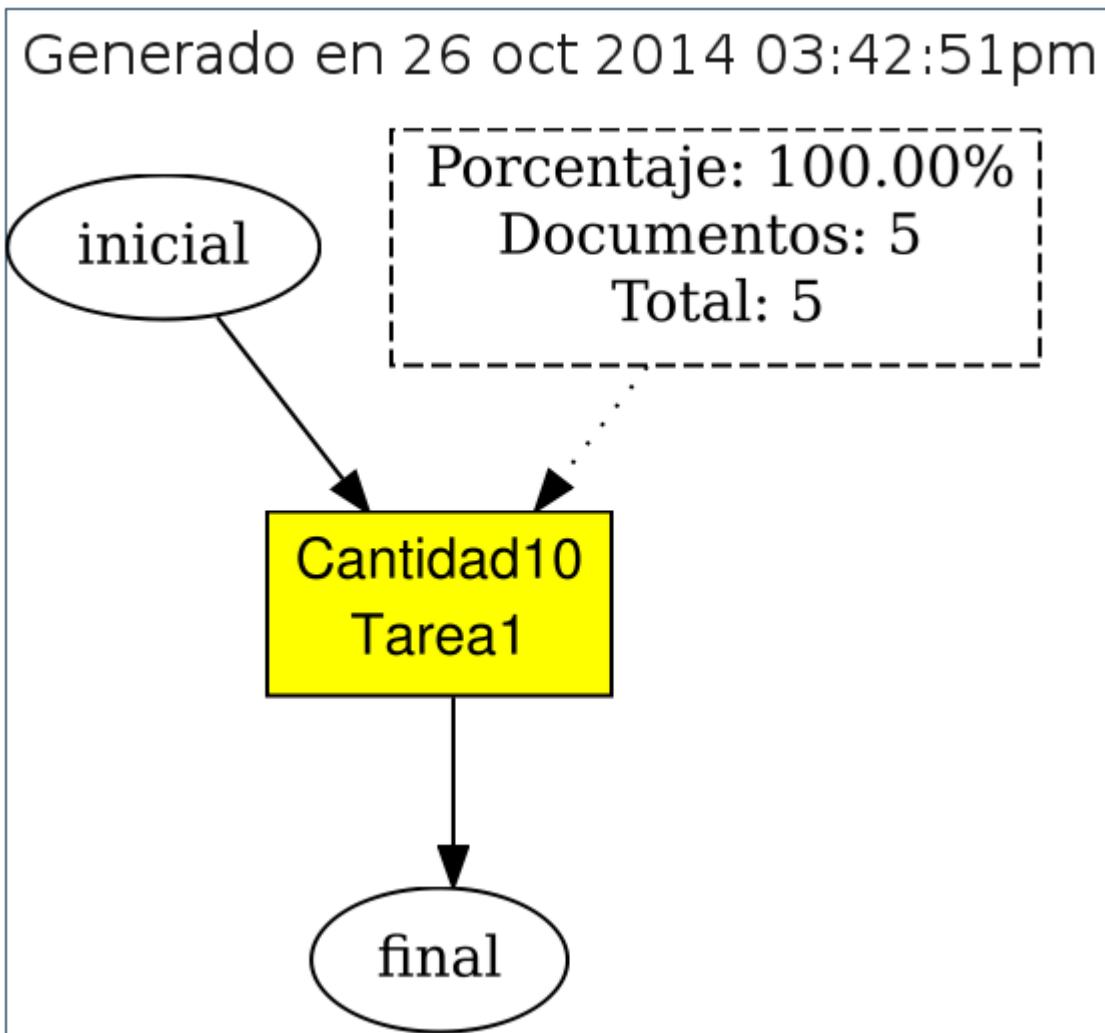


Figura 7.210: **Figura 290: Gráfico**

## 7.10 Diseño de agregar nuevos nodos al gráfico (.xml)

### 7.10.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 291: Autenticación de Inflow*.



Figura 7.211: *Figura 291: Autenticación de Inflow*.

#### 3° TERCER PASO

- Damos click en la segunda opción (**Firmar/verificar documento**), como se muestra en la siguiente *Figura 292: Opcion Diseño*

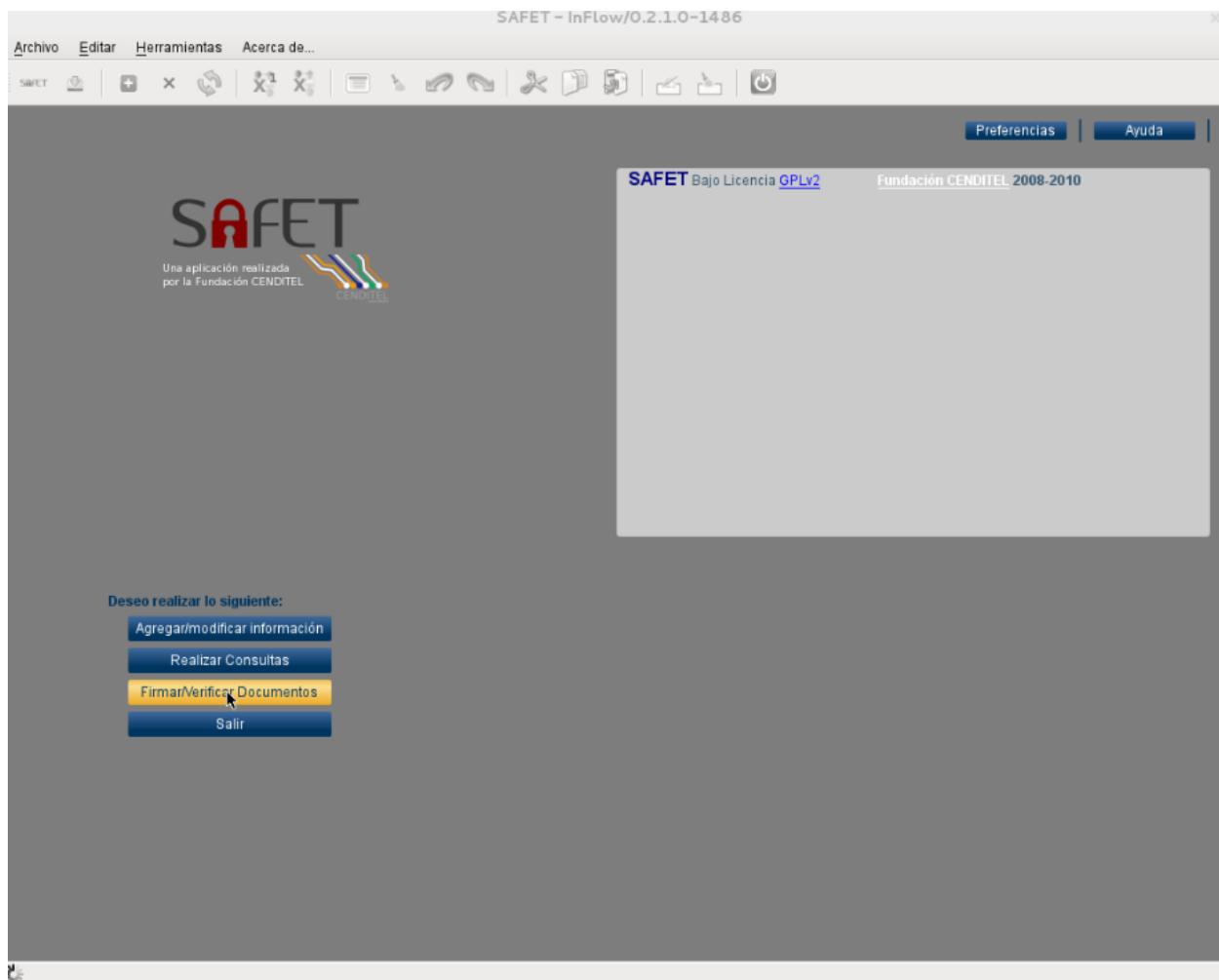


Figura 7.212: **Figura 292: Opcion Diseño**

### 4° CUARTO PASO

- Damos click a la segunda opción (**Mostar/Ocultar Campos**), como se muestra en la siguiente *Figura 293: Campos del Diseño*

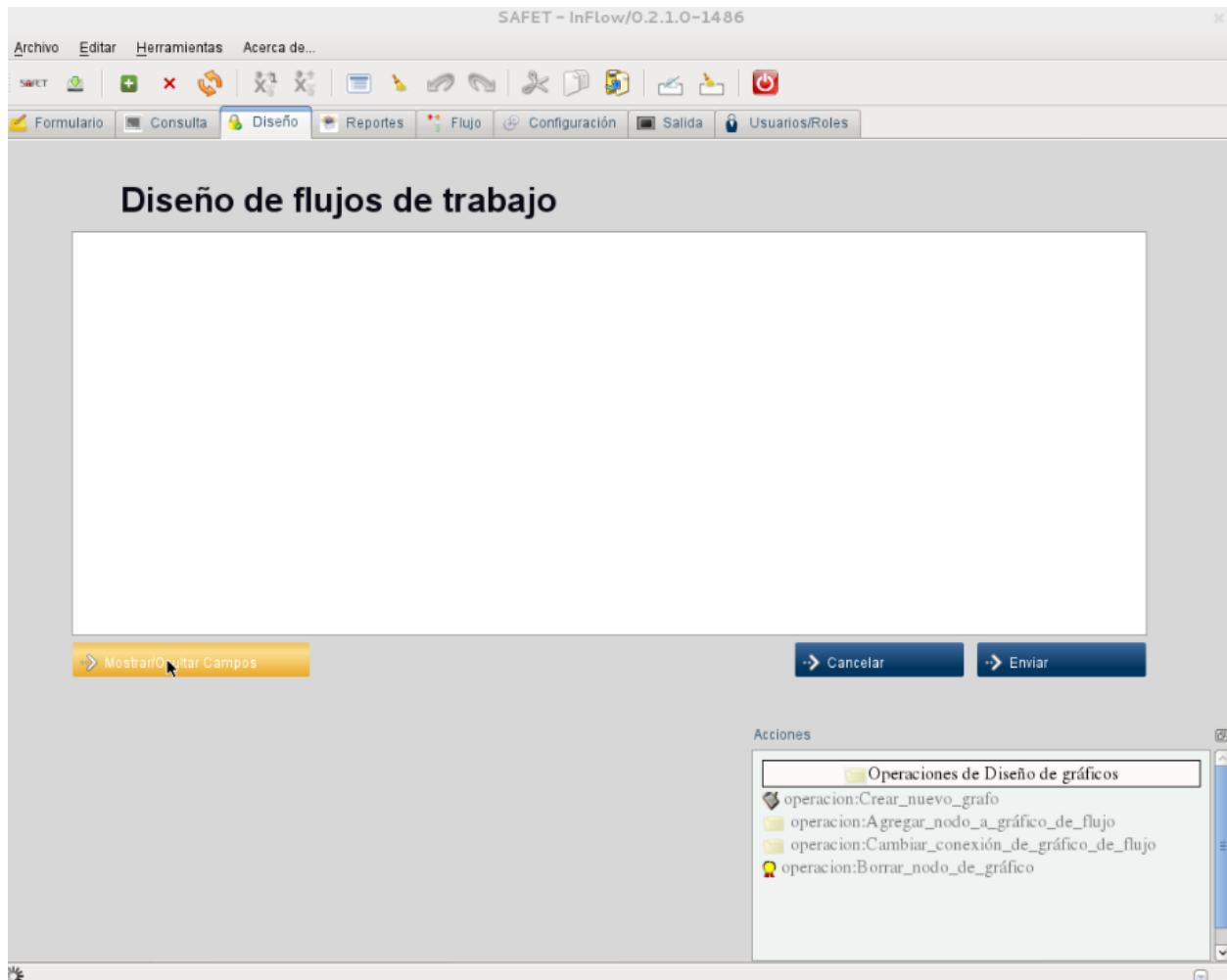


Figura 7.213: **Figura 293: Campos del Diseño**

### 7.10.2 B.- Agregar Nodo al flujo de trabajo o gráfo

#### 1° PRIMER PASO

- Damos click a la operación (**Agregar\_nodo\_a\_gráfico\_de\_flujo**), como se muestra en la siguiente *Figura 294: Agregar\_nodo\_a\_gráfico\_de\_flujo*

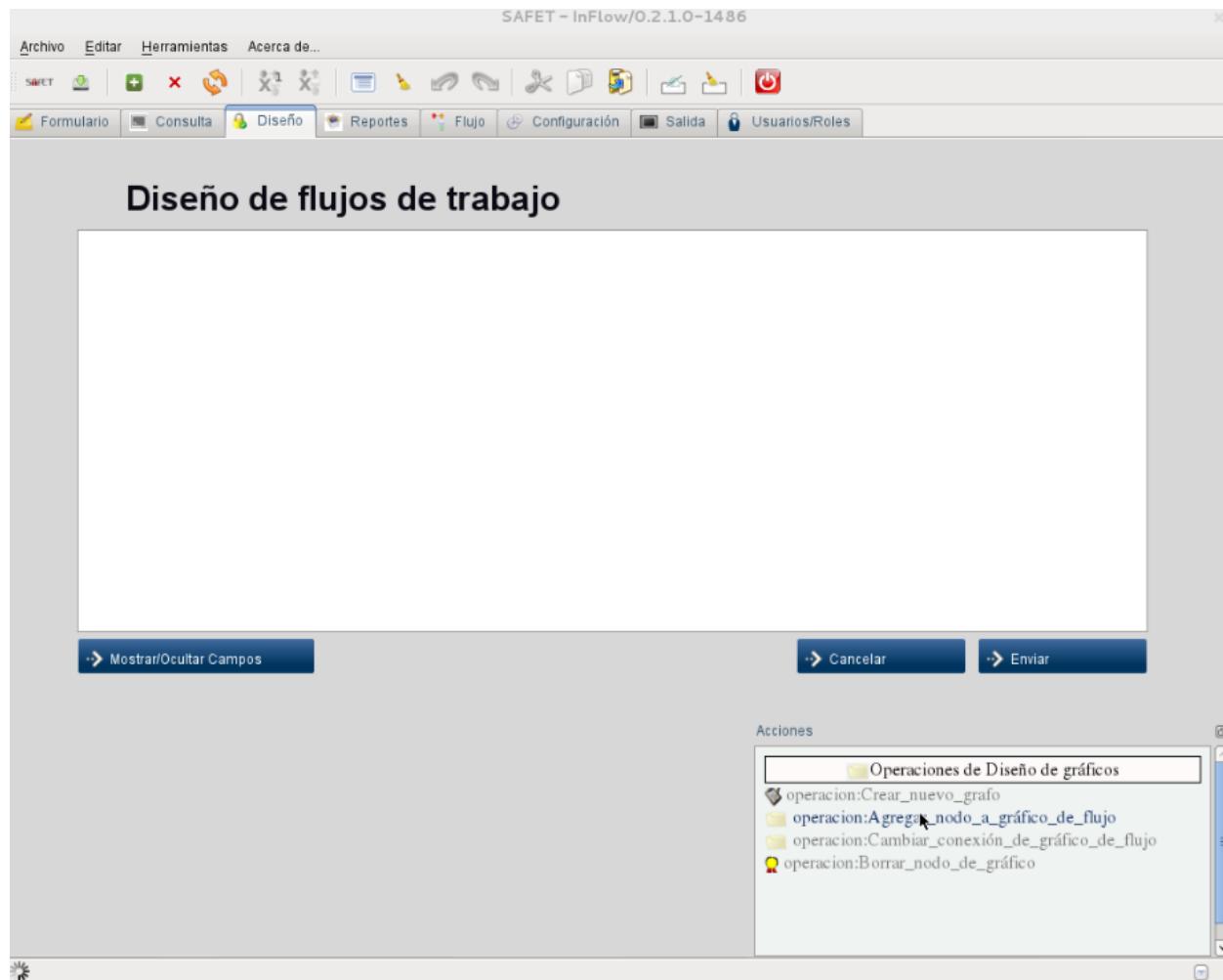


Figura 7.214: Figura 294: Agregar\_nodo\_a\_gráfico\_de\_flujo

### 2° SEGUNDO PASO

- Damos click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 295: Campo (Cargar\_archivo\_flujo\*)*

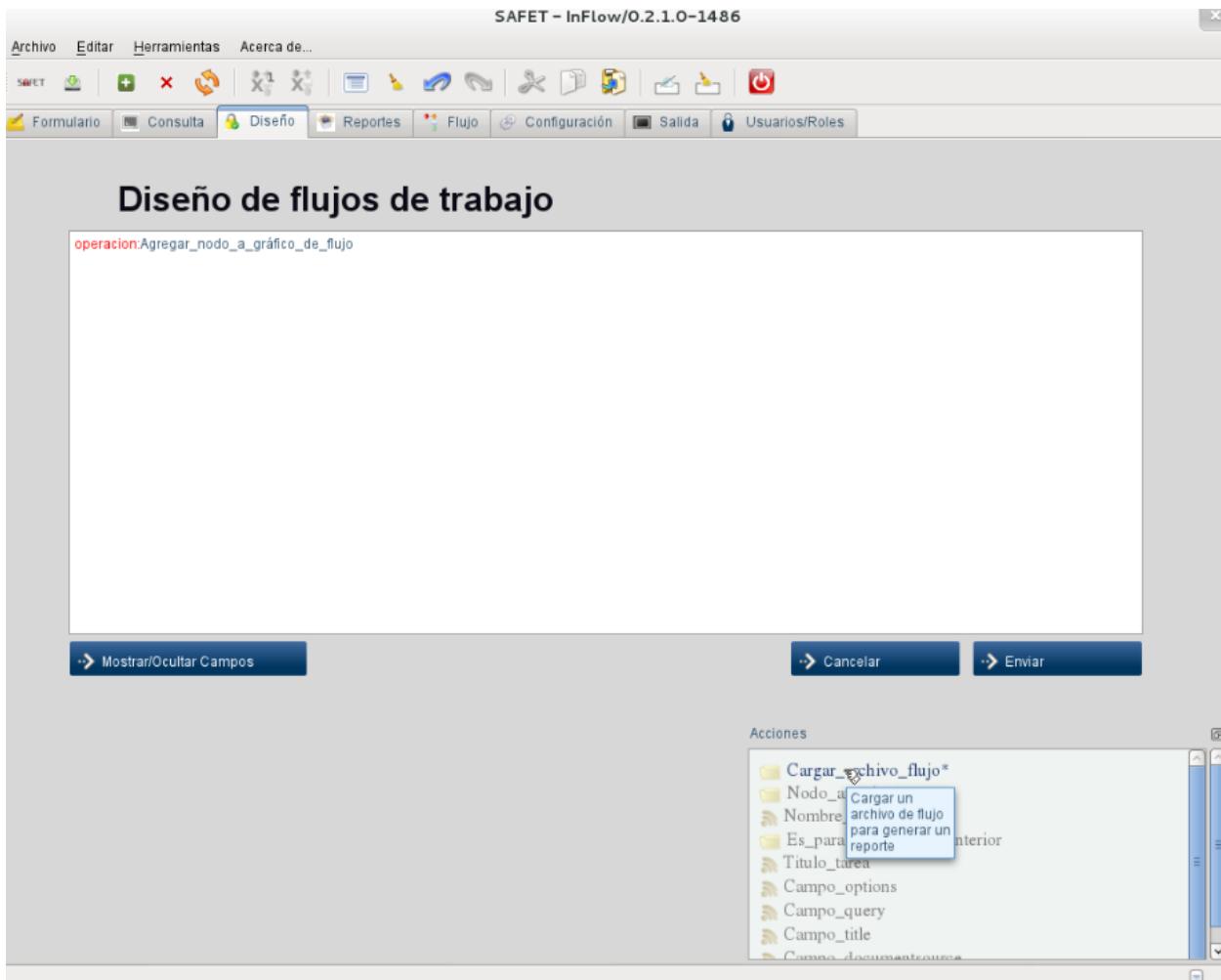


Figura 7.215: **Figura 295: Campo (Cargar\_archivo\_flujo\*)**

### 3° TERCER PASO

- Damos click al botón para buscar el archivo (**Ejemplo1.xml**), como se muestra en la siguiente *Figura 296: Botón buscar*

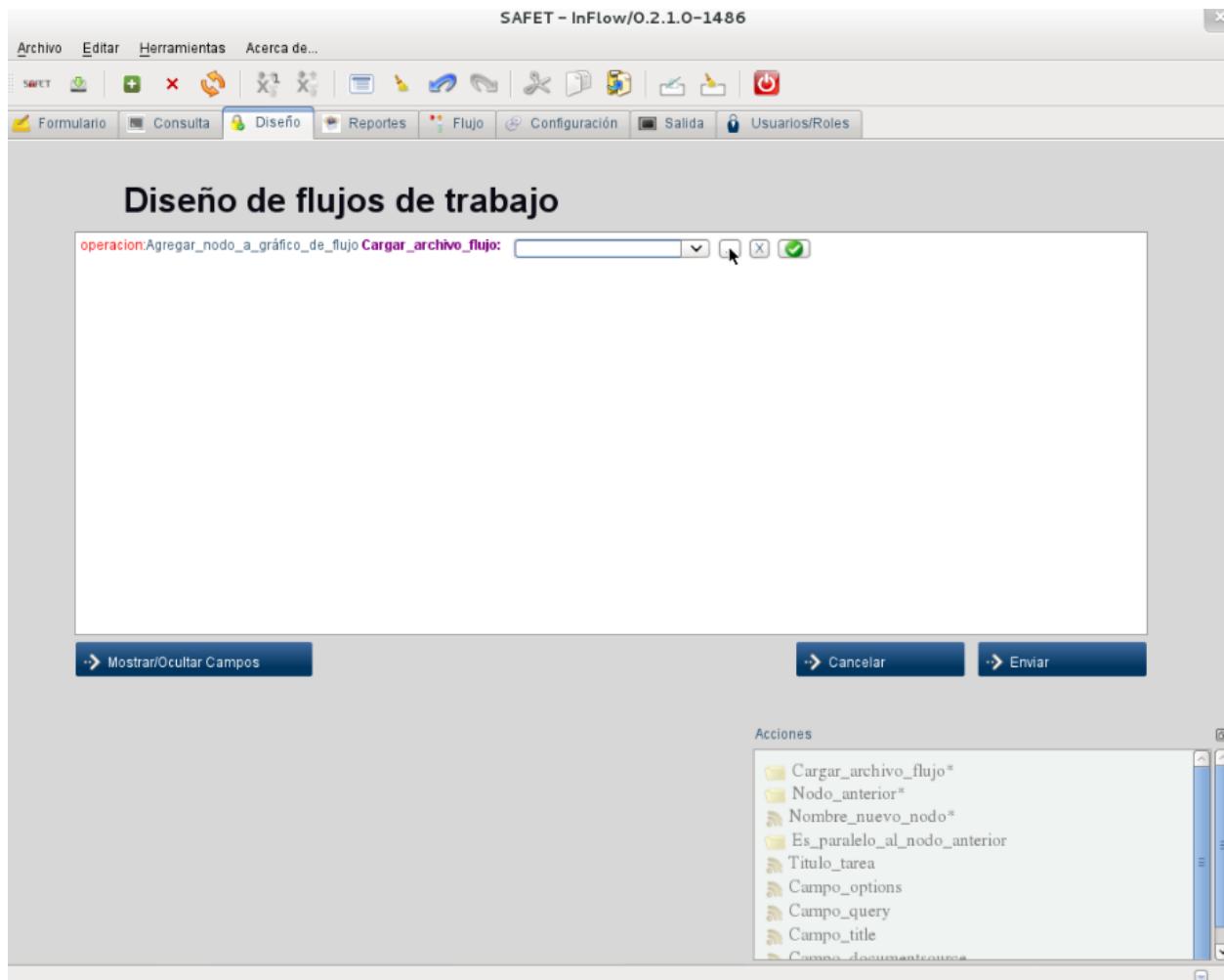


Figura 7.216: **Figura 296: Botón buscar**

### 4° CUARTO PASO

- Seleccionamos el archivo (**Ejemplo1.xml**) del directorio (**/home/usuario/.safet/flowfiles**) y pulsamos en botón (**Abrir**), como se muestra en la siguiente *Figura 297: Seleccionar archivo (.xml)*

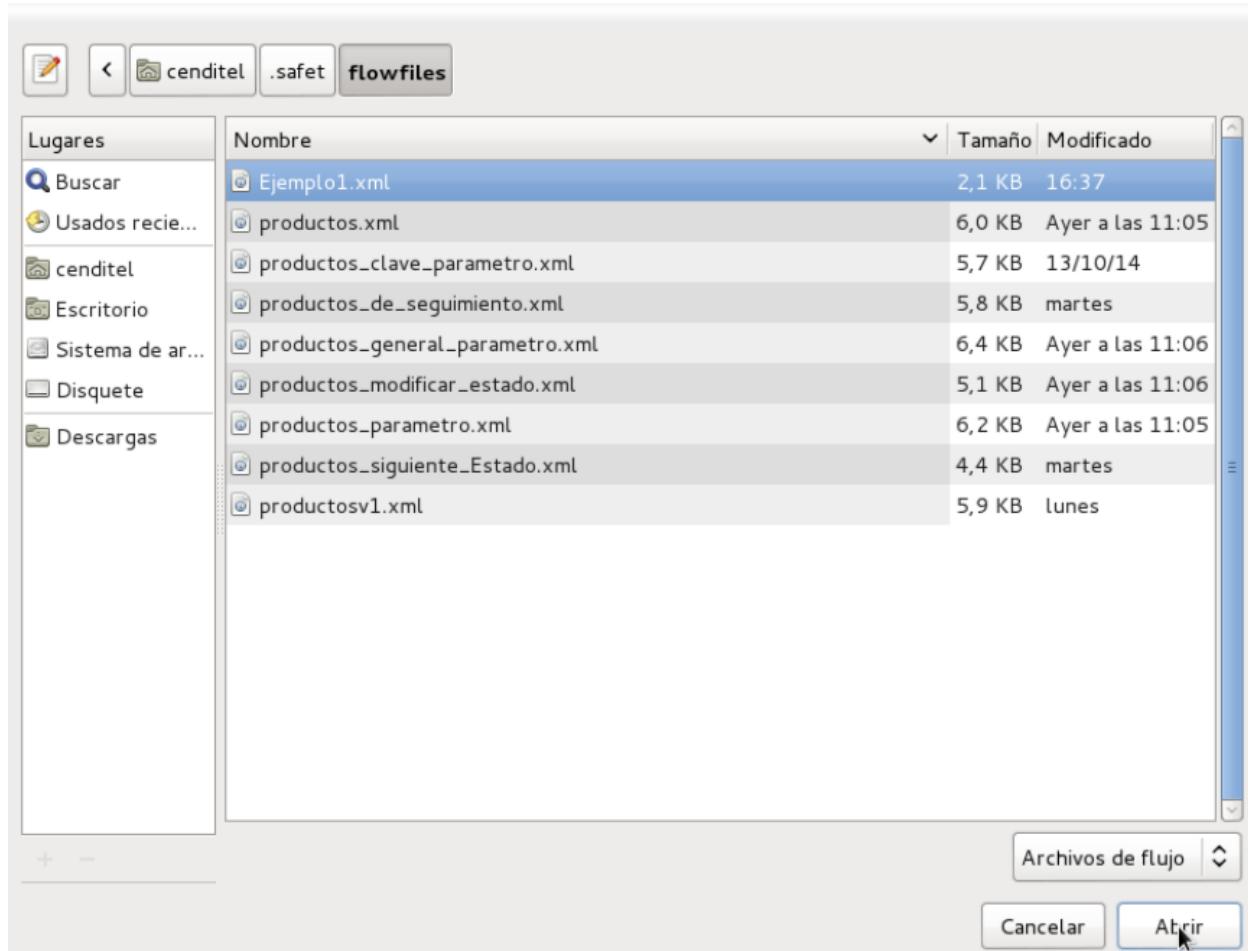


Figura 7.217: **Figura 297: Seleccionar archivo (.xml)**

### 5° QUINTO PASO

- Damos un click al botón con la flecha (**verde**) significando que ya está lleno el campo, como se muestra en la siguiente *Figura 298: Botón (Fin del campo)*

### 6° SEXTO PASO

- Damos click al siguiente campo obligatorio (**Nodo\_anterior\***), como se muestra en la siguiente *Figura 299: Campo (Nodo\_anterior\*)*

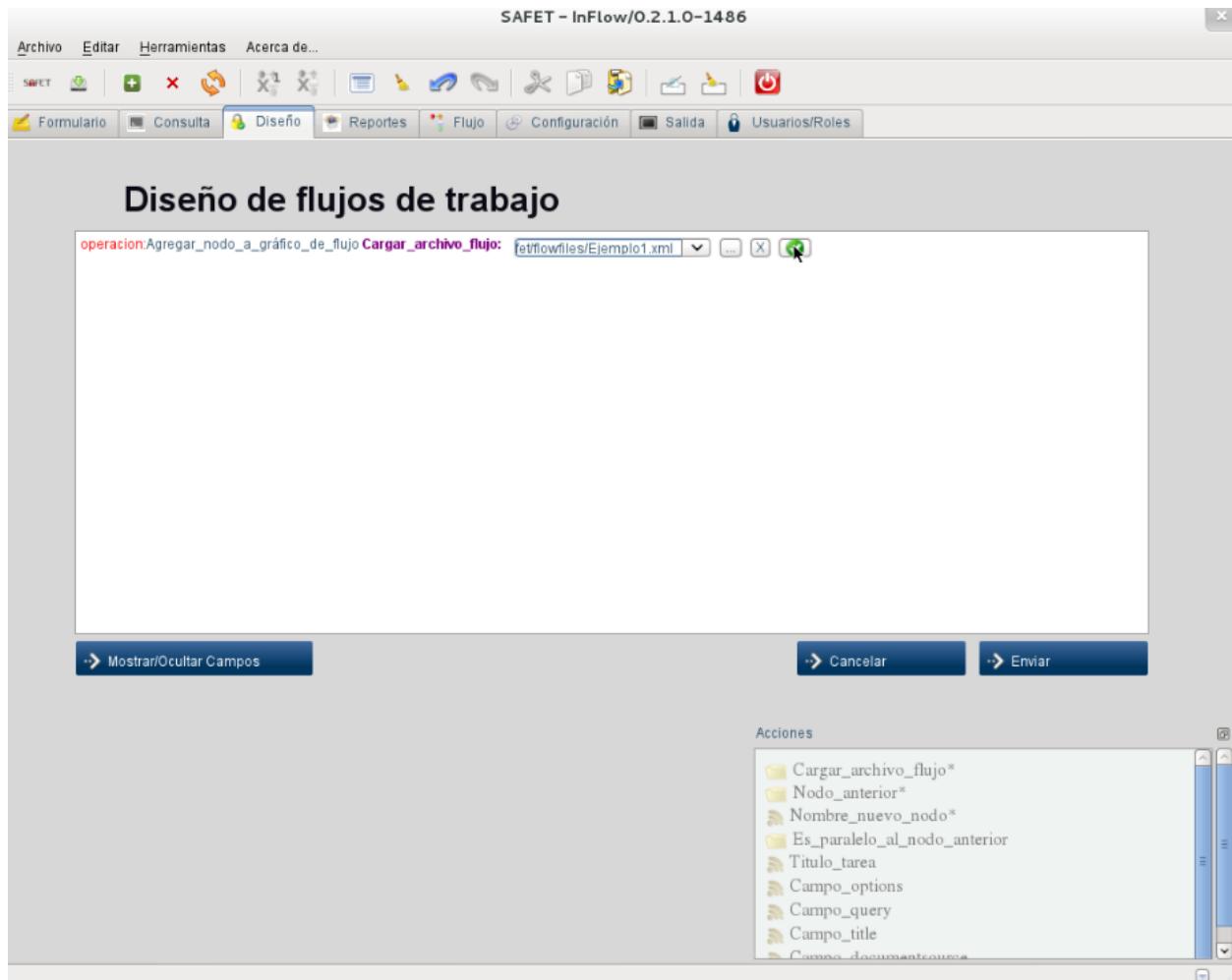


Figura 7.218: Figura 298: Botón (Fin del campo)

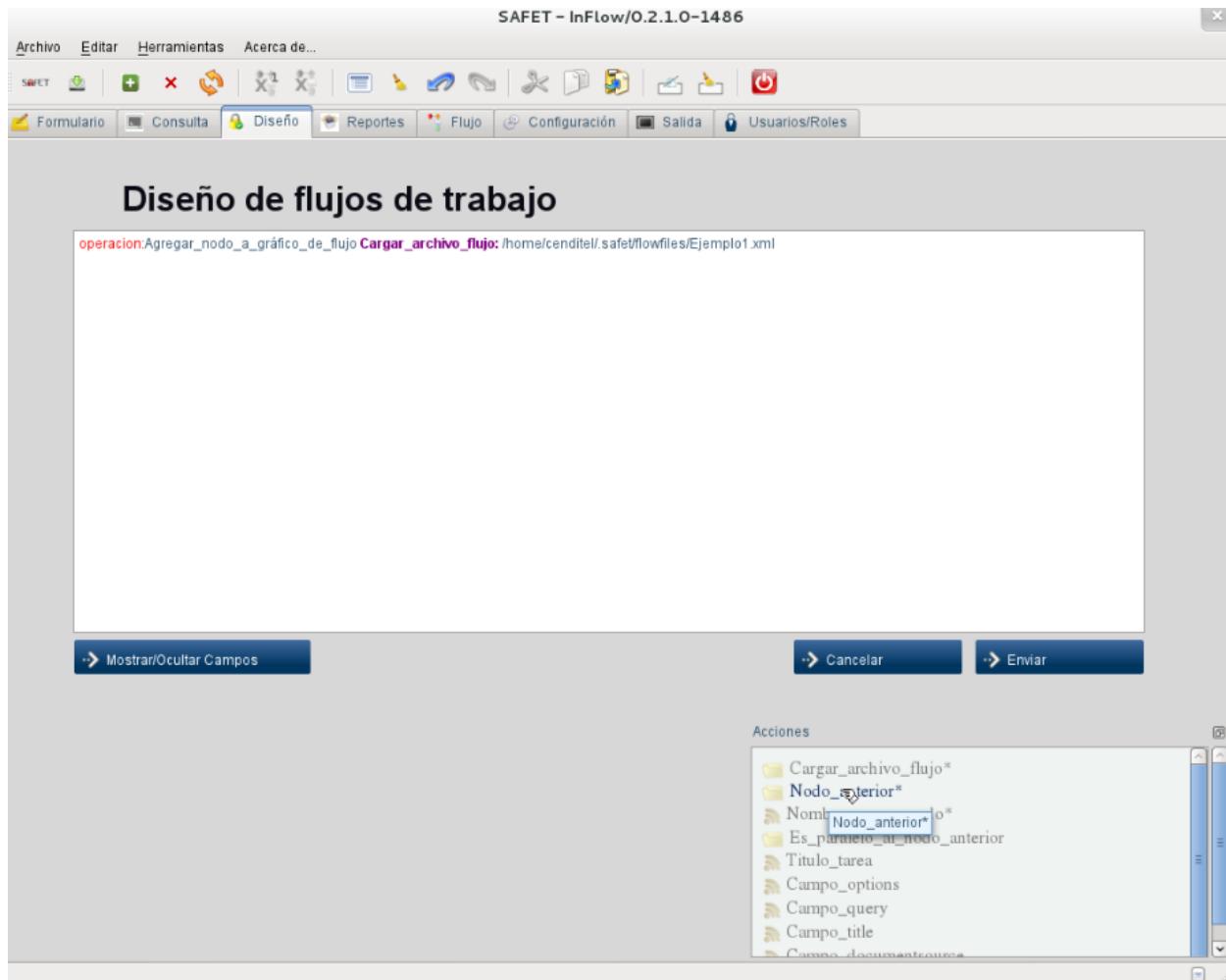


Figura 7.219: Figura 299: Campo (Nodo\_anterior\*)

## 7° SEPTIMO PASO

- Seleccionamos el nodo anterior (**cantidad**), como se muestra en la siguiente *Figura 300: Selección de nodo*

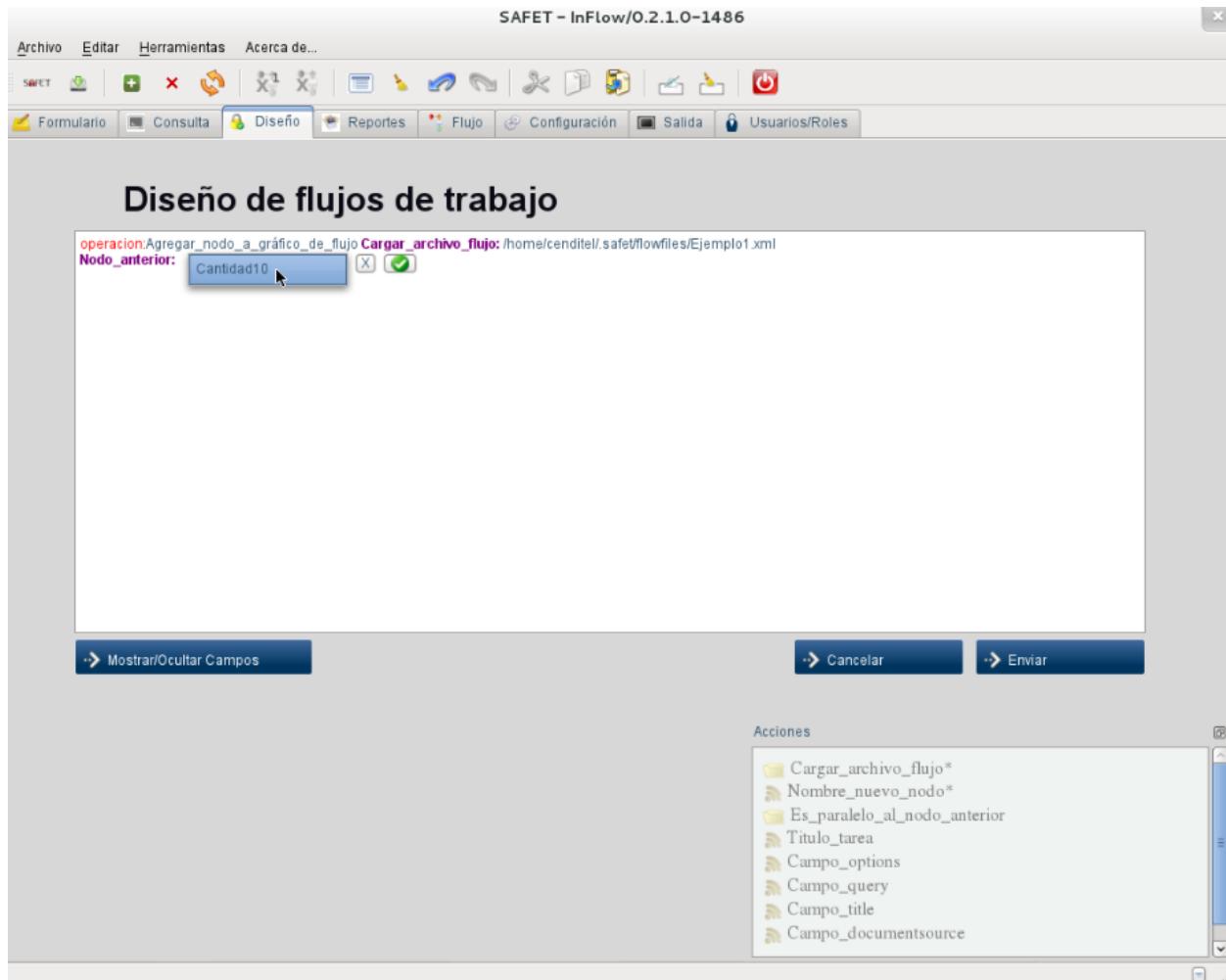


Figura 7.220: **Figura 300: Selección de nodo**

## 8° OCTAVO PASO

- Damos un click al **botón con la flecha (verde)** significando que ya está lleno el campo, como se muestra en la siguiente *Figura 301: Botón (Fin del campo)*

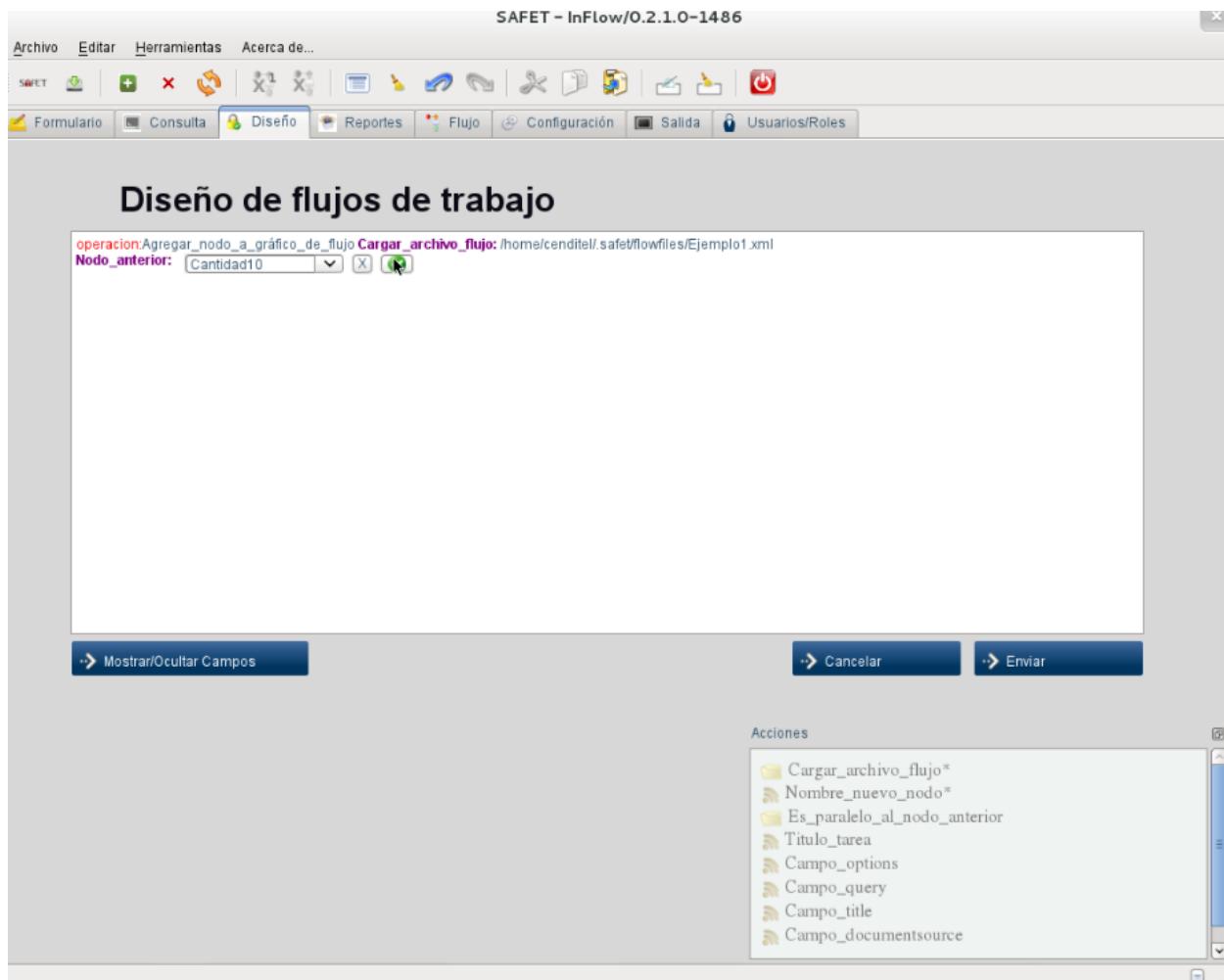


Figura 7.221: Figura 301: Botón (Fin del campo)

## 9° NOVENO PASO

- Damos click al siguiente campo obligatorio (**Nombre\_nuevo\_nodo\***), como se muestra en la siguiente *Figura 302: Campo (Nombre\_nuevo\_nodo\*)*

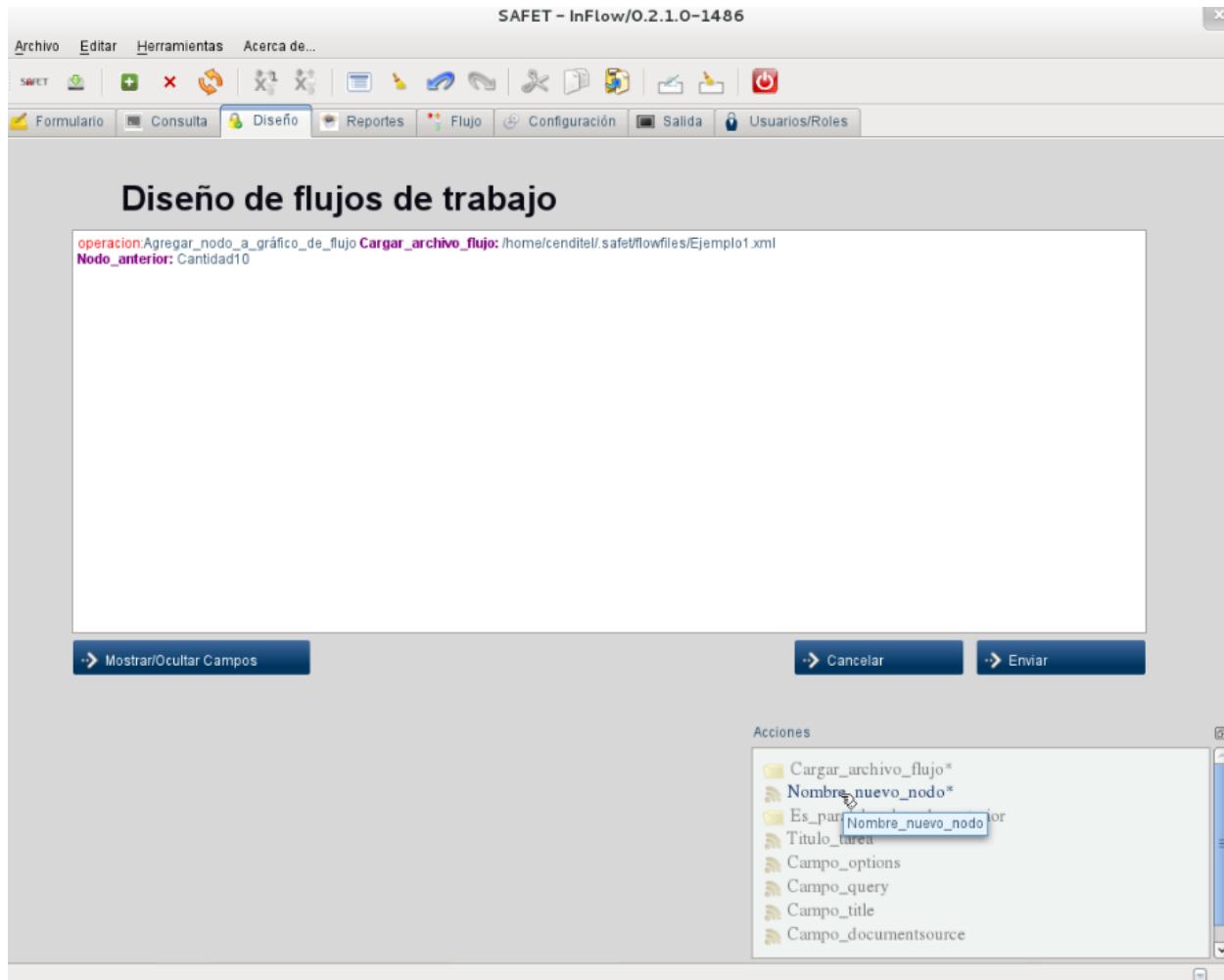


Figura 7.222: **Figura 302: Campo (Nombre\_nuevo\_nodo\*)**

## 10° DECIMO PASO

- Agregamos el nombre del **nodo**, por ejemplo (**Cantidad40**) y pulsamos en el **botón** con la flecha **(verde)** significando que ya está lleno el campo, como se muestra en la siguiente *Figura 303: Nombre del nodo*

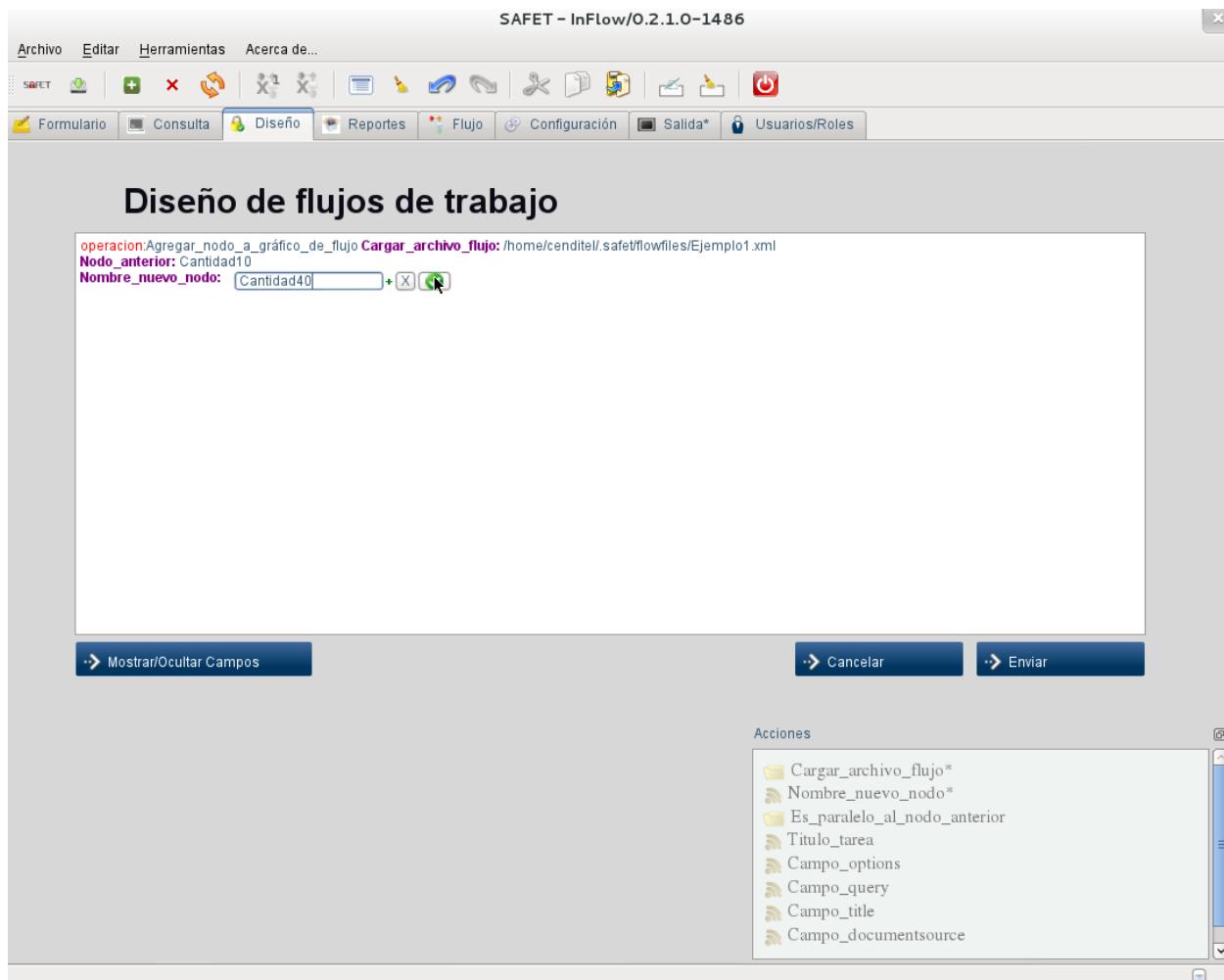


Figura 7.223: **Figura 303: Nombre del nodo**

## 11° DECIMO PRIMERO PASO

- Damos click al siguiente campo opcional (**Titulo\_tarea**), como se muestra en la siguiente *Figura 304: Campo (Titulo\_tarea)*

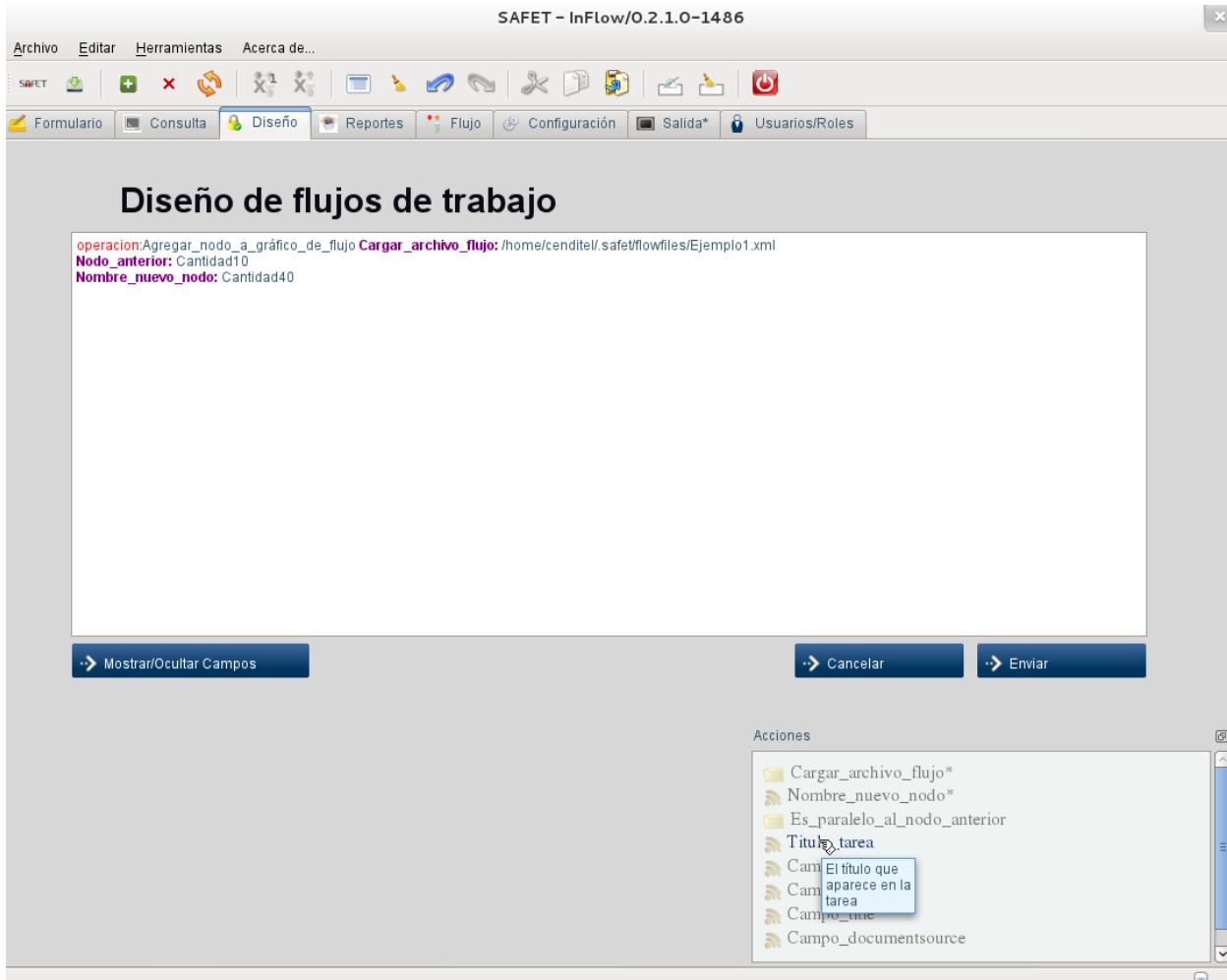


Figura 7.224: **Figura 304: Campo (Titulo\_tarea)**

## 12° DECIMO SEGUNDO PASO

- Agregamos el titulo de la **tarea**, por ejemplo (**Tarea2**) y pulsamos en el **botón con la flecha (verde)** significando que ya está lleno el campo, como se muestra en la siguiente *Figura 305: Titulo de la tarea*

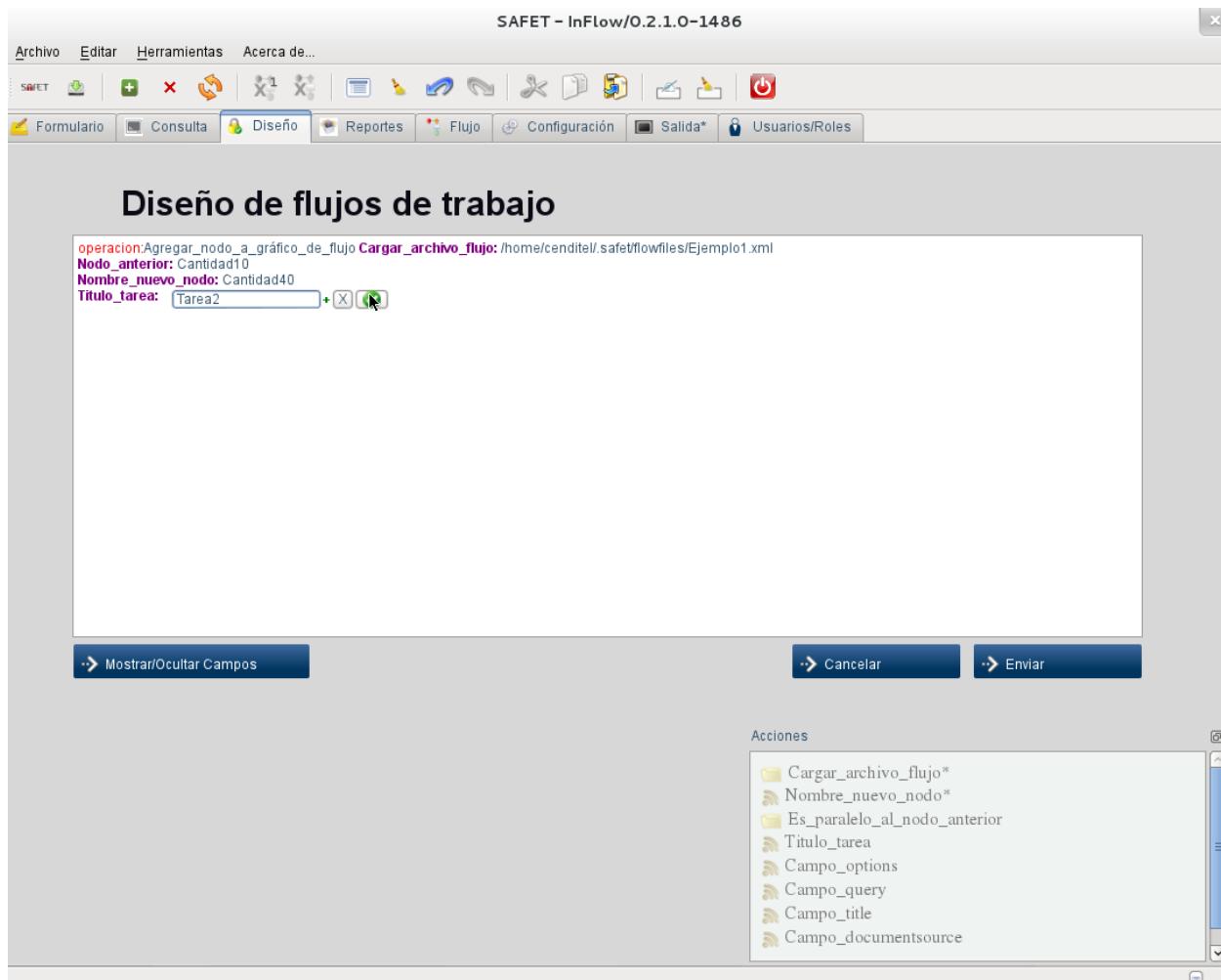


Figura 7.225: **Figura 305: Título de la tarea**

### 13° DECIMO TERCER PASO

- Damos click al siguiente campo opcional (**Campo\_options**), como se muestra en la siguiente *Figura 306: Campo (Campo\_options)*

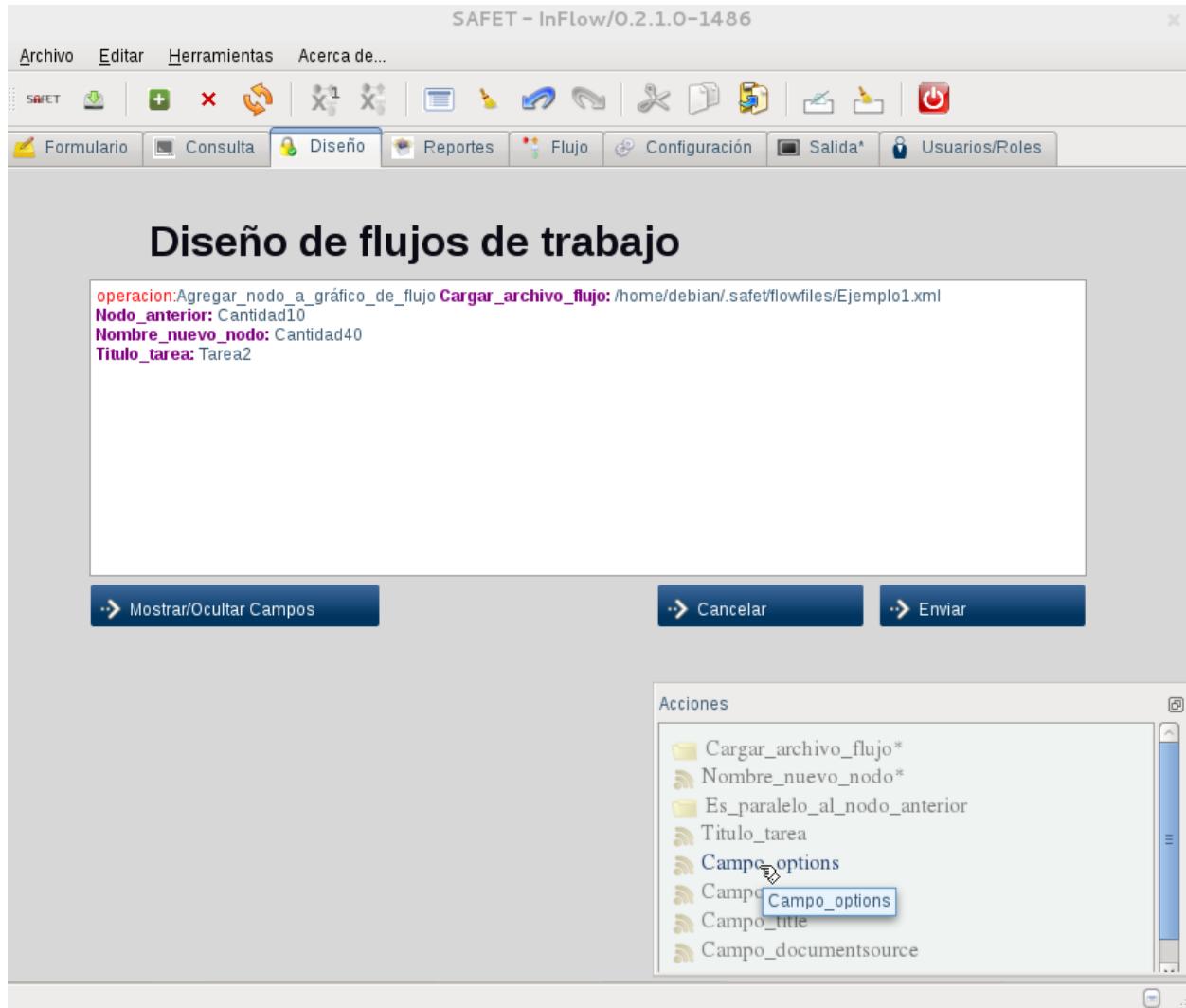


Figura 7.226: **Figura 306: Campo (Campo\_options)**

### 14° DECIMO CUARTO PASO

- Agregamos la opción de búsqueda del datos, por ejemplo, que muestre cuantos productos hay mayor a o igual a 40, colocamos la opción ( $>=40$ ) y pulsamos el botón con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 307: Opción (>=10)*

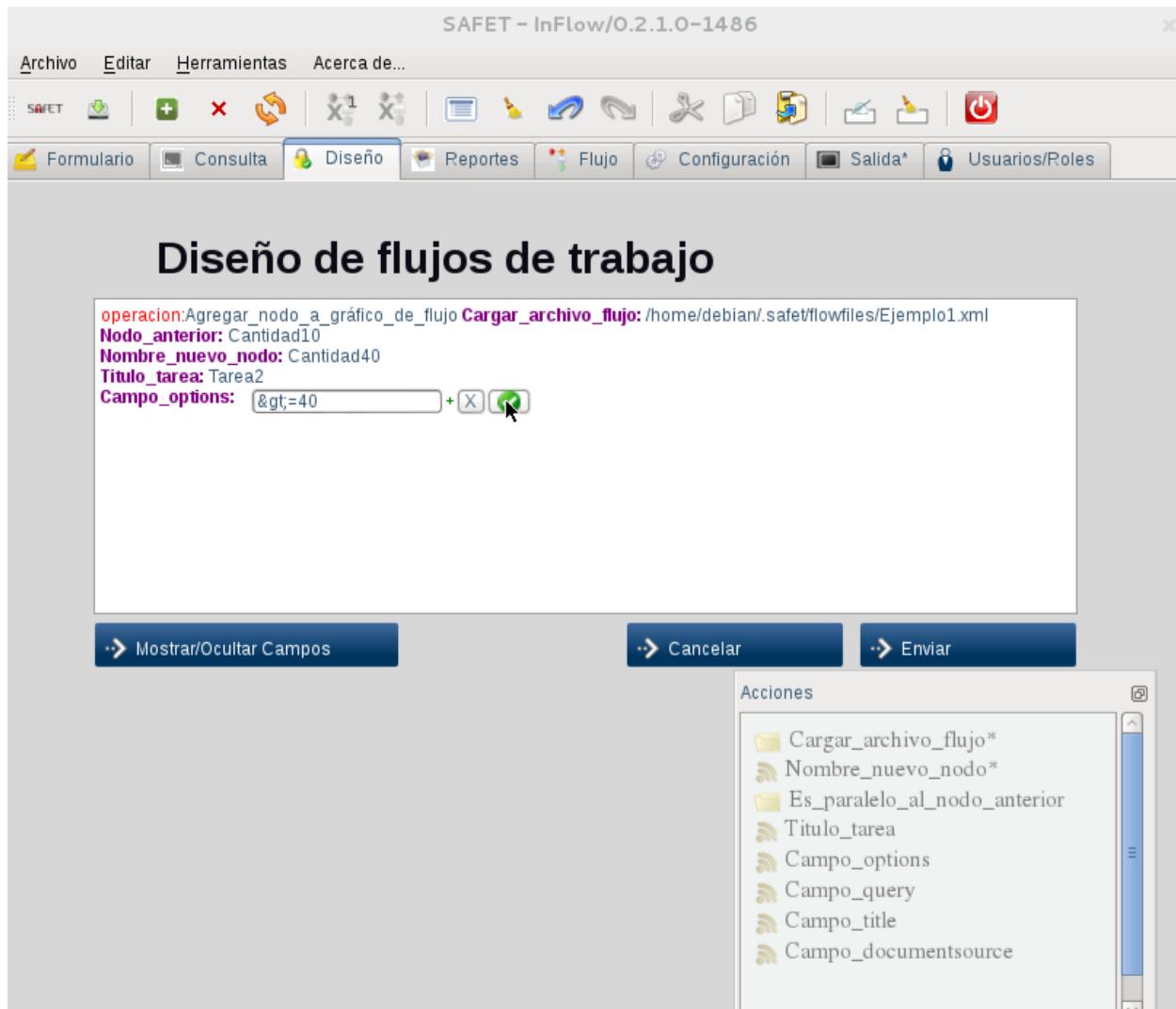


Figura 7.227: Figura 307: Opción (&gt;=10)

## 15° DECIMO QUINTO PASO

- Damos click al siguiente campo opcional (**Campo\_query**), como se muestra en la siguiente *Figura 308: Campo (Campo\_query)*

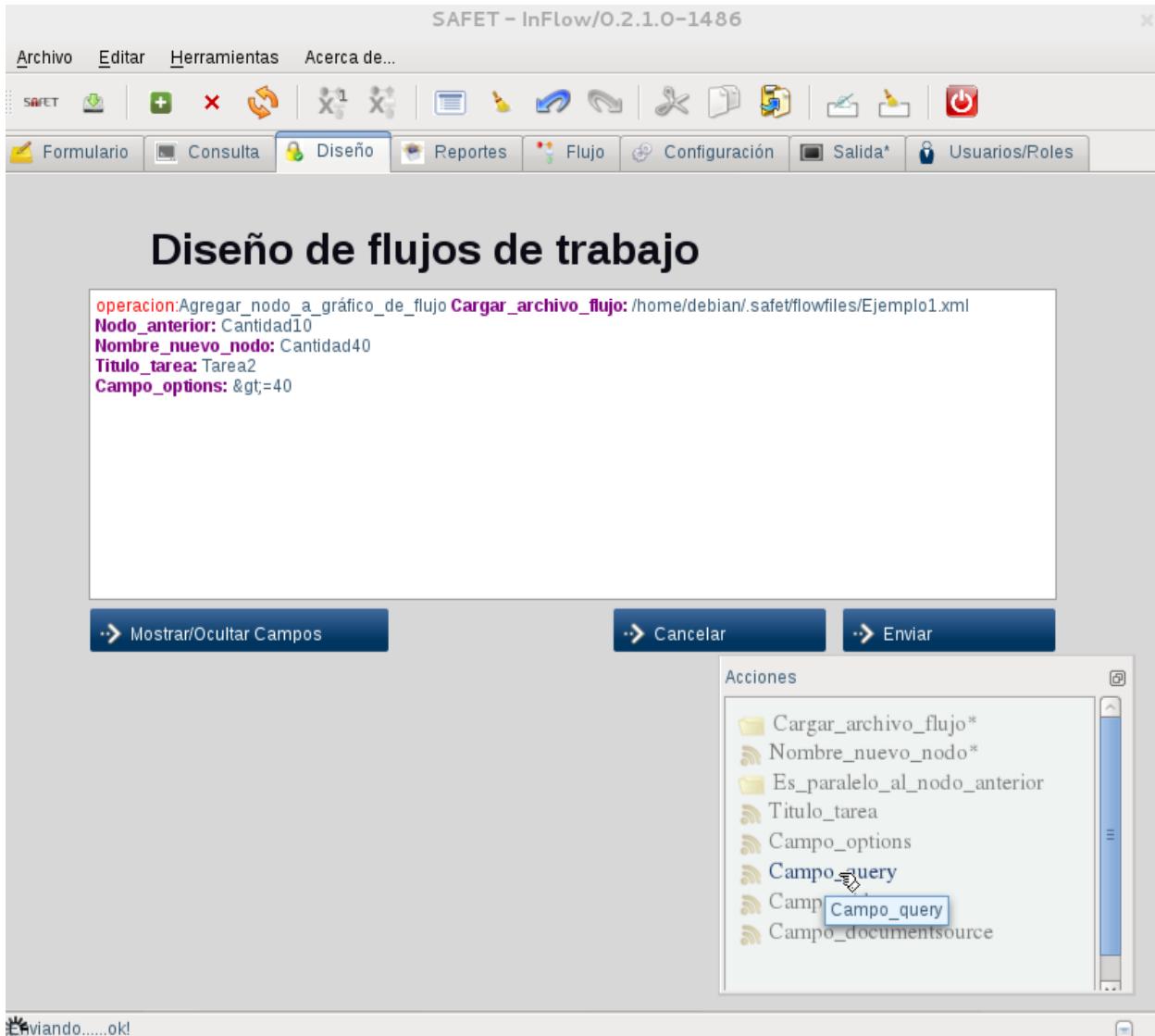


Figura 7.228: Figura 308: Campo (Campo\_query)

## 16° DECIMO SEXTO PASO

- Agregamos la consulta **sql** donde (**select cantidad from productos**), como se muestra en la siguiente *Figura 309: Consulta (sql)*

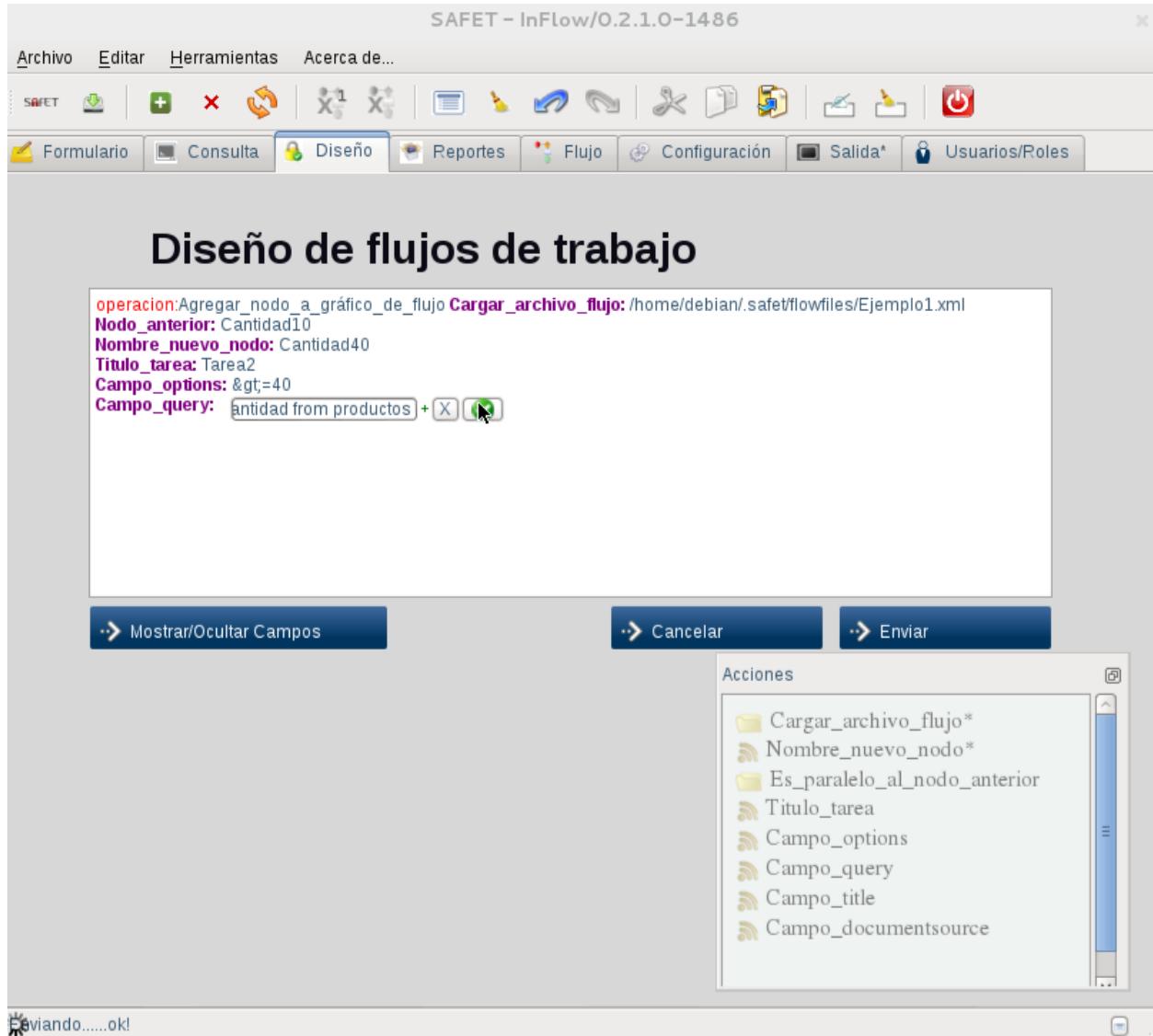


Figura 7.229: **Figura 309: Consulta (sql)**

## 17° DECIMO SEPTIMO PASO

- Damos un click al botón (**Enviar**) para finalizar con la operación la cual se nos mostrara como resultado (**Agregado nodo “Cantidad”** a “**/home/debian/.safet/flowfiles/**”), como se muestra en la siguiente *Figura 310: Fin de la operación*

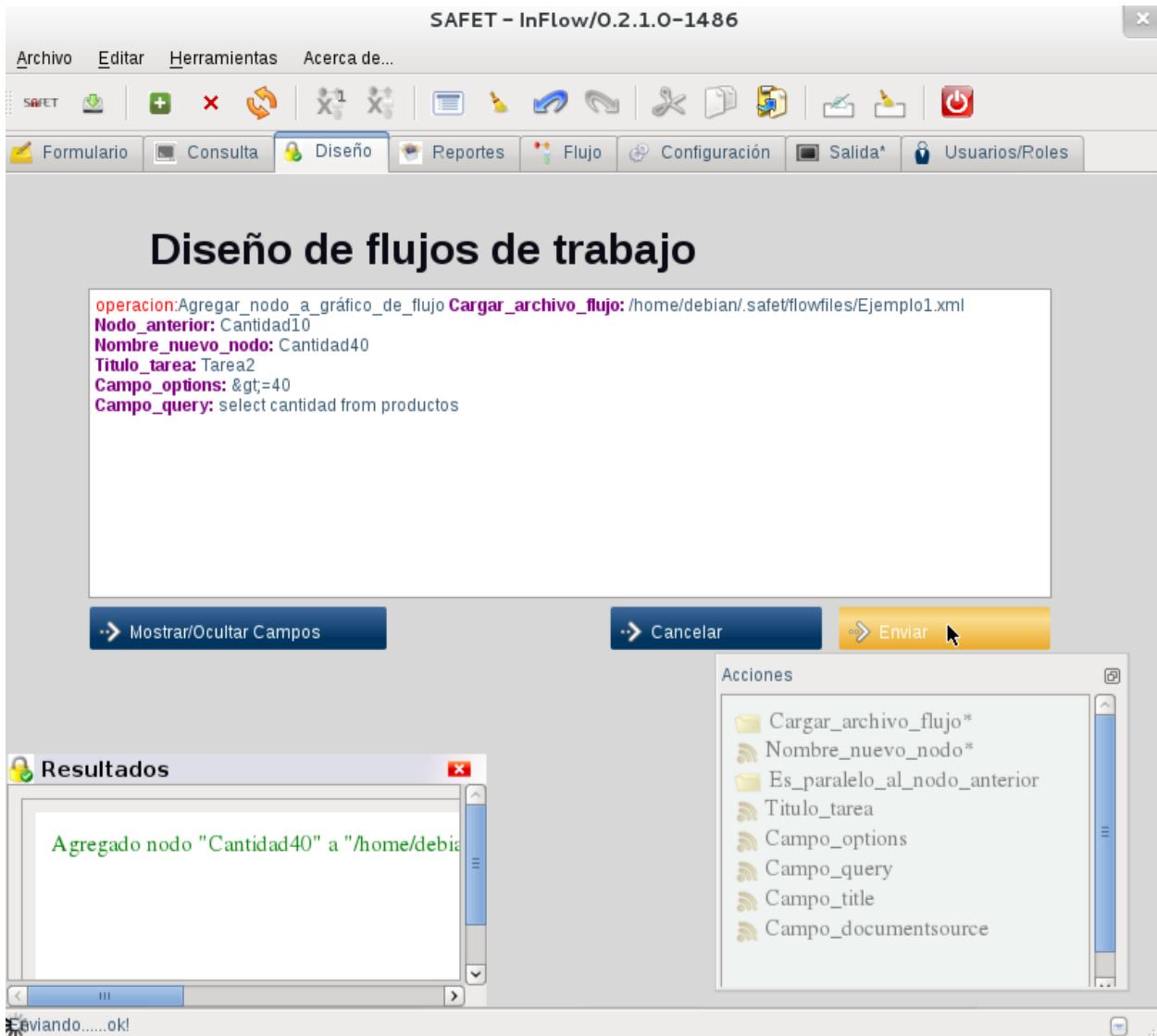


Figura 7.230: **Figura 310: Fin de la operación**

### 7.10.3 C.- Observemos el nodo agregado en flujo de trabajo

Para observar la gráfica del archivo (**Ejemplo1.xml**) del directorio (**/home/usuario/.safet/flowfiles/**), para ello damos click al siguiente link. *B.- PRIMERA OPERACIÓN (Gráfico coloreado general)*

**Nota:** Se nos mostrara el reporte del gráfico agregado el nodo de la siguiente manera

Figura 311: Gráfico

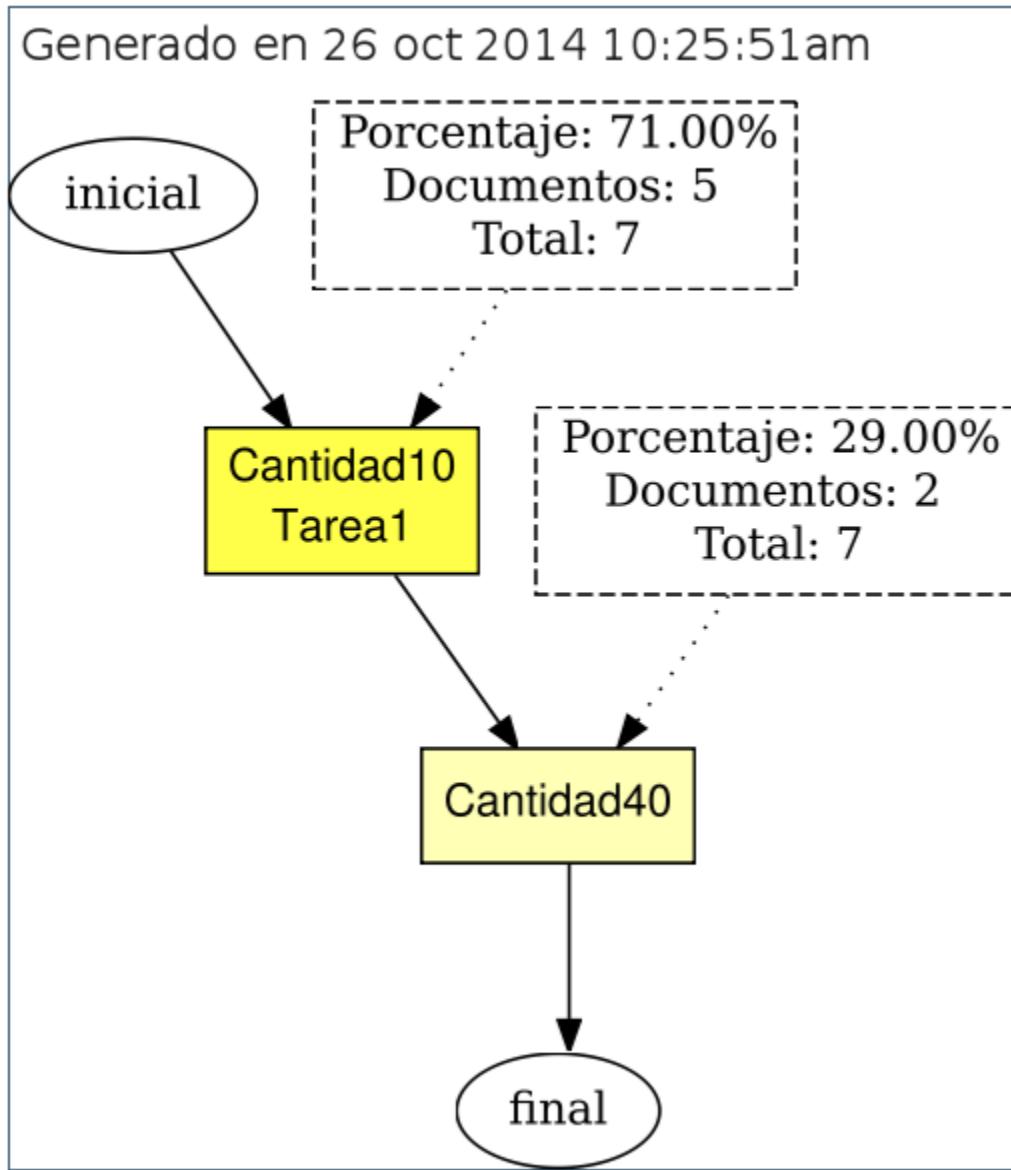


Figura 7.231: Figura 311: Gráfico

---

## 7.11 Diseñar el cambio de conexiones de nodos enlazados

En el gráfico coloreado se nos muestra la siguiente *Figura 312: Gráfico general*, en cual vamos a cambiar de posición al nodo (**Por\_llegar**) para ello seguimos los siguientes paso:

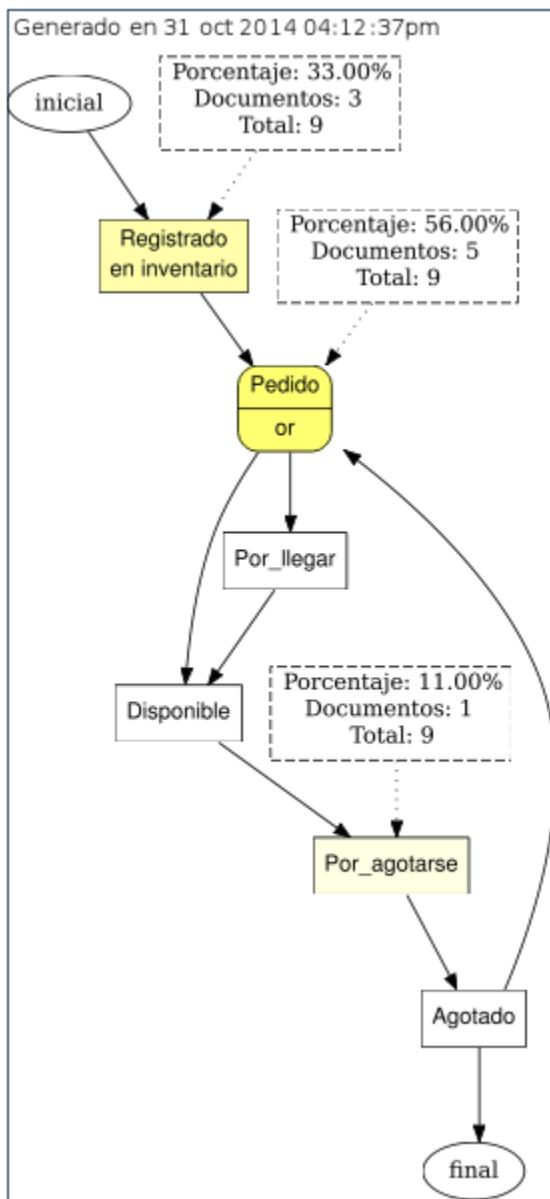


Figura 7.232: Figura 312: Gráfico general

### 7.11.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 312: Usuario/password*



Figura 7.233: **Figura 312: Usuario/password**

#### 3° TERCER PASO

- Damos click a la segunda opción (**Frimar/verificar documento**), como se muestra en la siguiente *Figura 313: Opcion Diseño*

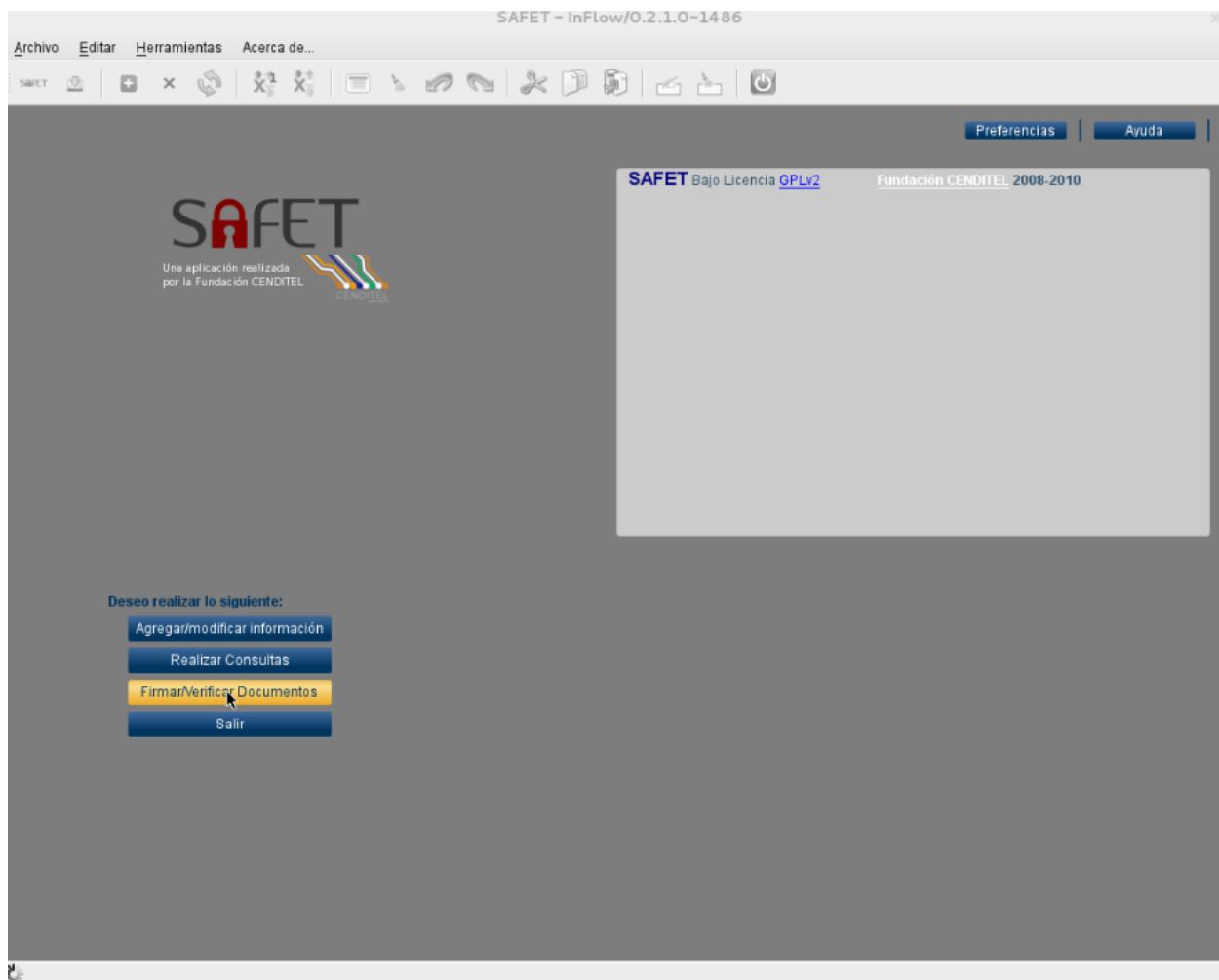


Figura 7.234: **Figura 313: Opcion Diseño**

### 4° CUARTO PASO

- Damos click a la segunda opción (**Mostar/Ocultar Campos**), como se muestra en la siguiente *Figura 314: Campos del Diseño*

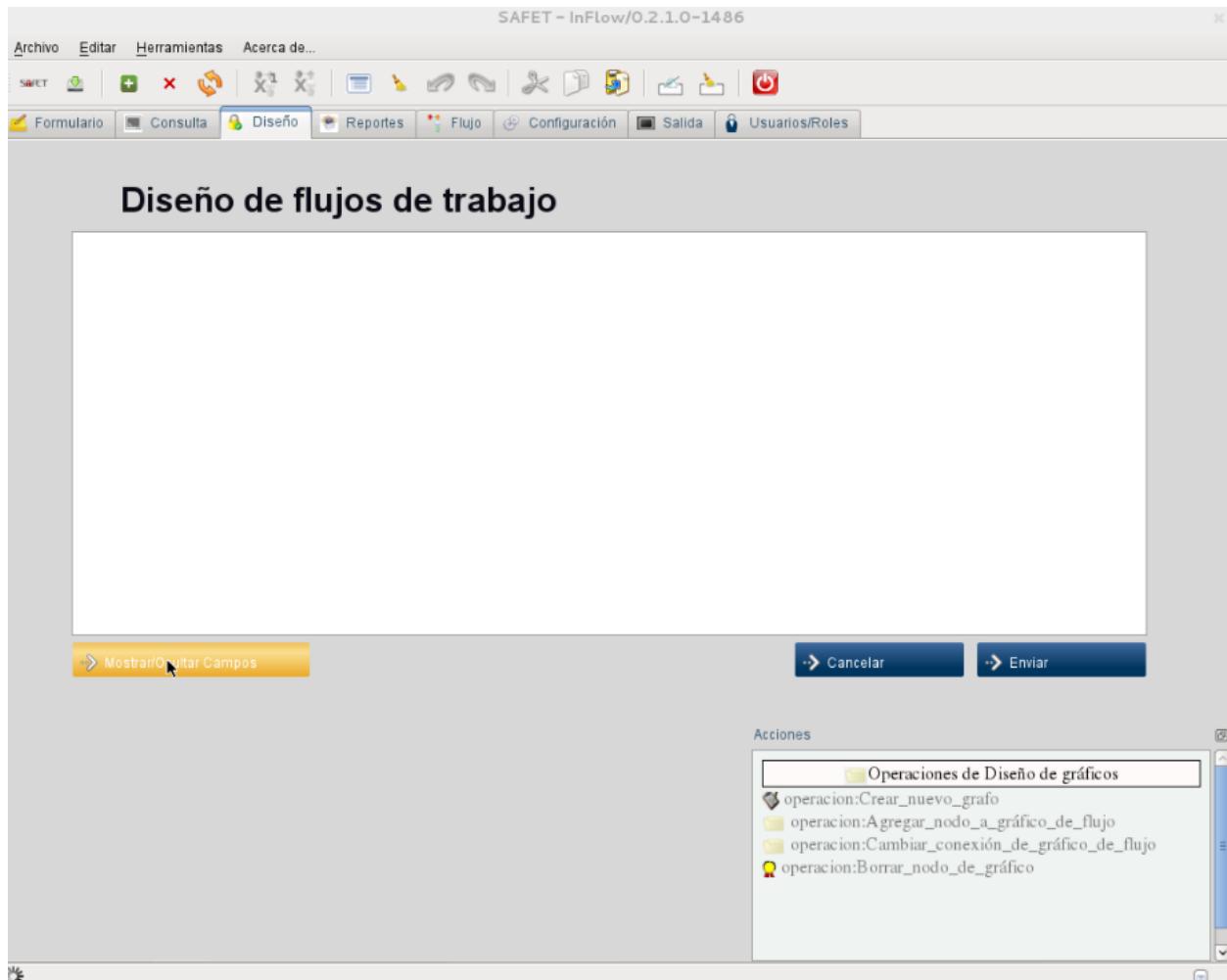


Figura 7.235: **Figura 314: Campos del Diseño**

### 7.11.2 B.- Cambiar conexiones de nodos

#### 1° PRIMER PASO

- Damos click a la operación (**Cambiar\_conexión\_de\_gráfico\_de\_flujo**), como se muestra en la siguiente *Figura 315: Cambiar\_conexión\_de\_gráfico\_de\_flujo*

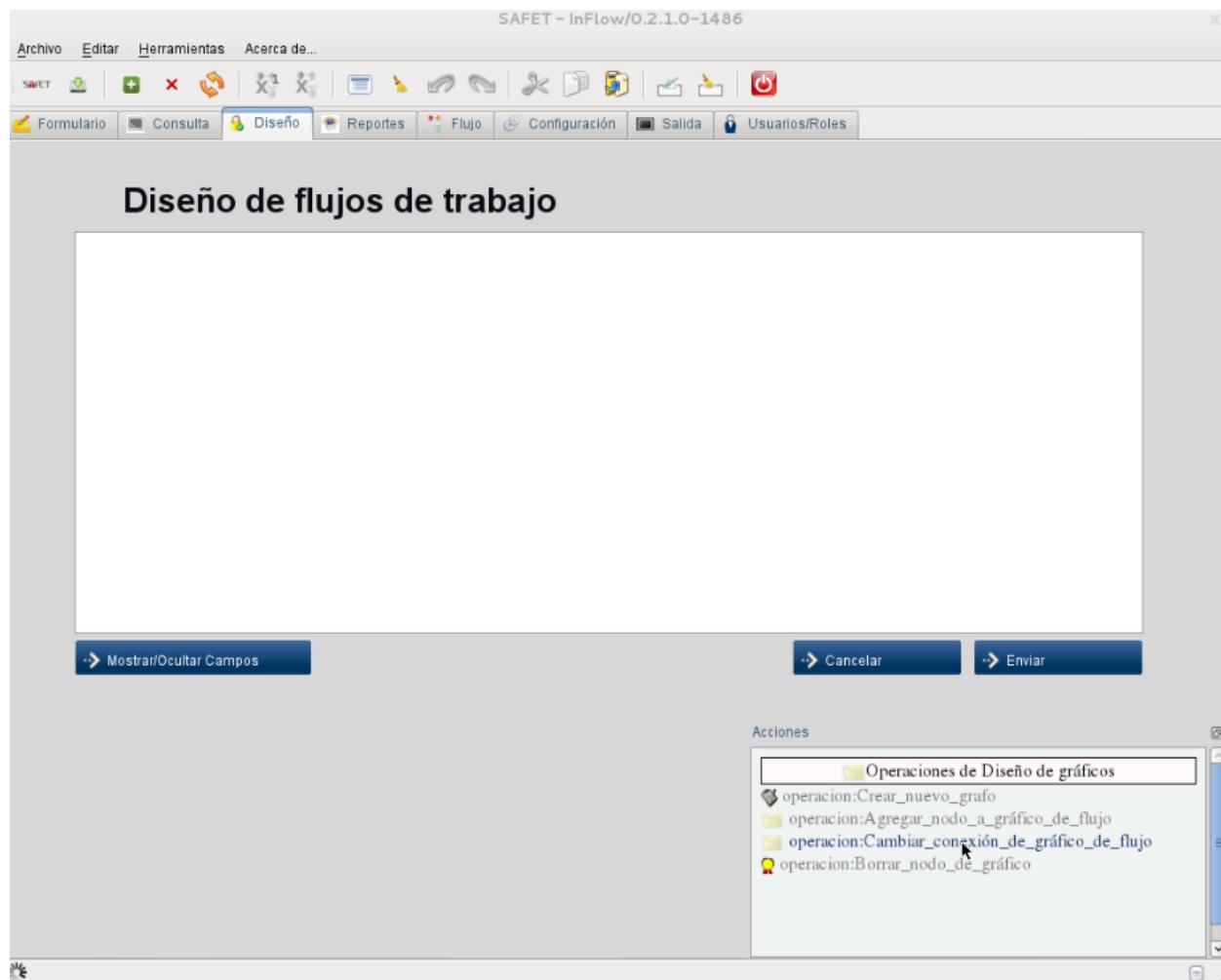


Figura 7.236: Figura 315: Cambiar\_conexión\_de\_gráfico\_de\_flujo

## 2° SEGUNDO PASO

- Damos click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 316: Campo (Cargar\_archivo\_flujo\*)*

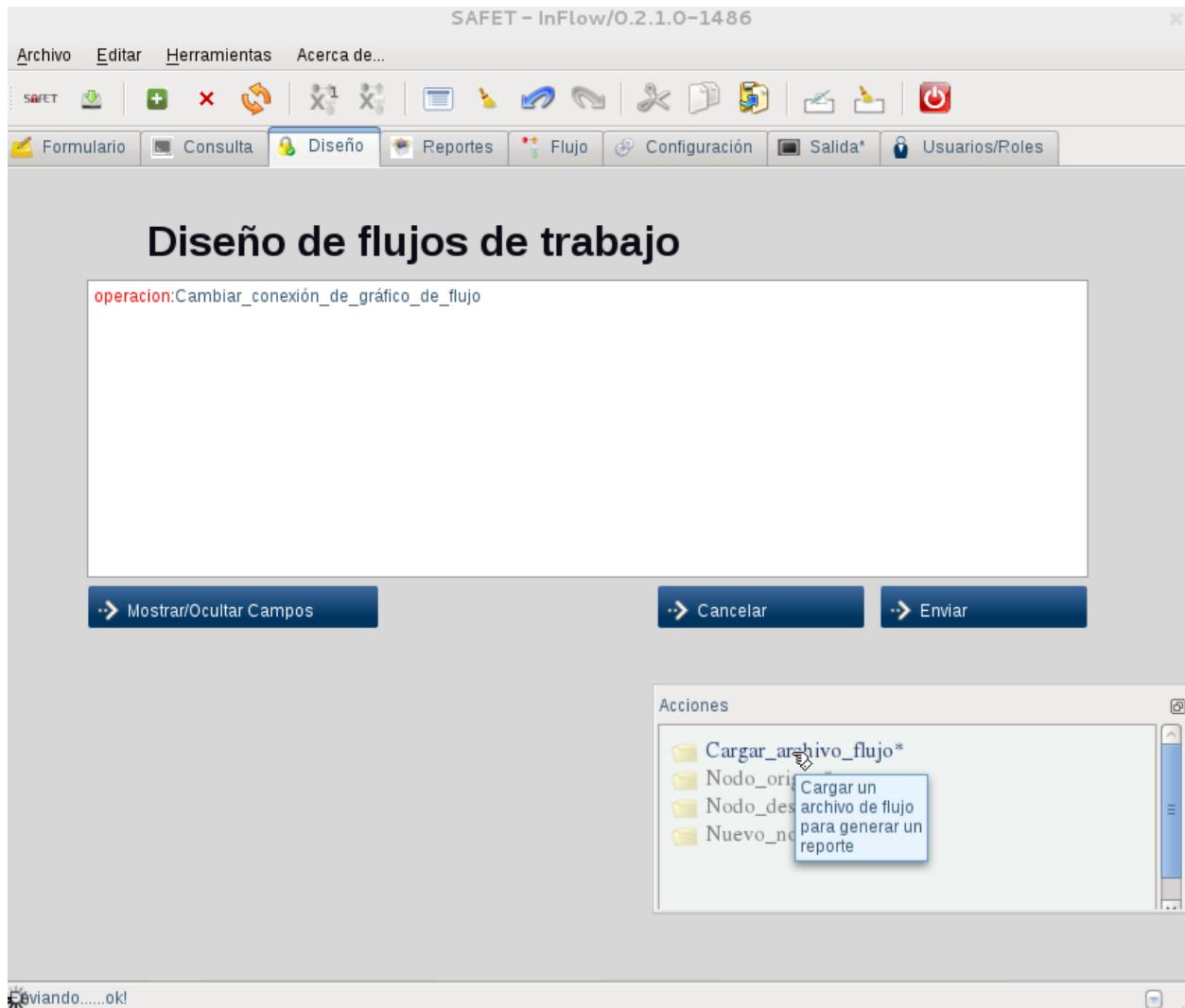


Figura 7.237: **Figura 316: Campo (Cargar\_archivo\_flujo\*)**

## 3° TERCER PASO

- Damos click al botón para buscar el archivo (**productos.xml**), como se muestra en la siguiente *Figura 317: Botón buscar*

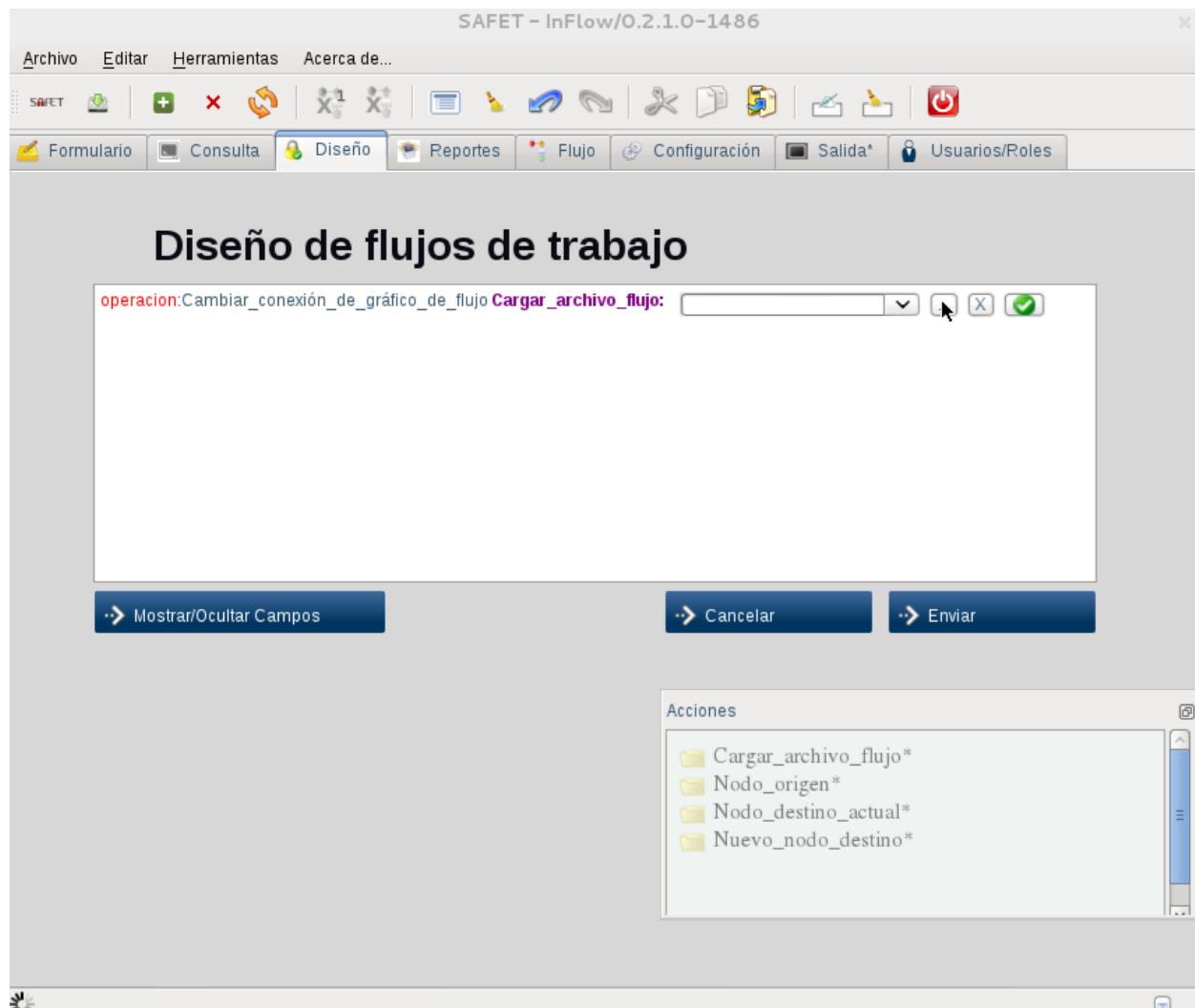


Figura 7.238: **Figura 317: Botón buscar**

### 4° CUARTO PASO

- Seleccionamos el archivo (**productos.xml**) del directorio (**/home/usuario/.safet/flowfiles**) y pulsamos en botón (**Abrir**), como se muestra en la siguiente *Figura 318: Seleccionar archivo (XML)*

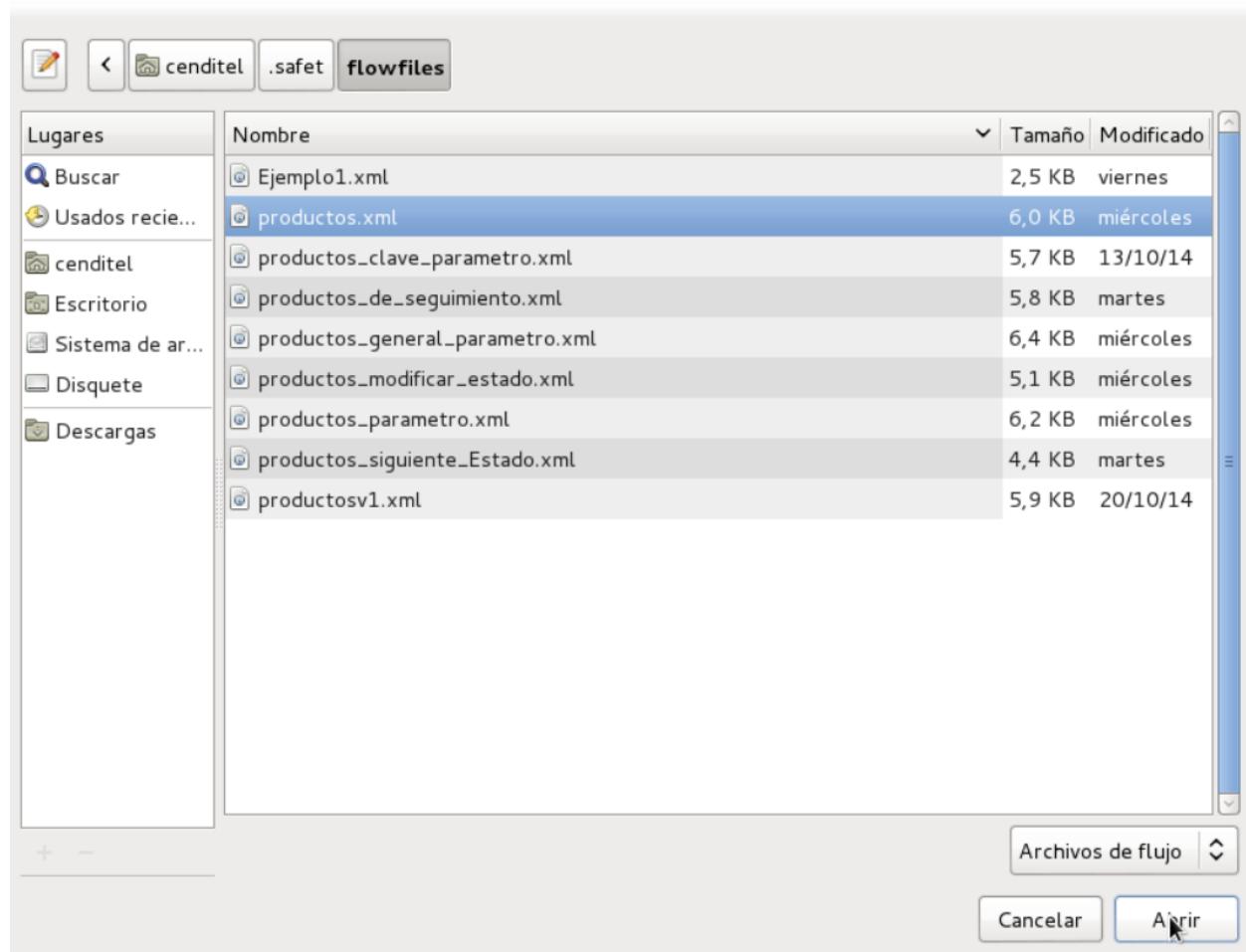


Figura 7.239: **Figura 318: Seleccionar archivo (XML)**

### 5° QUINTO PASO

- Damos un click al botón con la flecha (**verde**) significando que ya está lleno el campo, como se muestra en la siguiente *Figura 319: Botón (Fin de campo)*

### 6° SEXTO PASO

- Damos click al siguiente campo obligatorio (**Nodo\_origen\***), como se muestra en la siguiente *Figura 320: Campo (Nodo\_origen\*)*

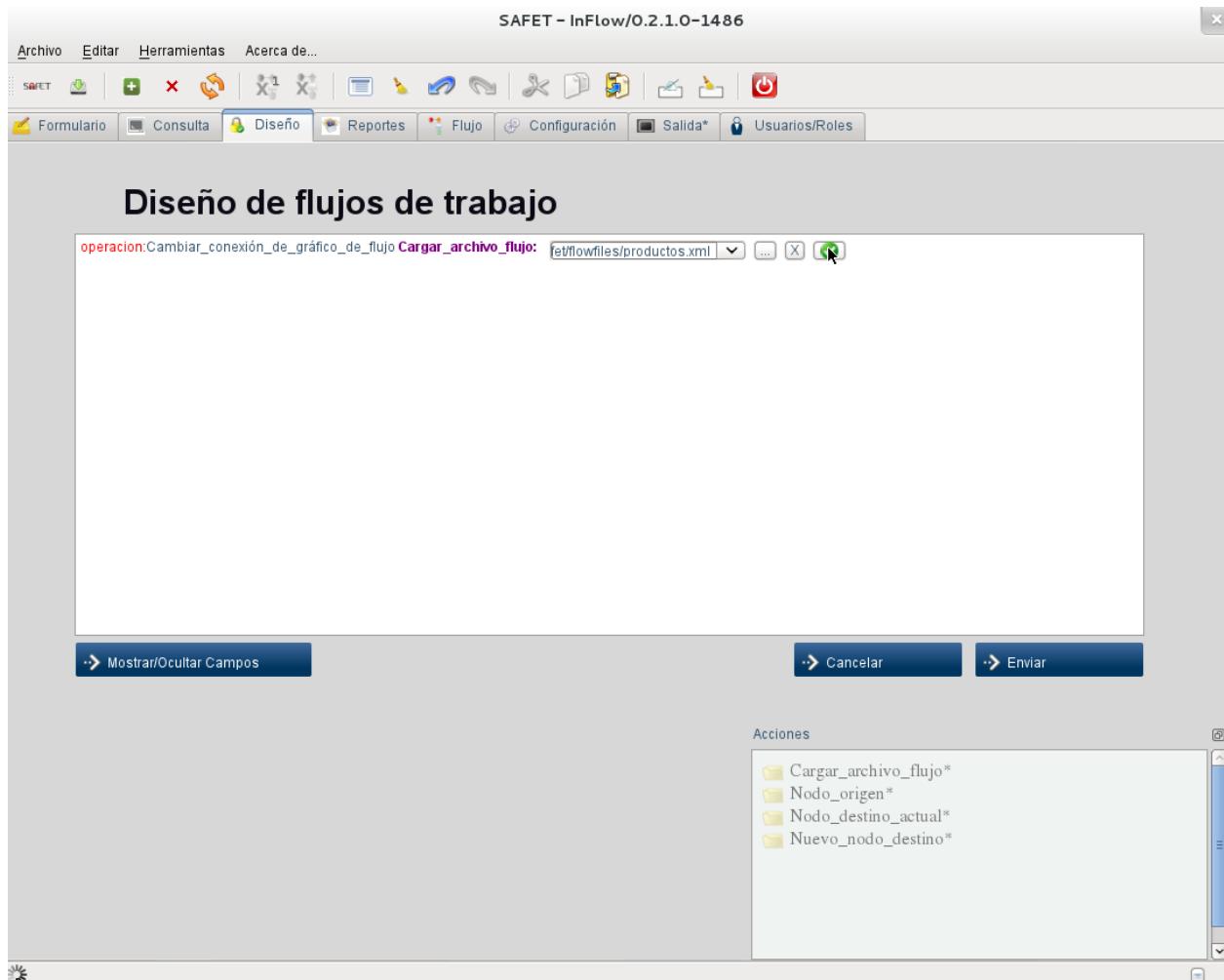


Figura 7.240: Figura 319: Botón (Fin de campo)

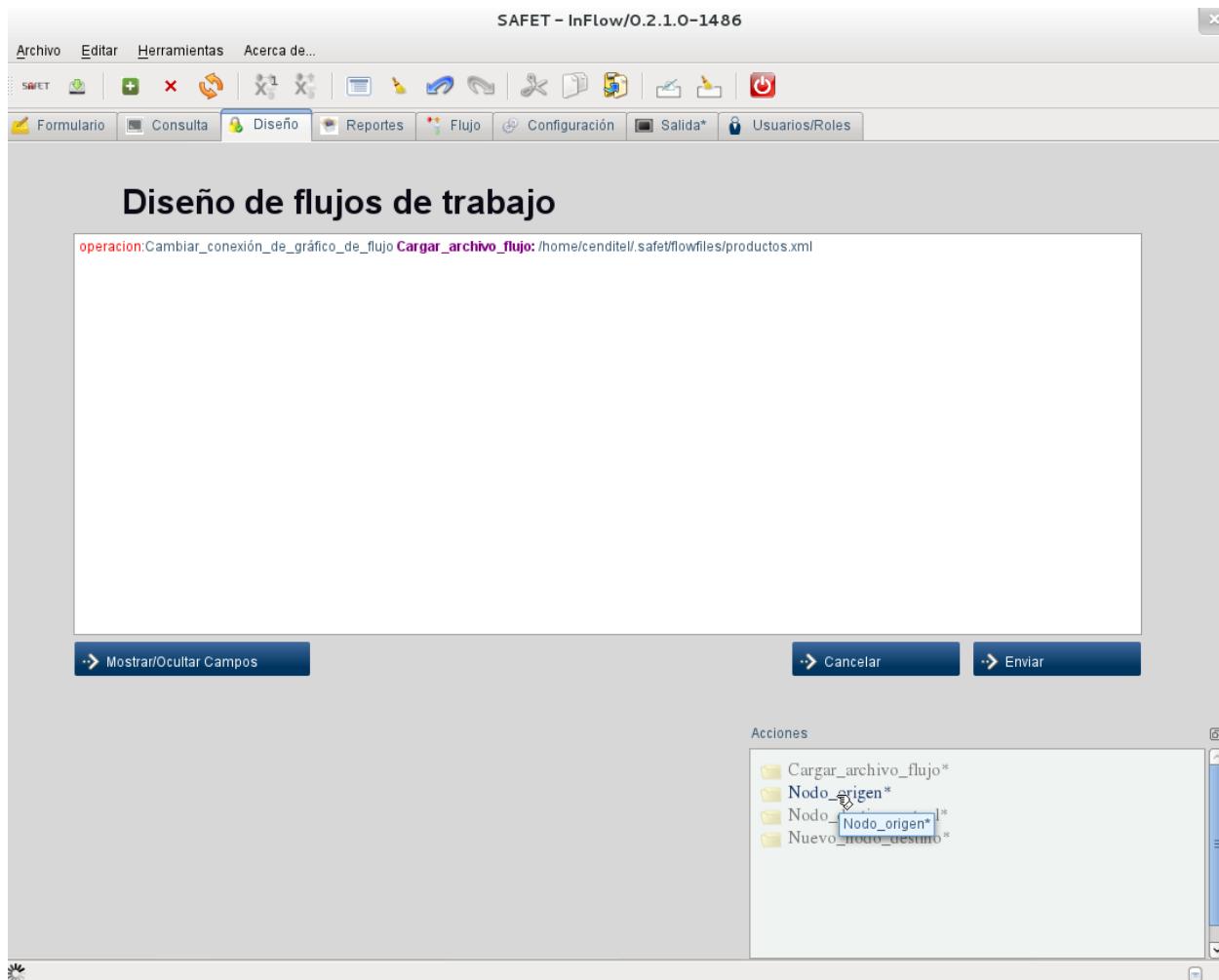


Figura 7.241: Figura 320: Campo (Nodo\_origen\*)

## 7° SEPTIMO PASO

- Seleccionamos el nodo origen es decir el que quiero cambiar, por ejemplo el nodo (**Por\_llegar**), como se muestra en la siguiente *Figura 321: Selección de nodo a cambiar*

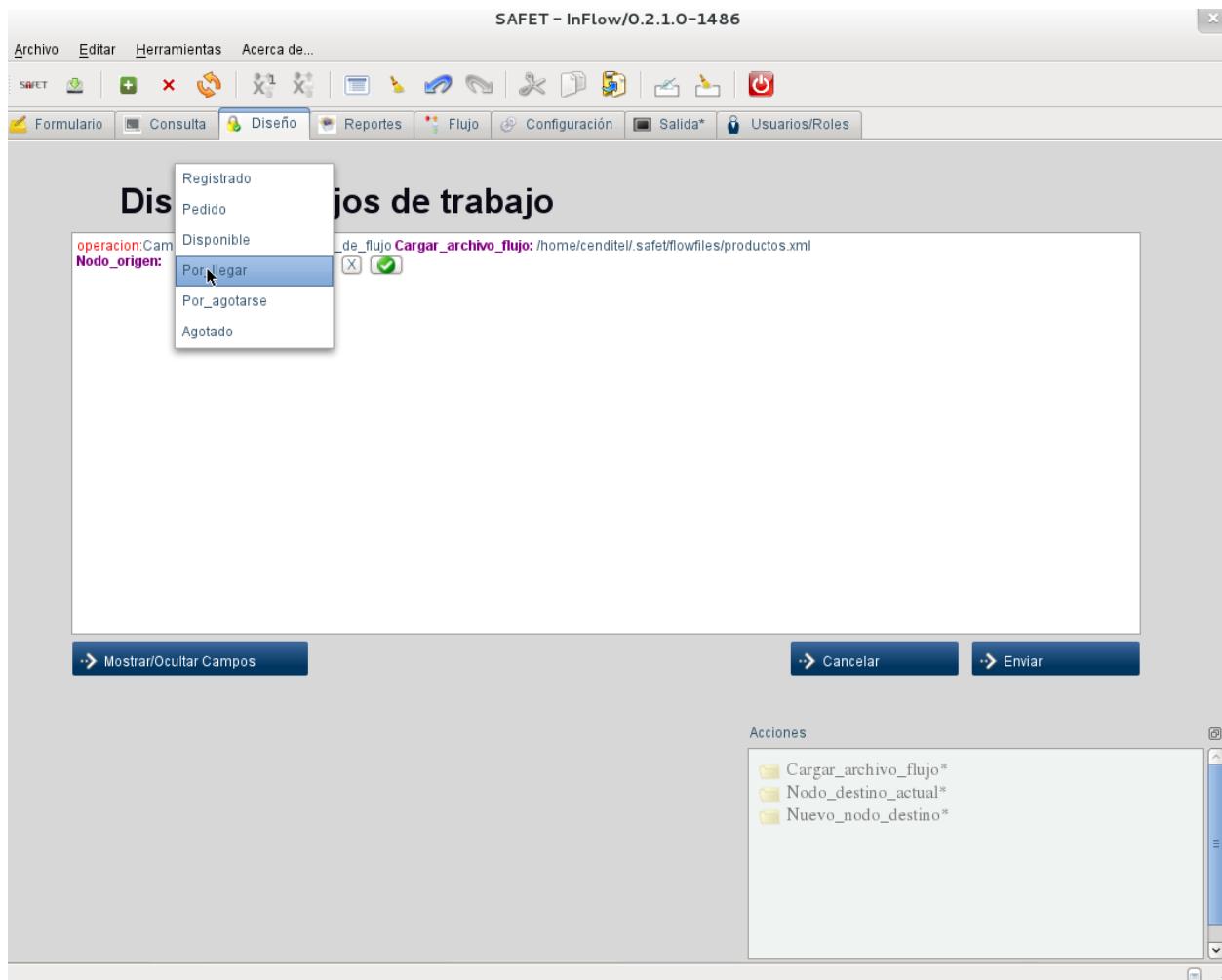


Figura 7.242: **Figura 321: Selección de nodo a cambiar**

## 8° OCTAVO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 322: Botón (Fin de campo)*

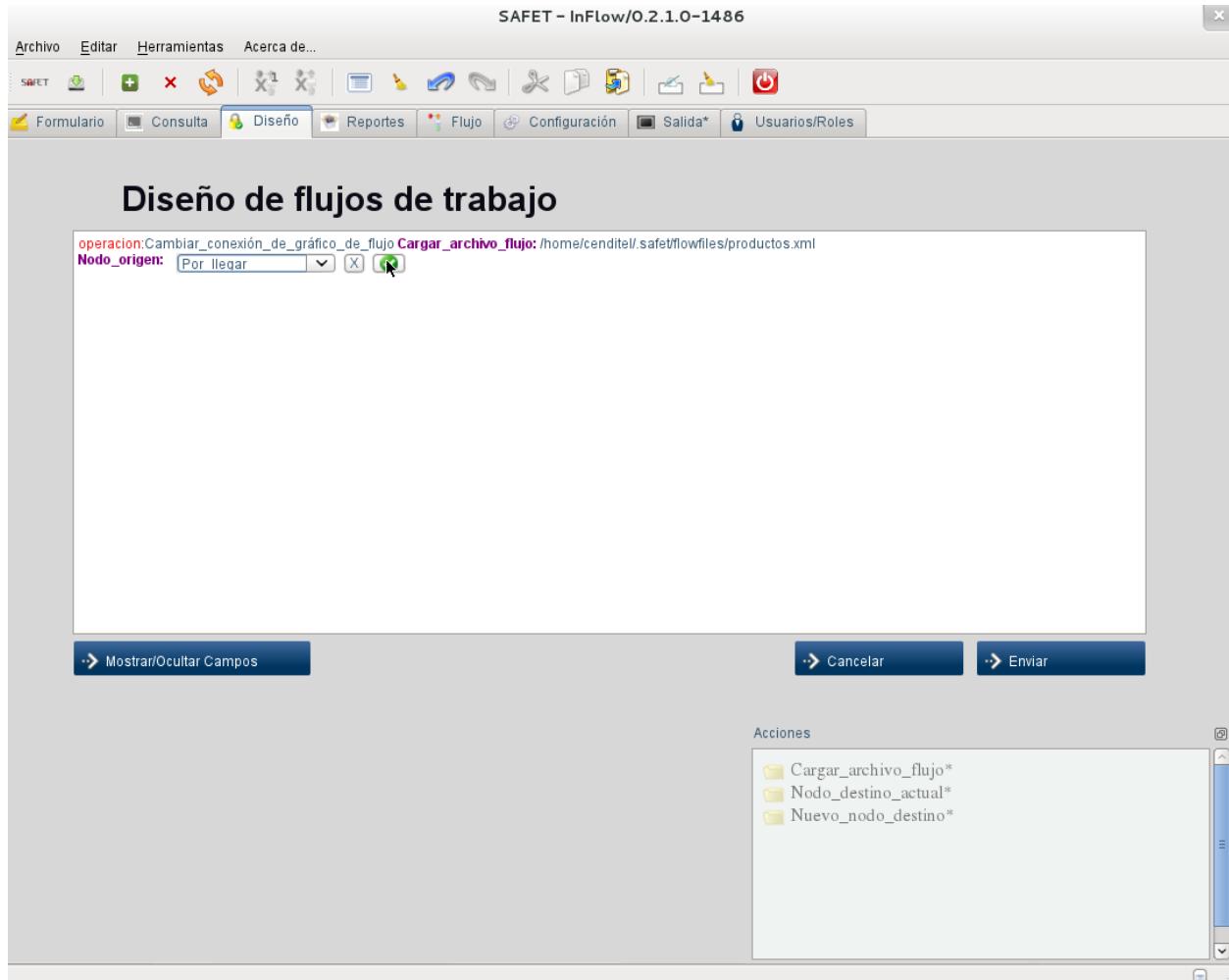


Figura 7.243: Figura 322: Botón (Fin de campo)

## 9° NOVENO PASO

- Damos click al siguiente campo obligatorio (**Nodo\_destino\_actual\***), como se muestra en la siguiente *Figura 323: Campo (Nodo\_destino\_actual\*)*

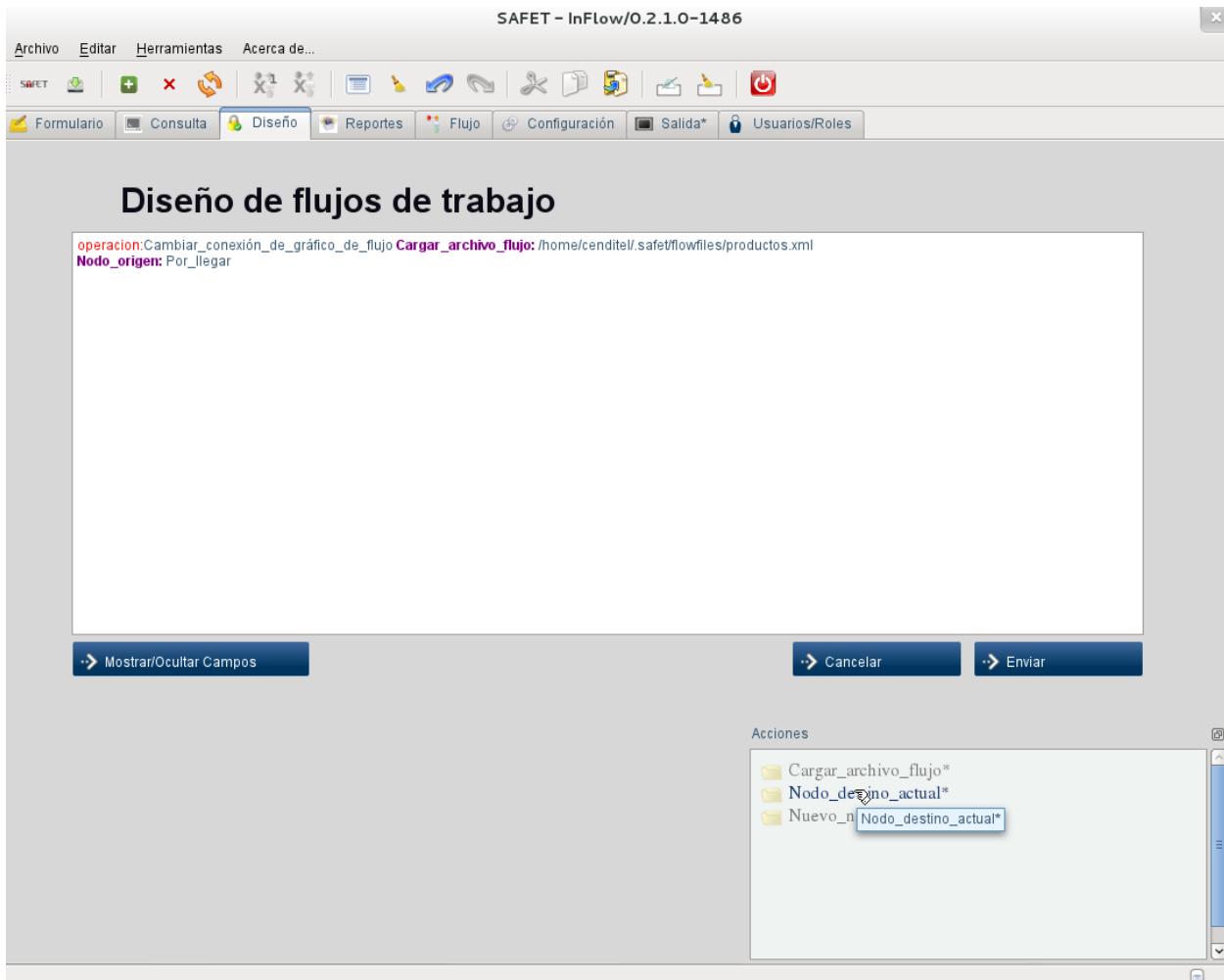


Figura 7.244: *Figura 323: Campo (Nodo\_destino\_actual\*)*

## 10° DECIMO PASO

- Seleccionamos el nodo destino actual, es decir donde el nodo origen esta apuntando en este caso seria el nodo (**Disponible**), como se muestra en la siguiente *Figura 324: Seleccionamos el (nodo destino actual)*

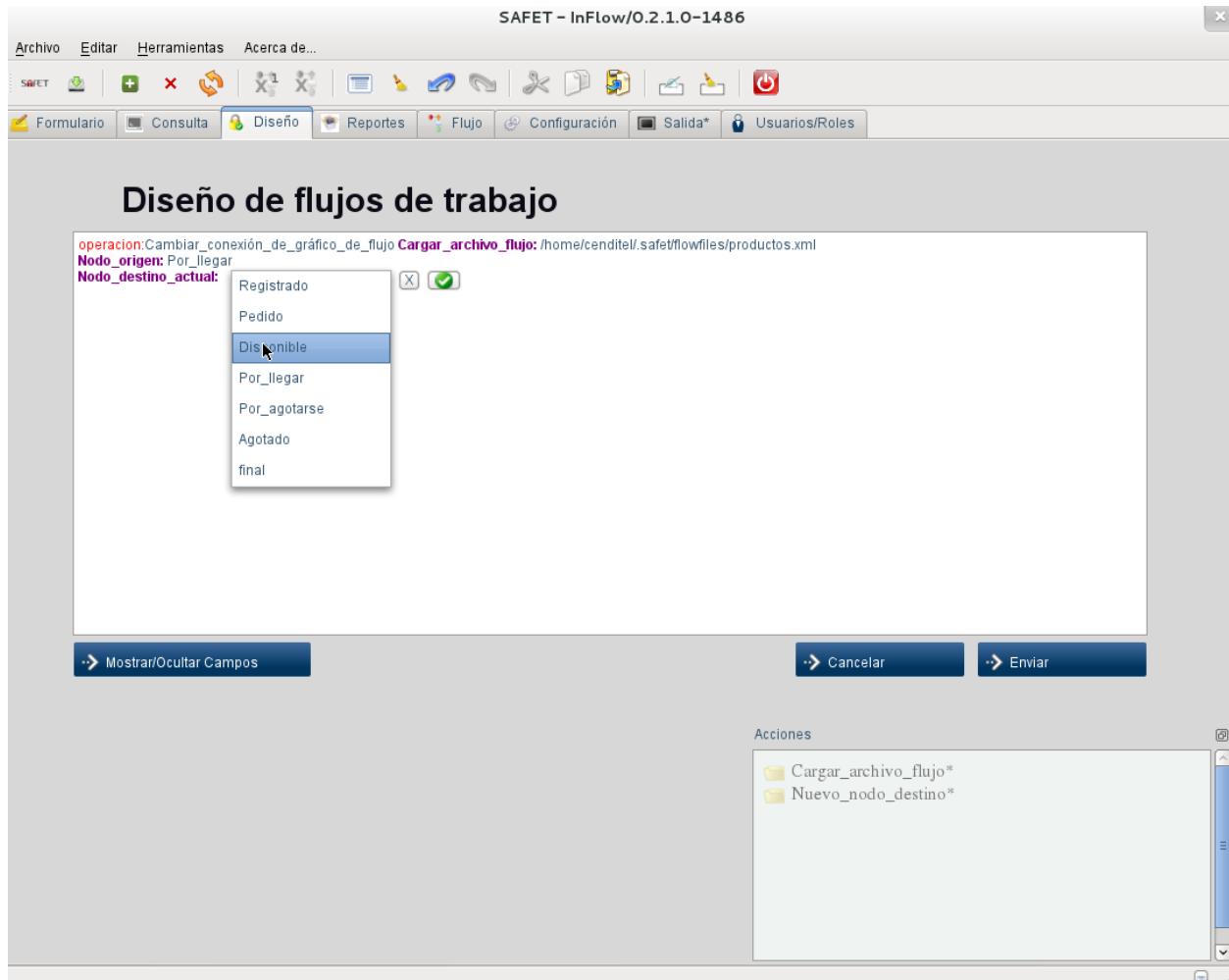


Figura 7.245: Figura 324: Seleccionamos el (nodo destino actual)

## 11° DECIMO PRIMERO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 325: Botón (Fin de campo)*

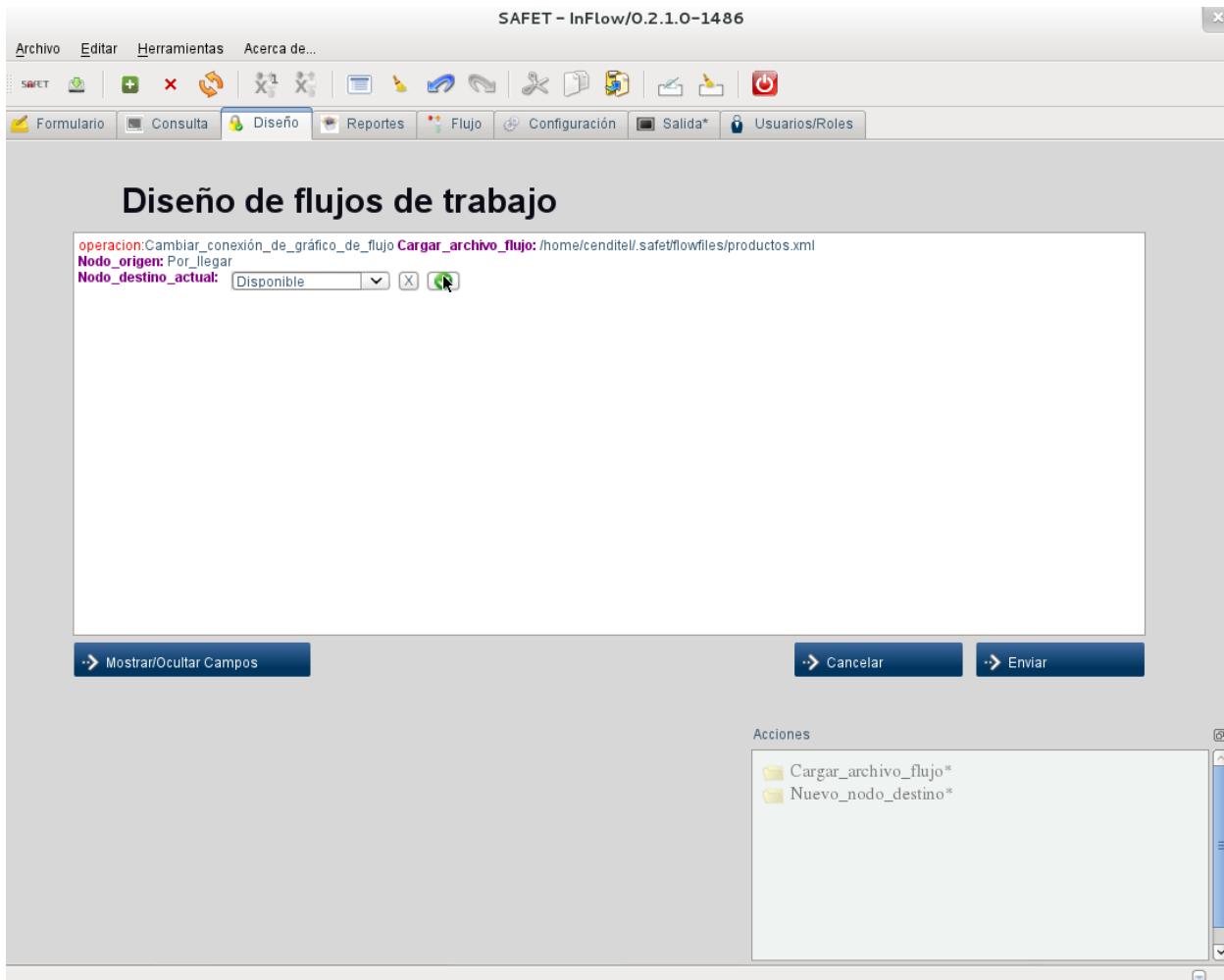


Figura 7.246: *Figura 325: Botón (Fin de campo)*

## 12° DECIMO SEGUNDO PASO

- Damos click al siguiente campo obligatorio (**Nuevo\_nodo\_destino\***), como se muestra en la siguiente *Figura 326: Campo (Nuevo\_nodo\_destino\*)*

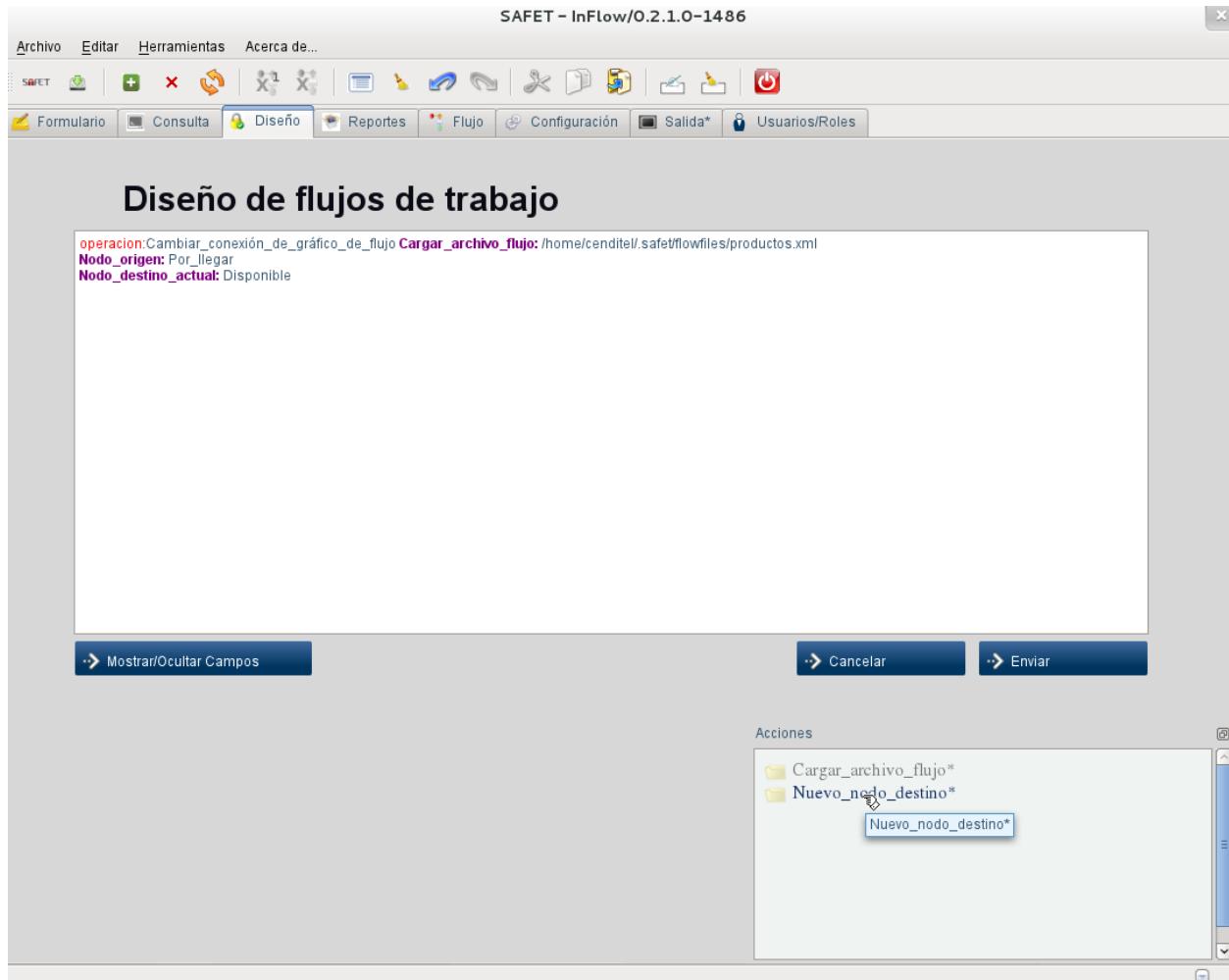


Figura 7.247: Figura 326: Campo (Nuevo\_nodo\_destino\*)

### 13° DECIMO TERCER PASO

- Seleccionamos el nodo nuevo donde el nodo origen (**Por\_llegar**) va apuntar, por ejemplo que a punte al nodo (**Por\_agotarse**), como se muestra en la siguiente *Figura 327: Seleccionamos el nuevo nodo*

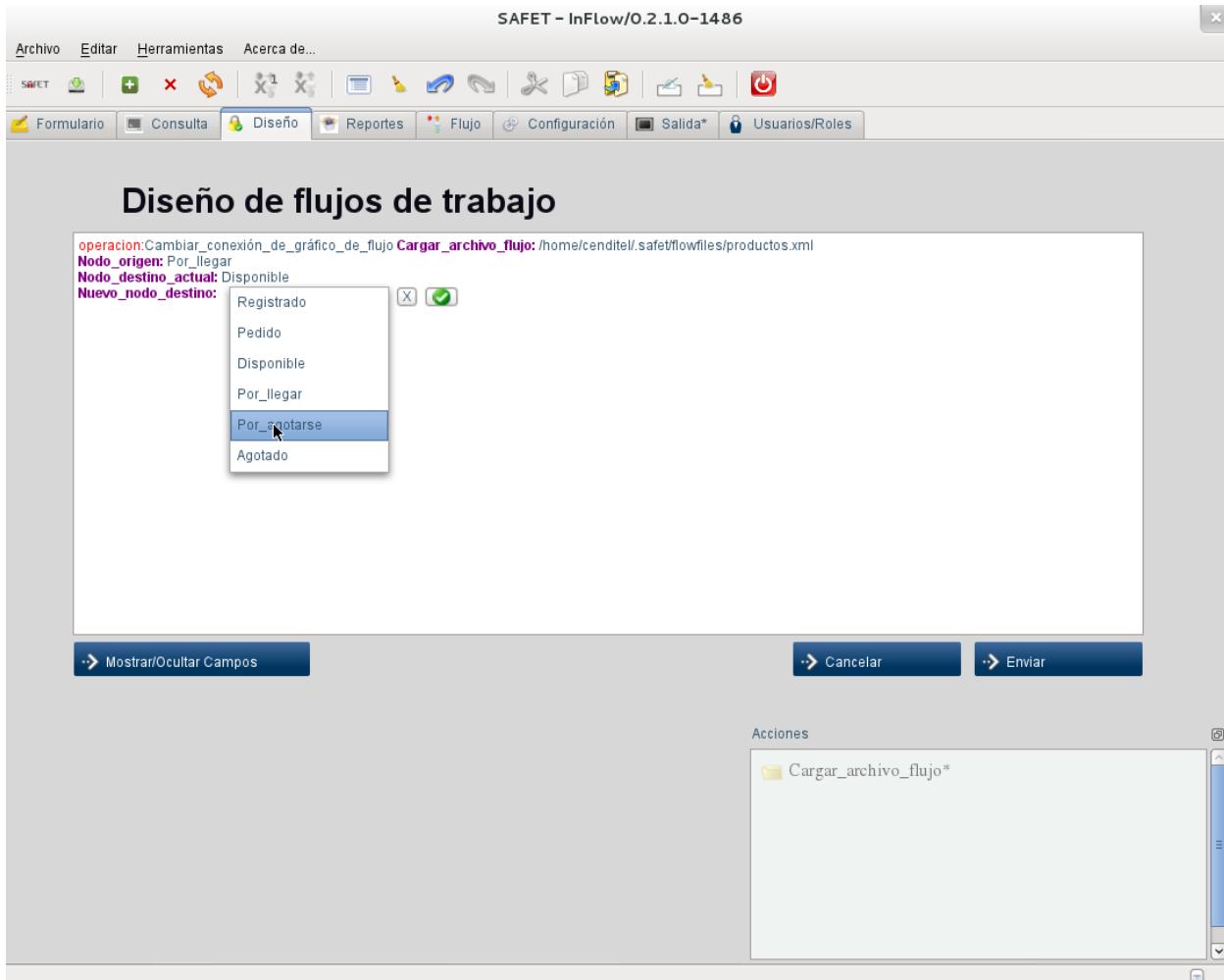


Figura 7.248: *Figura 327: Seleccionamos el nuevo nodo*

### 14° DECIMO CUARTO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 328: Botón (Fin de campo)*

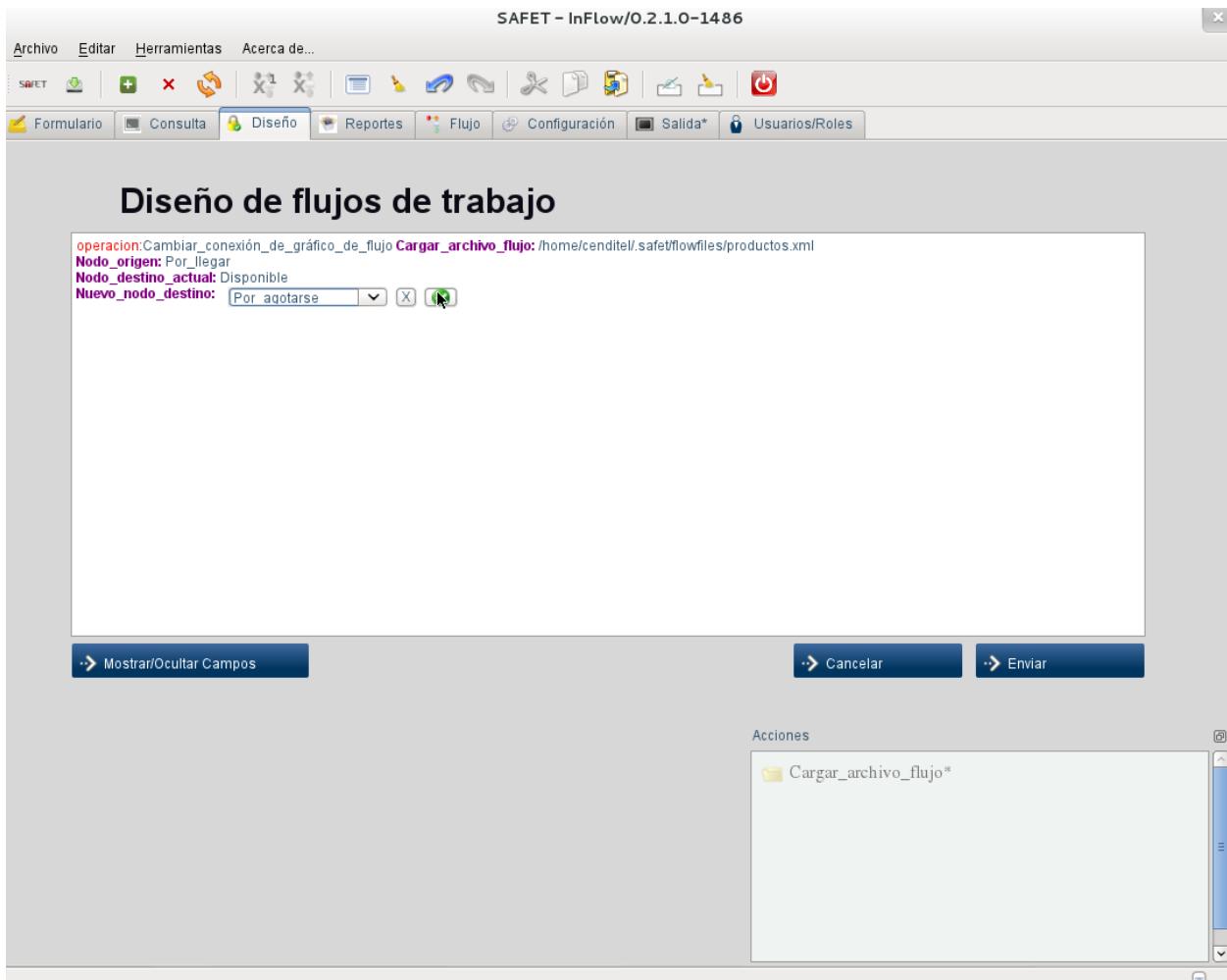


Figura 7.249: Figura 328: Botón (Fin de campo)

## 15° DECIMO QUINTO PASO

- Damos un click al botón (**Enviar**) para finalizar con la operación la cual se nos mostrara como resultado (**Se cambió la conexión en el grafo “/home/cenditel/.safet/flowfiles/productos.xml” de “Disponible” a “Por\_agotarse” satisfactoriamente! ....ok!**), como se muestra en la siguiente *Figura 329: Fin de la operación*

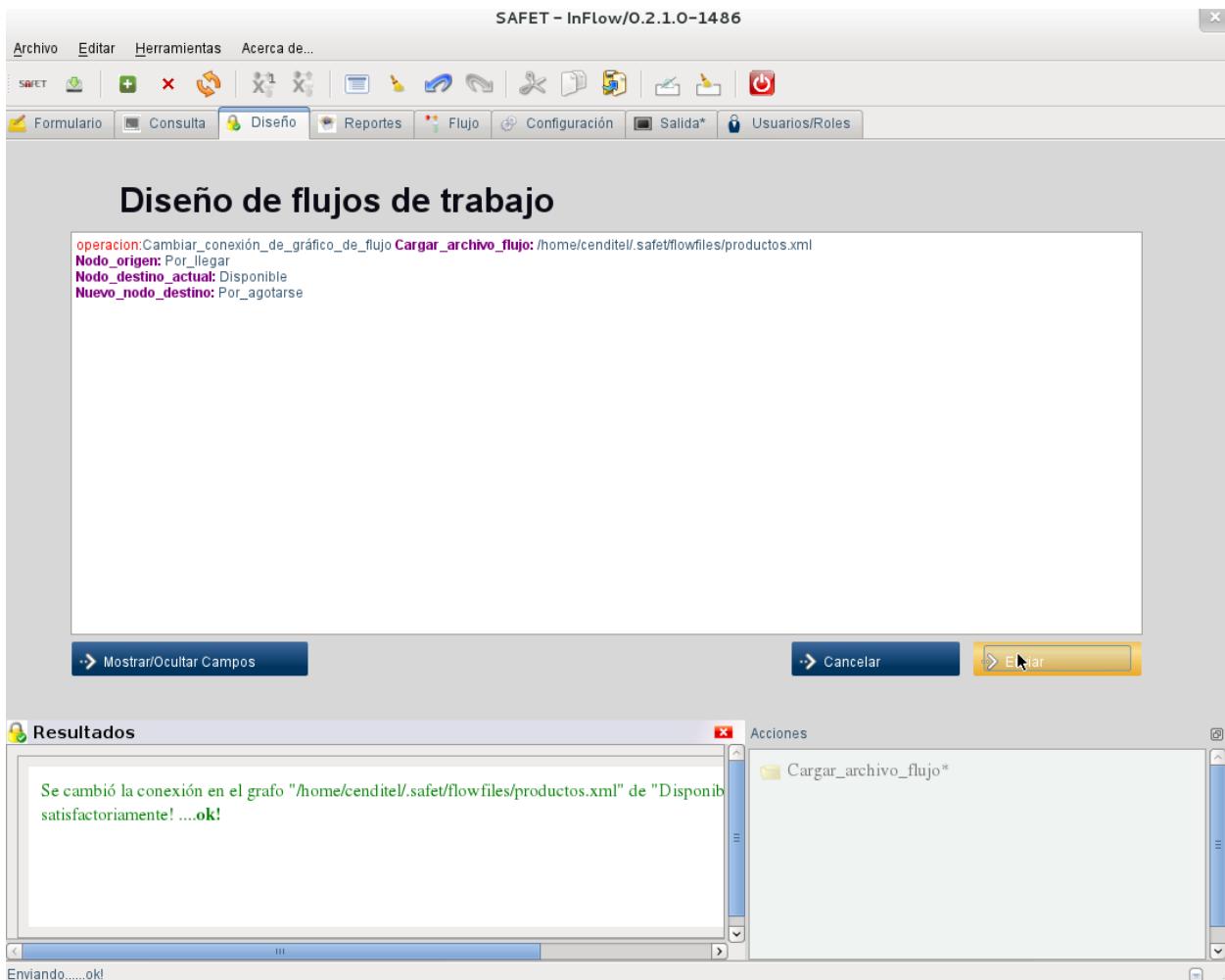


Figura 7.250: **Figura 329: Fin de la operación**

### 7.11.3 C.- Observemos el cambio de conexiones de los nodos en el gráfico de flujo de trabajo

Para observar la gráfica del archivo (**productos.xml**) del directorio (**/home/usuario/.safet/flowfiles/**), para ello damos click al siguiente **link**. *B.- PRIMERA OPERACIÓN (Gráfico coloreado general)*

**Nota:** Se nos mostrara en la gráfica el nodo que cambio de dirección como se muestra en la siguiente figura *Figura 330: Gráfico de cambio de nodo*

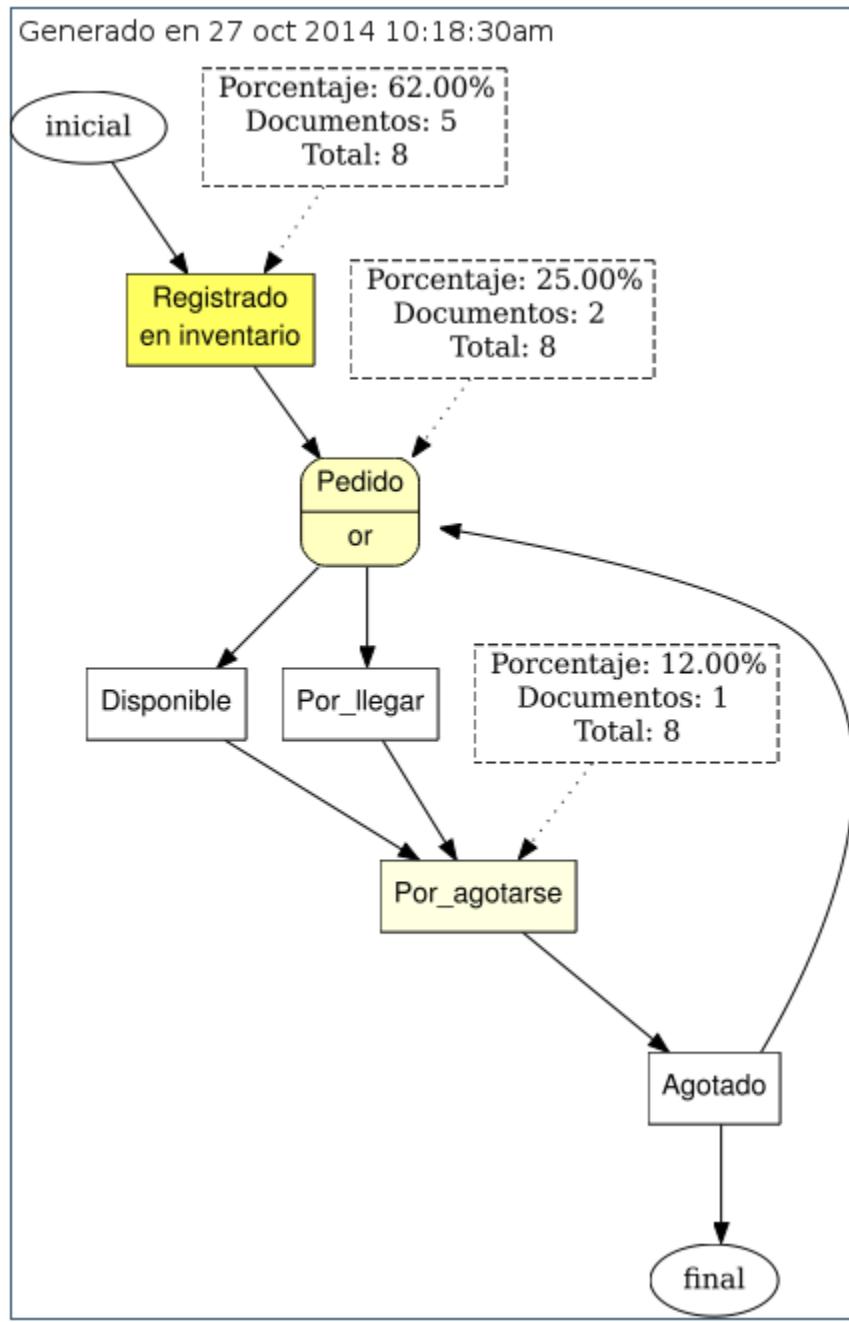


Figura 7.251: Figura 330: Gráfico de cambio de nodo

## 7.12 Diseño de borrar un nodos del gráficos de flujo de trabajo

En el gráfico coloreado se nos muestra el siguiente *Figura 331: Gráfico general*, en cual vamos a borrar el nodo (**Cantidad100**) para ello seguimos los siguientes paso:

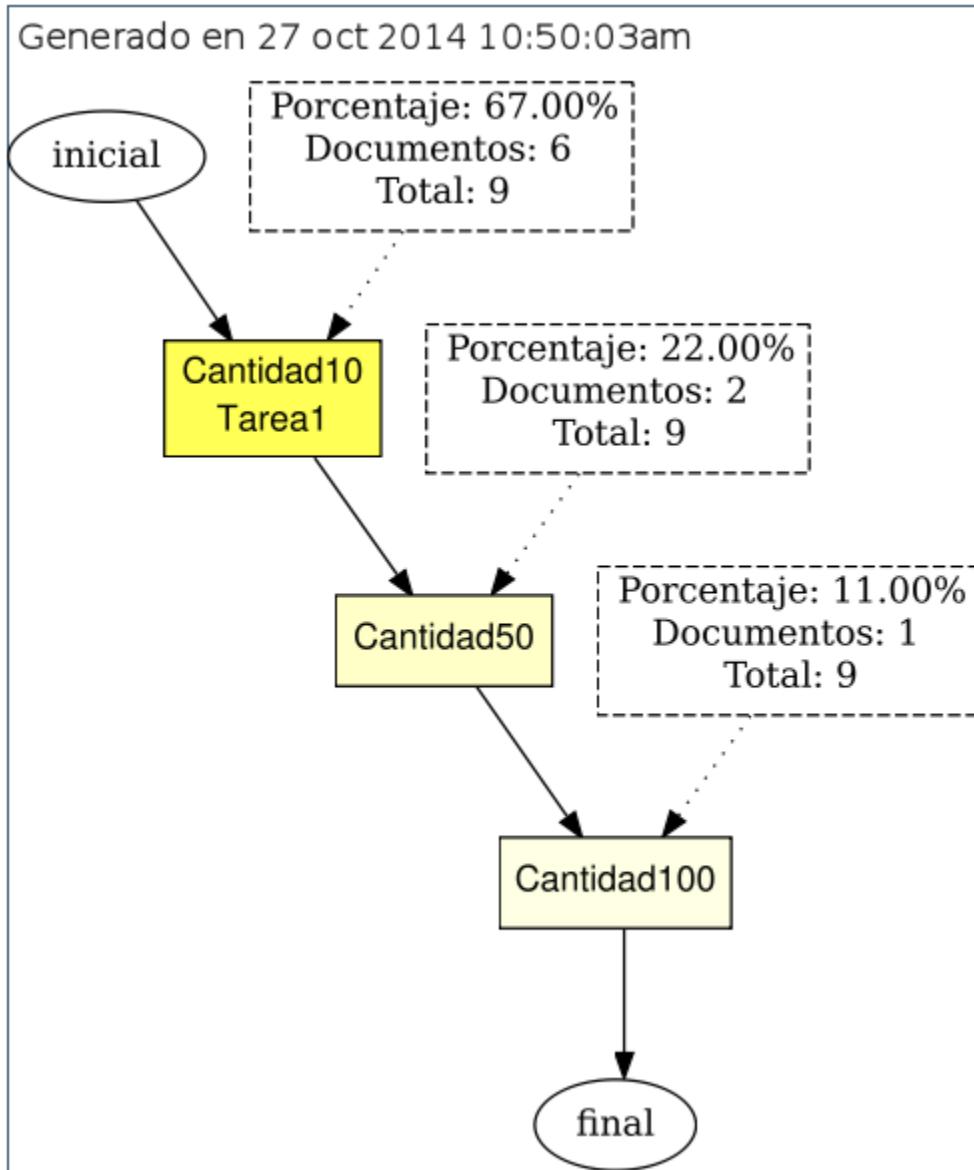


Figura 7.252: **Figura 331: Gráfico general**

### 7.12.1 A.- Abrimos la interfaz gráfica de inflow

#### 1° PRIMER PASO

- Desde la consola de comando, como usuario **normal**, ejecutamos **inflow** como se muestra a continuación:

```
$ inflow
```

#### 2° SEGUNDO PASO

- Agregamos el **Usuario/password-(admin/admin)**, como se muestra en la siguiente *Figura 332: Usuario/password*



Figura 7.253: **Figura 332: Usuario/password**

#### 3° TERCER PASO

- Damos click en la segunda opción (**Frimar/verificar documento**), como se muestra en la siguiente *Figura 333: Opcion Diseño*

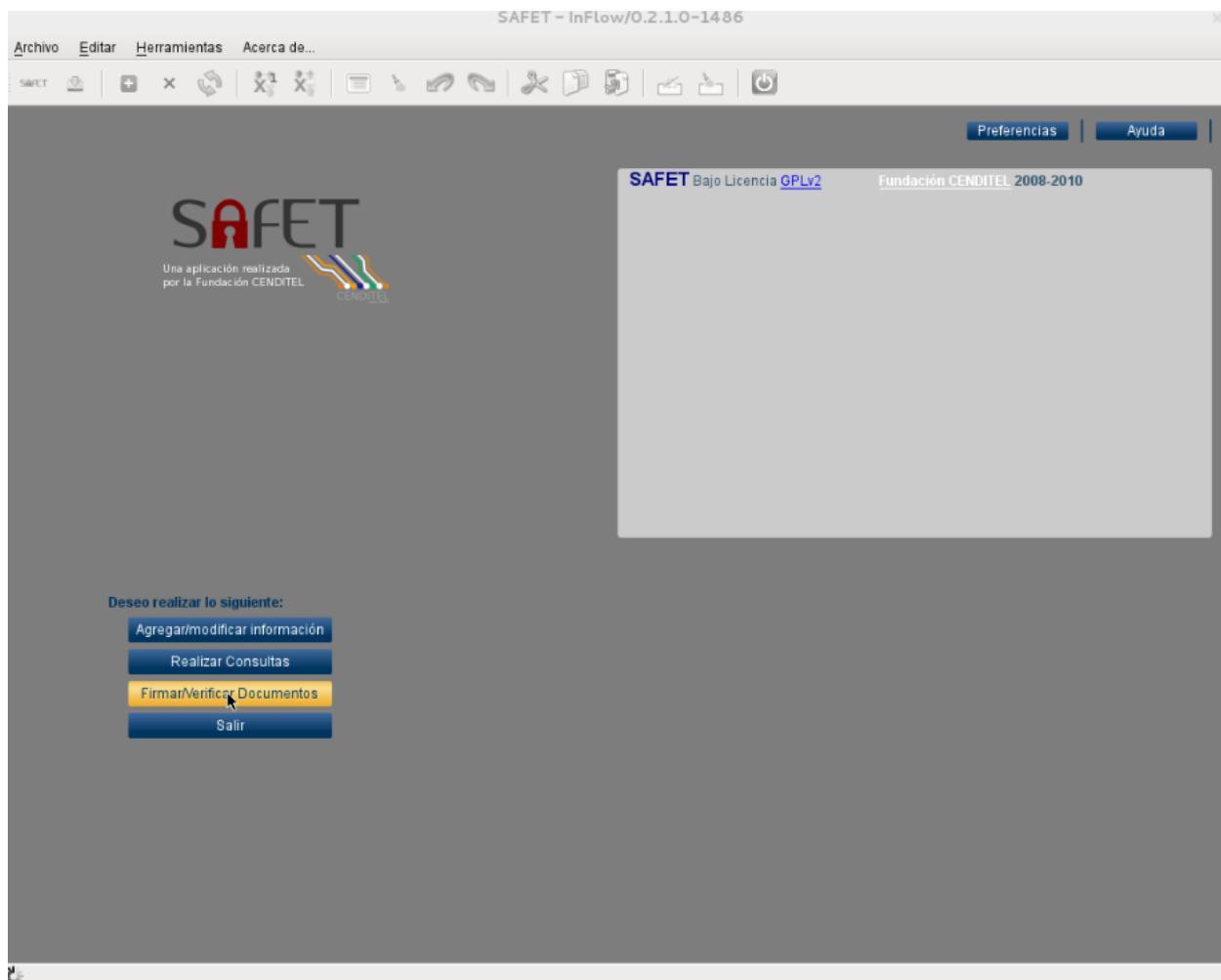


Figura 7.254: **Figura 333: Opcion Diseño**

### 4° CUARTO PASO

- Damos click a la segunda opción (**Mostar/Ocultar Campos**), como se muestra en la siguiente *Figura 334: Campos del Diseño*

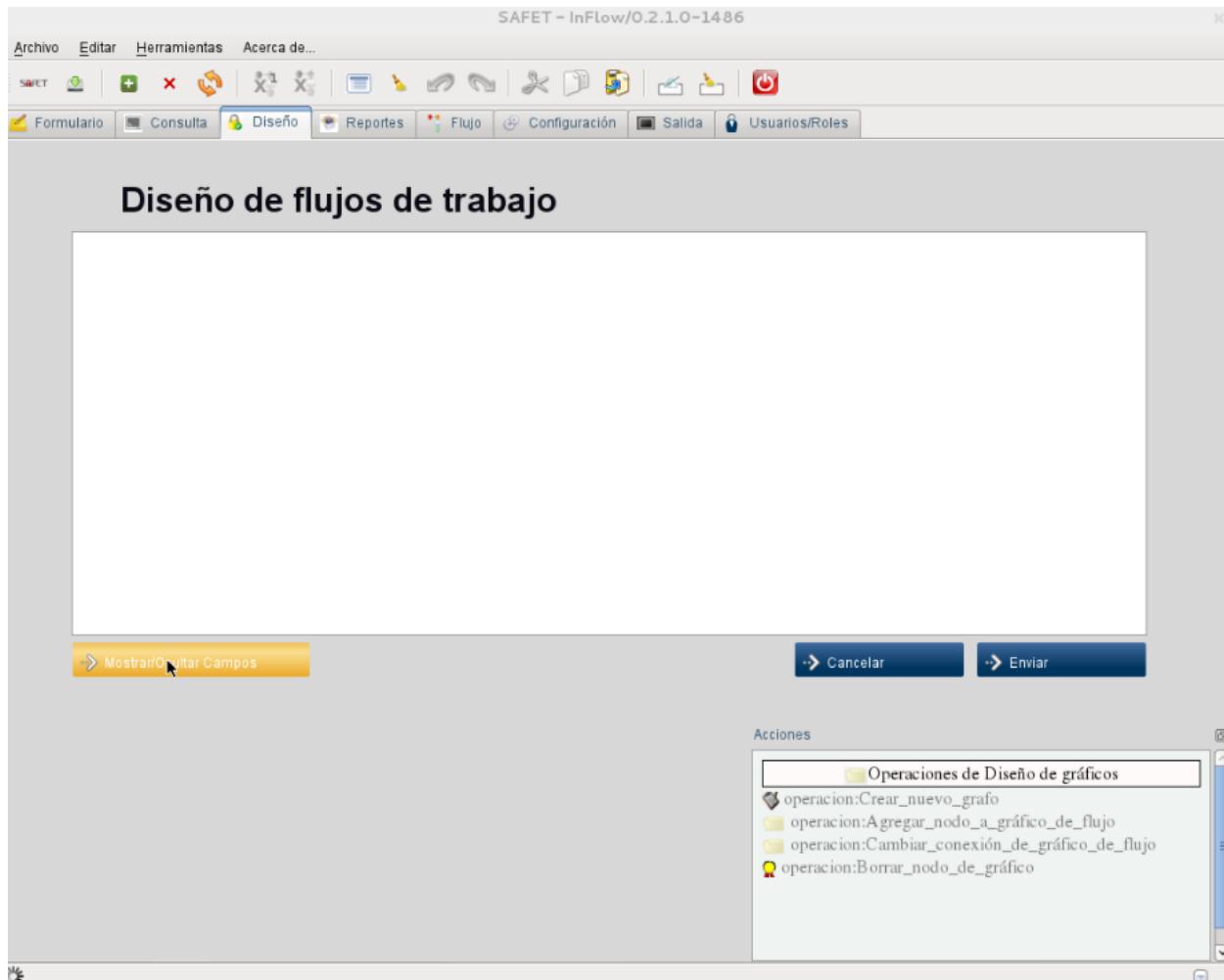


Figura 7.255: **Figura 334: Campos del Diseño**

### 7.12.2 B.- Borrar nodo del gráfo

#### 1° PRIMER PASO

- Damos click a la operación (**Borrar\_nodo\_de\_gráfico**), como se muestra en la siguiente *Figura 335: Borrar\_nodo\_de\_gráfico*

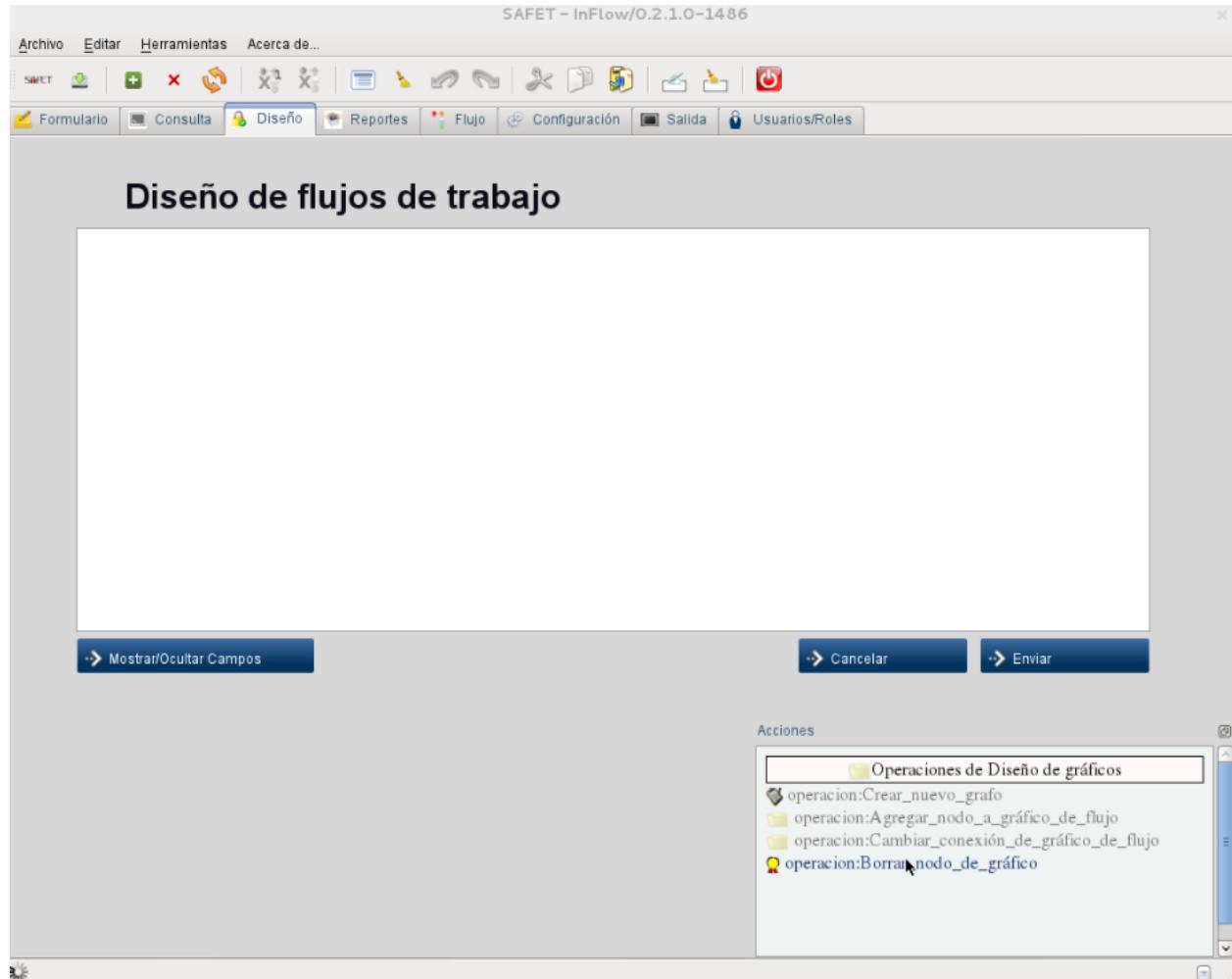


Figura 7.256: Figura 335: Borrar\_nodo\_de\_gráfico

## 2° SEGUNDO PASO

- Damos click al campo obligatorio (**Cargar\_archivo\_flujo\***), como se muestra en la siguiente *Figura 336: Campo (Cargar\_archivo\_flujo\*)*

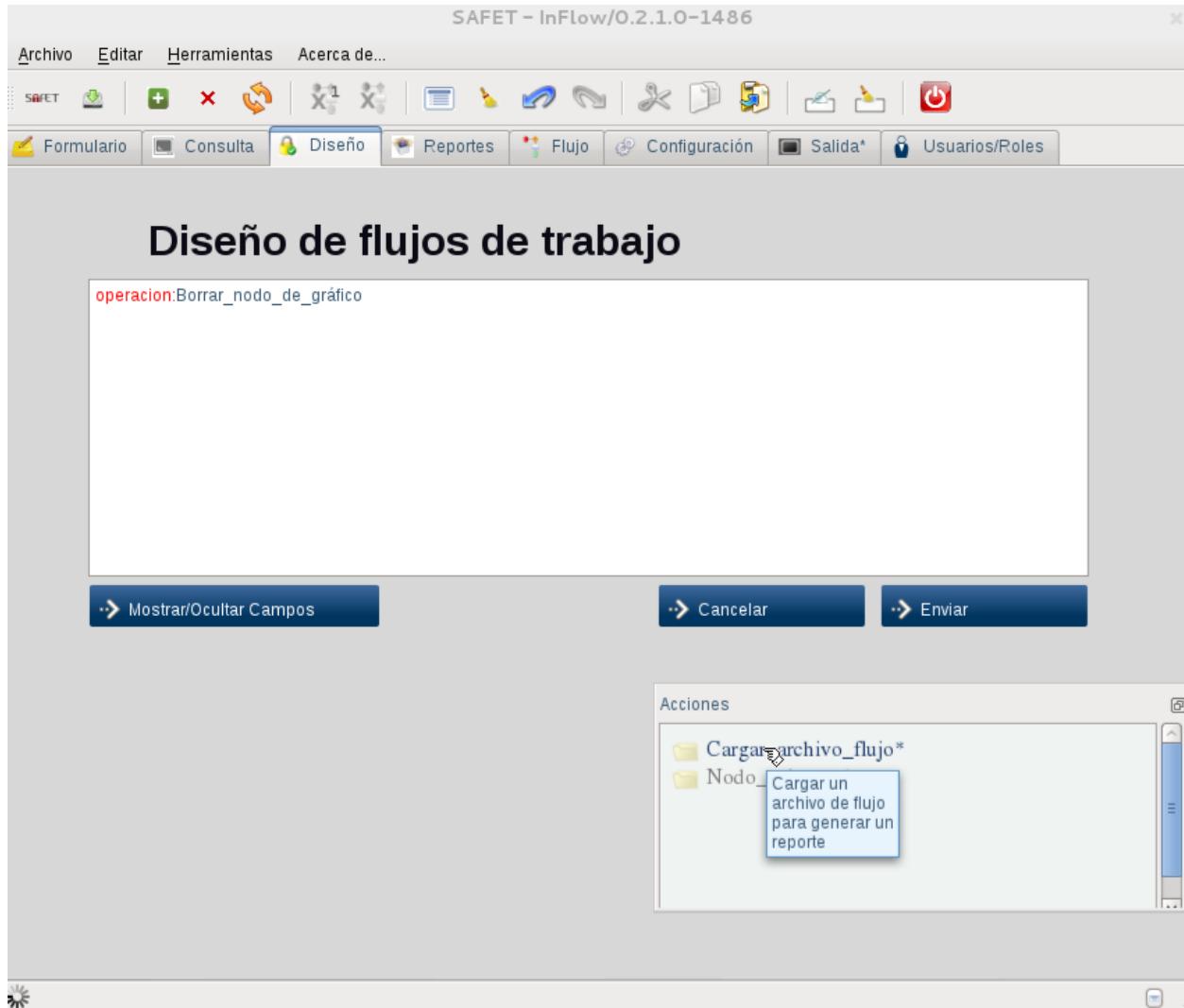


Figura 7.257: **Figura 336: Campo (Cargar\_archivo\_flujo\*)**

## 3° TERCER PASO

- Damos click al botón para buscar el archivo (**Ejemplo1.xml**), como se muestra en la siguiente *Figura 337: Botón buscar*

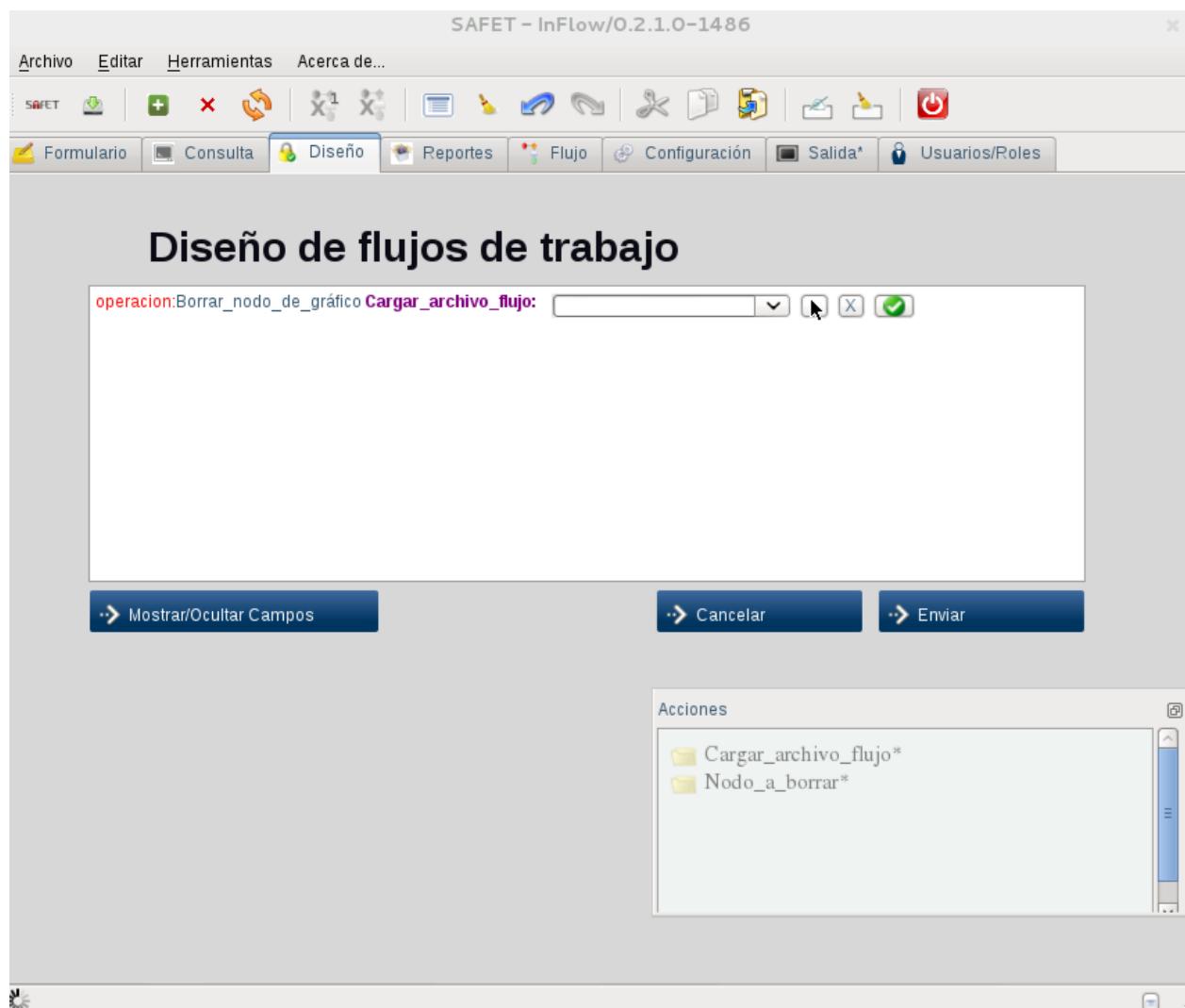


Figura 7.258: **Figura 337: Botón buscar**

### 4° CUARTO PASO

- Seleccionamos el archivo (**Ejemplo1.xml**) del directorio (**/home/usuario/.safet/flowfiles**) y pulsamos en botón (**Abrir**), como se muestra en la siguiente *Figura 338: Seleccionar archivo (.xml)*

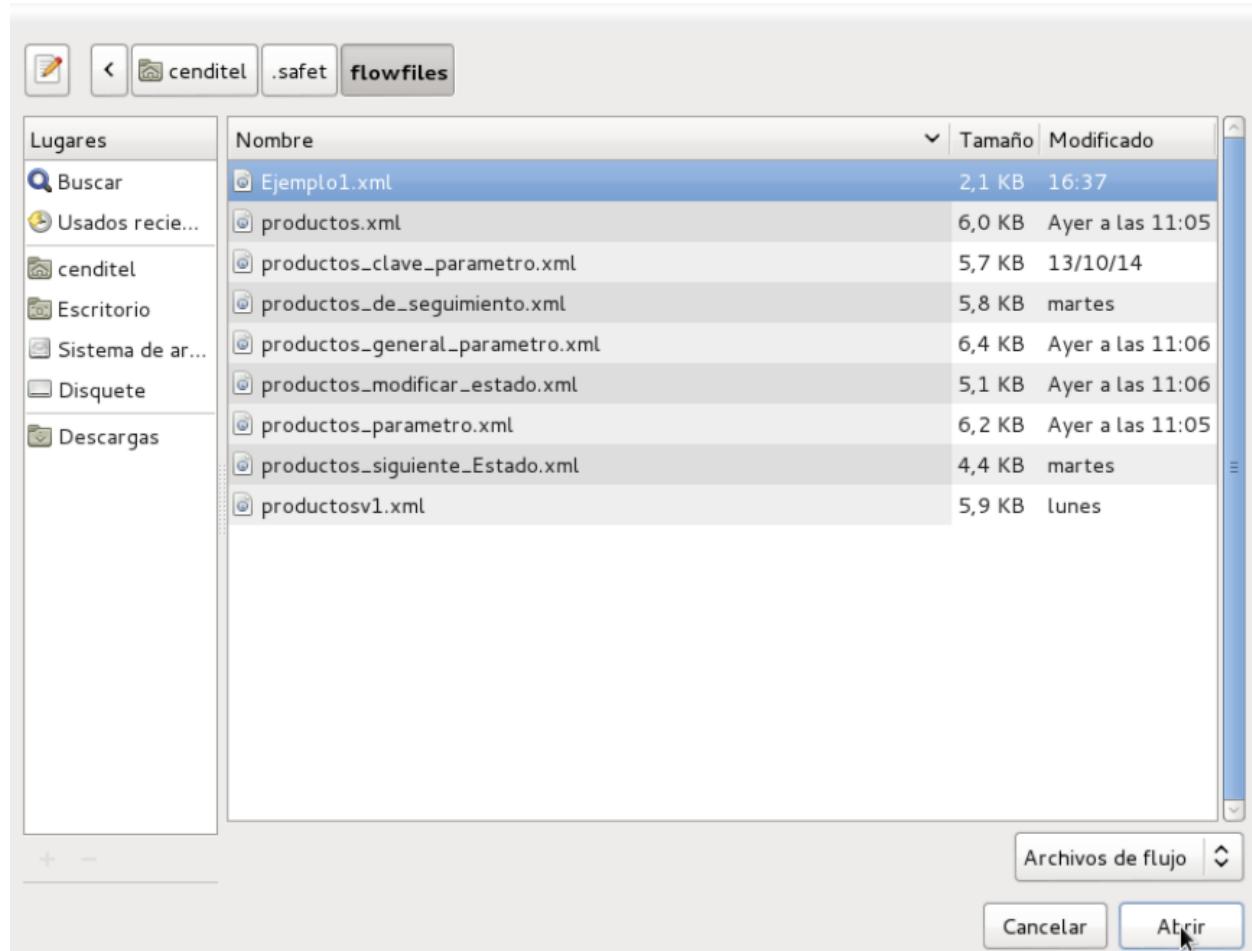


Figura 7.259: **Figura 338: Seleccionar archivo (.xml)**

### 5° QUINTO PASO

- Damos un click al botón con la flecha (**verde**) significando que ya está lleno el campo, como se muestra en la siguiente *Figura 339: Botón (Fin de campo)*

### 6° SEXTO PASO

- Damos click al siguiente campo obligatorio (**Nodo\_a\_borrar\***), como se muestra en la siguiente *Figura 340: Campo (Nodo\_a\_borrar\*)*

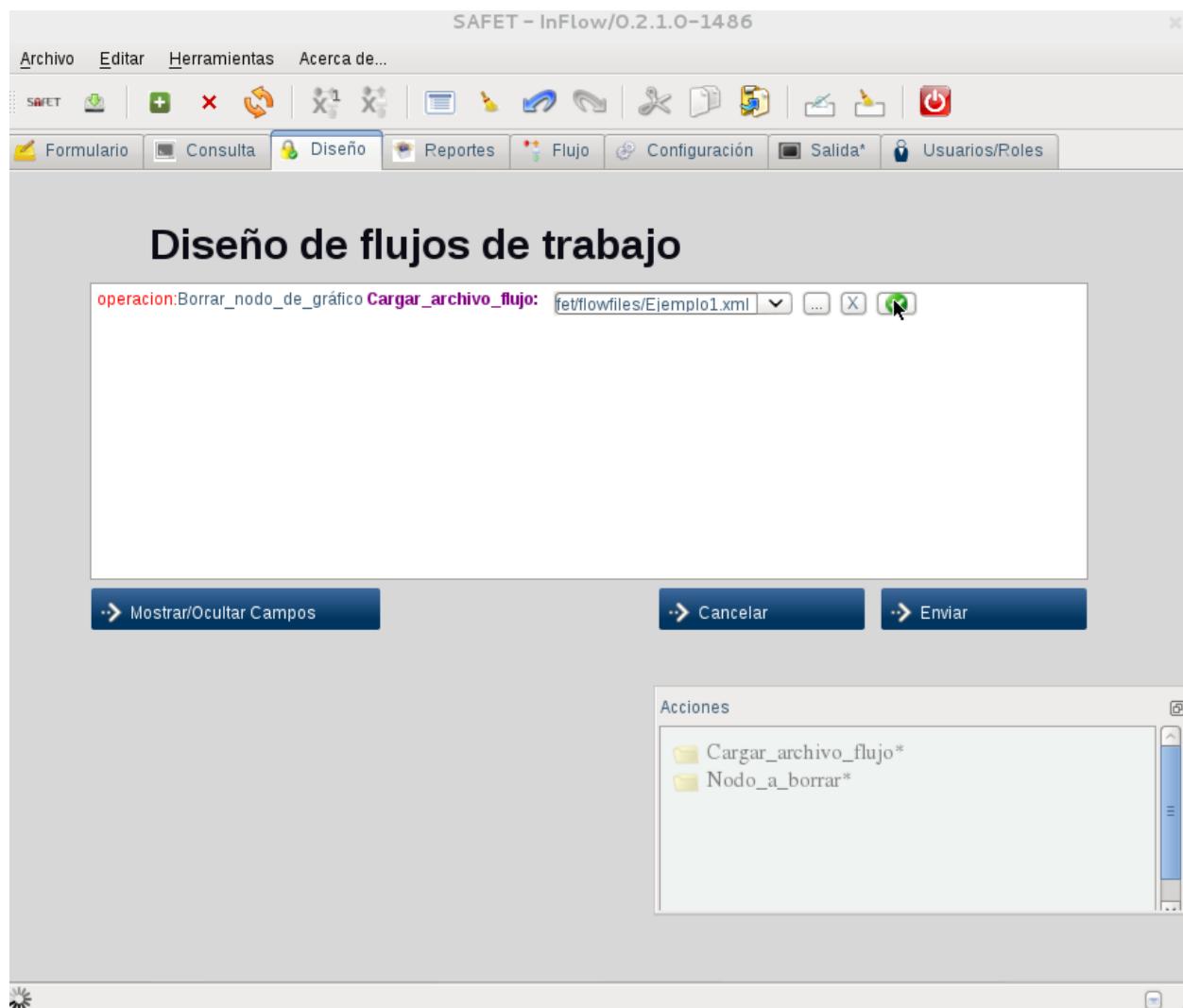


Figura 7.260: Figura 339: Botón (Fin de campo)

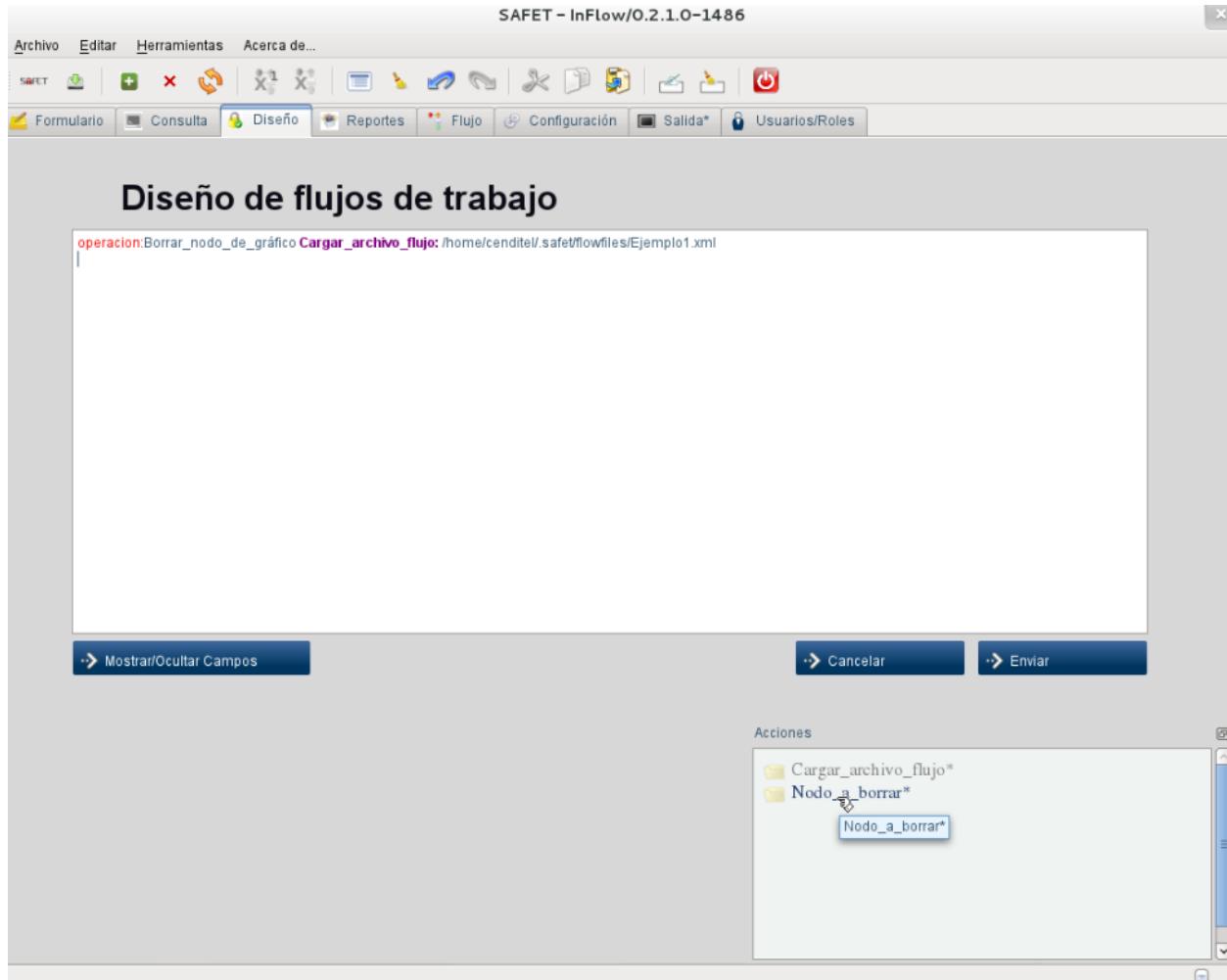


Figura 7.261: **Figura 340: Campo (Nodo\_a\_borrar\*)**

## 7° SEPTIMO PASO

- Seleccionamos el nodo a borrar de la gráfica, por ejemplo el nodo (**Cantidad100**), como se muestra en la siguiente *Figura 341: Seleccionar nodo a borrar*

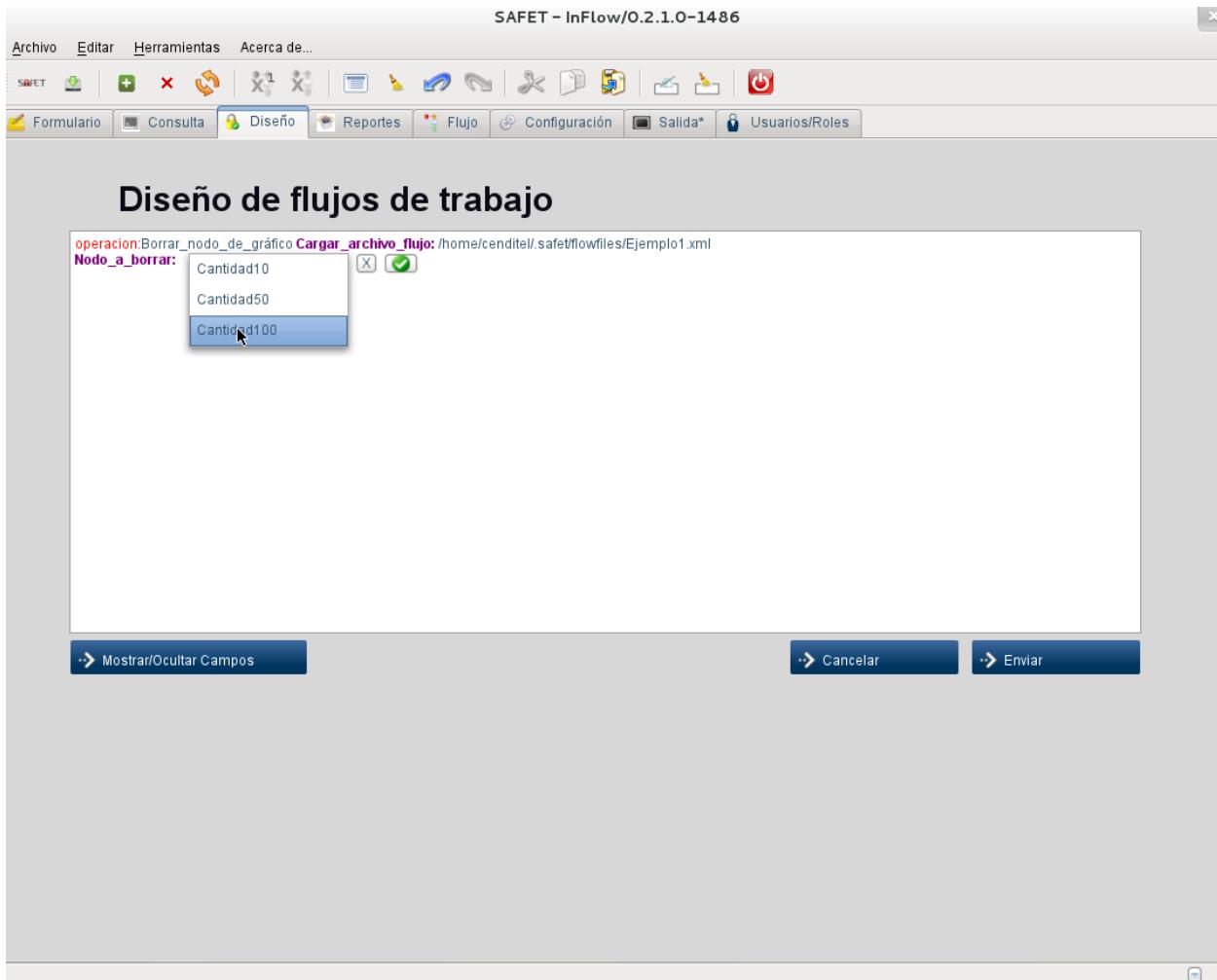


Figura 7.262: **Figura 341: Seleccionar nodo a borrar**

## 8° OCTAVO PASO

- Damos un click al **botón** con la flecha verde significando que ya está lleno el campo, como se muestra en la siguiente *Figura 342: Botón (Fin de campo)*

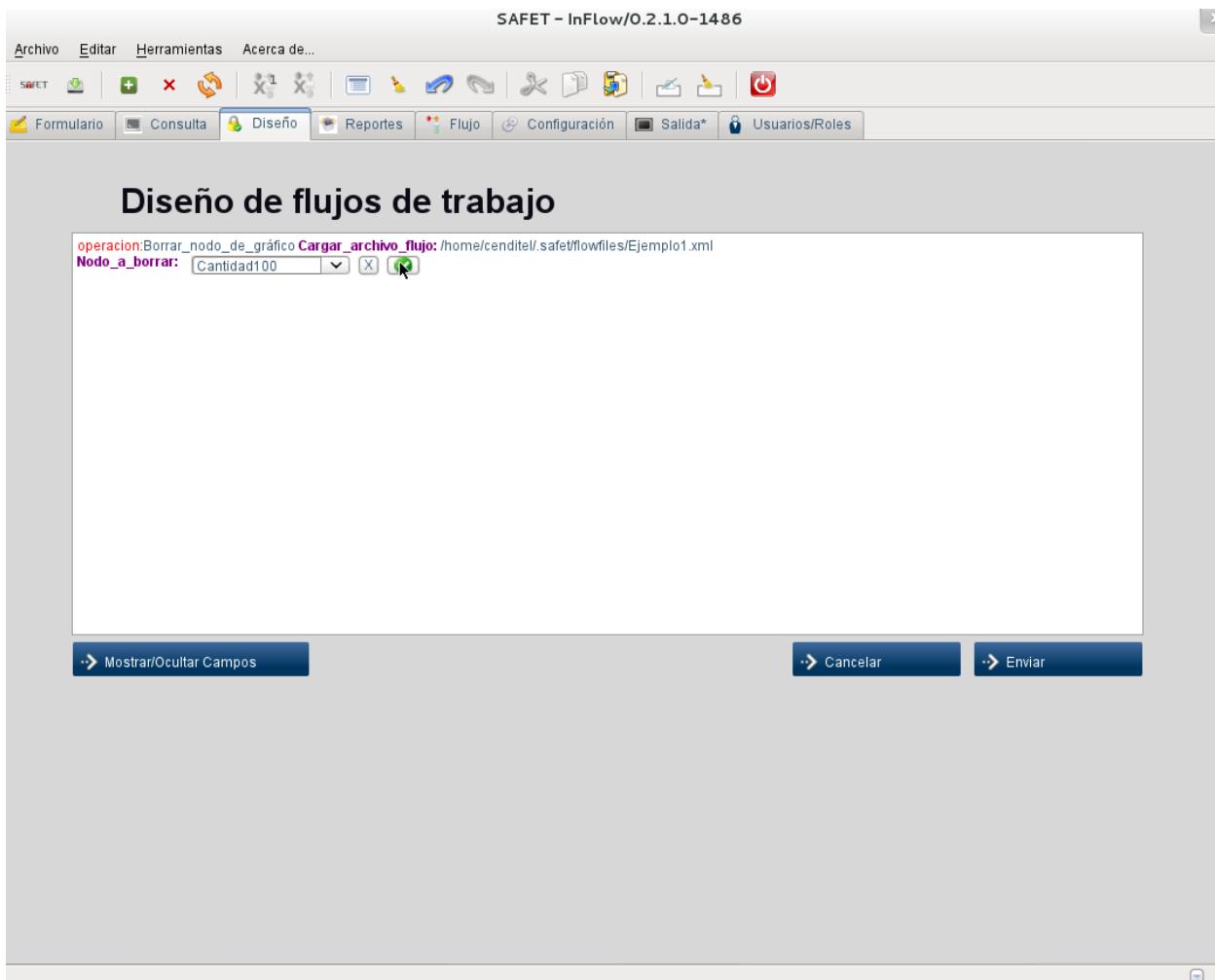


Figura 7.263: Figura 342: Botón (Fin de campo)

## 9° NOVENO PASO

- Damos un click al botón (**Enviar**) para finalizar con la operación, la cual se nos mostrara como resultado (**Eliminado nodo “Cantidad100” de “/home/cenditel/.safet/flowfiles/Ejemplo1.xml” satisfactoriamente! ....ok!**), como se muestra en la siguiente *Figura 343: Fin de la operación*

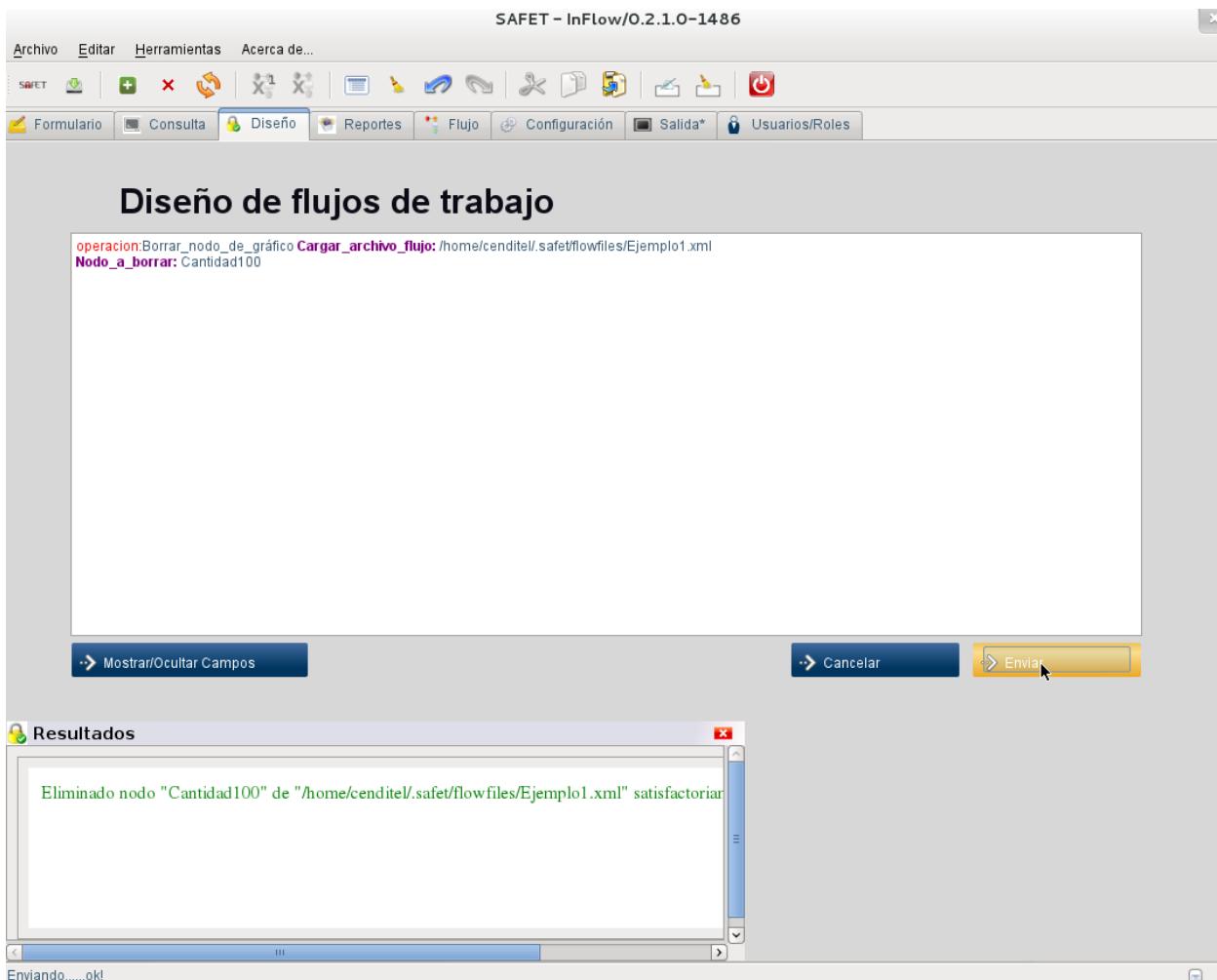


Figura 7.264: **Figura 343: Fin de la operación**

### 7.12.3 C.- Observemos en el gráfico de flujo de trabajo el nodo Borrado

Para observar la gráfica del archivo (**Ejemplo1.xml**) del directorio (**/home/usuario/.safet/flowfiles/**), para ello damos click al siguiente **link**. *B.- PRIMERA OPERACIÓN (Gráfico coloreado general)*

**Nota:** Se nos mostrara en la gráfica sin el nodo (**Cantidad100**) la cual los borramos en el paso anterior *Figura 344: Gráfico de nodo borrado*

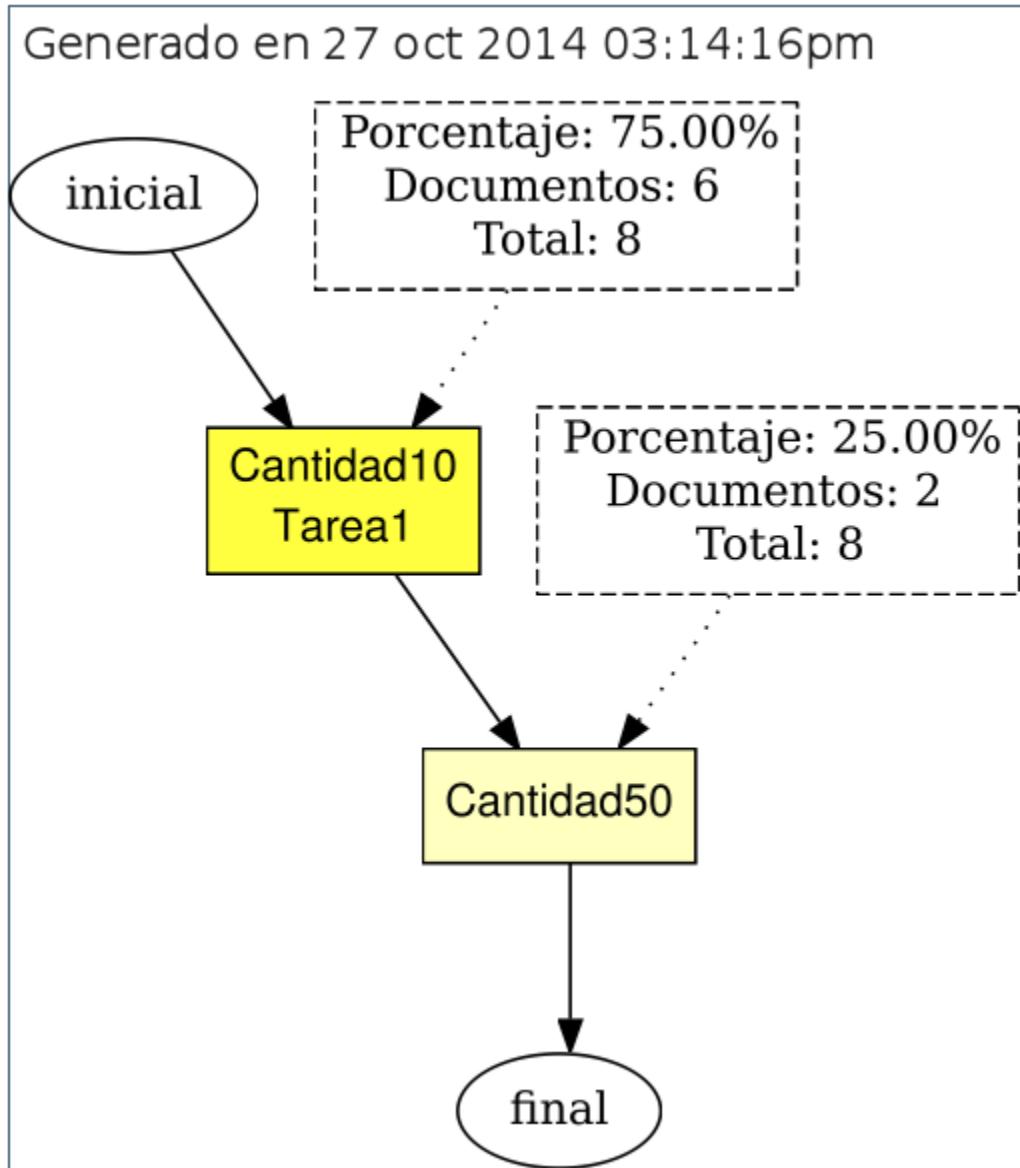


Figura 7.265: Figura 344: Gráfico de nodo borrado

---

## 8.- Tutorial del “ejemplo de inventario” usando la interfaz gráfica (Framework Web-Safet)

---

### 8.1 Instalación y configuración

En este tutorial instalaremos y configuración el framework **Web-Safet**, para ello debe tener instalado **PySafet** dando click al siguiente link. Instalación de PySafet

A continuación seguimos los siguientes pasos del (framework Web-Safet):

#### 8.1.1 A.- Instalación de servidor apache y su modulo python

##### 1° PRIMER PASO

- Desde la consola de comando como usuario (**root**), instalamos apache2 como se muestra en el siguiente código:

```
# aptitude install apache2
```

##### 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**root**),instalamos el modulo python como se muestra en el siguiente código:

```
# aptitude install libapache2-mod-python
```

##### 3° TERCER PASO

- Desde la consola de comando como usuario (**root**),instalamos **pika**, como se muestra en el siguiente código:

```
# easy_install pika
```

---

**Nota:** También lo puedes instalar con (**aptitude**), como se muestra en el siguiente código:

```
# aptitude install python-pika
```

---

### 4° CUARTO PASO

- Desde la consola de comando como usuario (**root**), instalamos **rpl**, como se muestra en el siguiente código:

```
# aptitude install rpl
```

#### 8.1.2 B.- Instalación de WebSafet

##### 1° PRIMER PASO

- Descargamos el archivo (**websafet.tar.gz**), dando un click al link que se muestra en la siguiente imagen:



Figura 8.1: WebSafet.tar.gz

##### 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**normal**), nos vamos al directorio (**/var/www/**) con el comando (**cd**), como se muestra en el siguiente código:

```
$ cd /var/www/
```

### 3° TERCER PASO

- Desde la consola de comando como usuario (**normal**), copiamos el archivo **media\_intranet.tar.gz** al directorio (**/var/www/**) con el comando (**cp**), la cual ya estas ubicado en el directorio, como se muestra en el siguiente código:

```
/var/www/ $ cp /home/cenditel/Descargas/websafet.tar.gz .
```

### 4° CUARTO PASO

- Desde la consola de comando como usuario (**normal**), descomprimimos el archivo **media\_intranet.tar.gz** con el comando **tar (xvzf)**, como se muestra en el siguiente código:

```
/var/www/ $ tar xvzf websafet.tar.gz
```

### 5° QUINTO PASO

- Desde la consola de comando como usuario (**normal**), con el comando (**rm**) eliminamos el archivo **media\_intranet.tar.gz** del directorio (**/var/www/**), como se muestra en el siguiente código:

```
/var/www/ $ rm websafet.tar.gz
```

### 6° SEXTO PASO

- Desde la consola de comando como usuario (**normal**), con el comando (**ls**) listamos los directorio (**intranet y medio**), como se muestra en el siguiente código:

```
/var/www/ $ ls
```

---

**Nota:** Se nos mostrara los dos directorio (intranet y media), como se muestra a continuación:

```
intranet  media
```

---

### 8.1.3 C.- Configuración de servidor apache2

#### 1° PRIMER PASO

- Descargamos el archivo **httpd.conf.tar.gz** dando un clik al link que se muestra en la siguiente imagen:



Figura 8.2: httpd.conf.tar.gz

## 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**root**), nos vamos al directorio **/etc/apache2/conf.d/** con el comando (**cd**), como se muestra en el siguiente código:

```
# cd /etc/apache2/conf.d/
```

---

**Nota:** Si no se encuentra el directorio (**conf.d**) se crea con el comando \*\*(**mkdir**), como se muestra a continuación:\*\*

```
# mkdir /etc/apache2/conf.d
```

---

## 3° TERCER PASO

- Desde la consola de comando como usuario (**root**), con el comando (**cp**) copiamos el archivo **httpd.conf** que descargamos en el paso anterior al directorio **/etc/apache2/conf.d/**, como se muestra en el siguiente código:

```
/etc/apache2/conf.d/ # cp /home/cenditel/Descargas/httpd.conf.tar.gz .
```

## 4° CUARTO PASO

- Desde la consola de comando como usuario (**root**), descomprimimos el archivo (**httpd.conf.tar.gz**) con el comando **tar(xzvf)**, como se muestra en el siguiente código:

```
/etc/apache2/conf.d/ # tar xvzf httpd.conf.tar.gz
```

## 5° QUINTO PASO

- Desde la consola de comando como usuario (**root**), borramos el archivo comprimido (**httpd.conf.tar.gz**) con el comando (**rm**), como se muestra en el siguiente código:

```
/etc/apache2/conf.d/ # rm httpd.conf.tar.gz
```

## 6° SEXTO PASO

- Desde la consola de comando como usuario (**root**), reiniciamos el servicios **apache2** como se muestra en el siguiente código:

```
/etc/apache2/conf.d/ # /etc/init.d/apache2 restart
```

---

**Nota: Al reiniciar el servidor apache2 se nos mostrara el siguiente mensaje:**

```
[....] Restarting web server: apache2[apache2: Could not reliably
determine the server's fully qualified domain name, using
127.0.1.1 for ServerName
... waiting apache2: Could not reliably determine the server's fully
qualified domain name, using 127.0.1.1 for ServerName
. ok
```

---

### 8.1.4 D.- Configuración de WebSafet en los directorio (intranet y media)

#### 1° PRIMER PASO

- Desde la consola de comando como usuario (**normal**), nos ubicamos en el directorio (**/var/www/**) con el comando (**cd**), como se muestra en el siguiente código:

```
$ cd /var/www/
```

#### 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**normal**), abrimos el archivo (**safetconfig.py**) que esta en el dirctorio (**intranet**) con un editor de texto (**gedit,vi,nano etc**), como se muestra en el siguiente código:

```
/var/www/ $ gedit intranet/safetconfig.py
```

---

**Nota: Vemos el contenido del archivo (safetconfig.py), la cual hay que cambiarle las url al de nuestra maquina:**

```
#-*- coding: utf-8 -*-
#
# websafet
# archivo para normalizacion de directorios
```

```
#y rutas URL utilizadas en websafet

# Cambiar url:

SERVER_URL      = u"http://192.168.12.175/intranet"
LOGIN_URL       = u"http://192.168.12.175/intranet/login"
TEMPLATES_PATH  = u"/var/www/media/templates"
TEMPLATES_URL   = u"http://192.168.12.175/media/templates/"
HOMESAFET_PATH  = u"/home/cenditel"
MEDIA_PATH      = u"/var/www/media"
MEDIA_URL       = u"http://192.168.12.175/media"

TEMPLATES_PATH + u'/' + u'index.html',
TEMPLATES_PATH + u'/' + u'consoperations.html',
TEMPLATES_PATH + u'/' + u'resultgeneration.html',
TEMPLATES = (
    TEMPLATES_PATH + u'/' + u'listgeneration.html',
    TEMPLATES_PATH + u'/' + u'safetsimple.html',
    TEMPLATES_PATH + u'/' + u'formoperations.html',
    TEMPLATES_PATH + u'/' + u'formgeneration.html',
    TEMPLATES_PATH + u'/' + u'consgeneration.html',
    TEMPLATES_PATH + u'/' + u'resultgeneration.html',
    TEMPLATES_PATH + u'/' + u'login.html',
    TEMPLATES_PATH + u'/' + u'credits.html',
    TEMPLATES_PATH + u'/' + u'menusimple.html', #11
    TEMPLATES_PATH + u'/' + u'listgenerationc.html', #12
    TEMPLATES_PATH + u'/' + u'plugintemplate.html', #13
    TEMPLATES_PATH + u'/' + u'register.html', #14
    TEMPLATES_PATH + u'/' + u'resultgenerationm.html', #15
    TEMPLATES_PATH + u'/' + u'listgenerationg.html', #16
)
JS_SAFETPROCESSCONSOLE_HEAD = u"""
<style type="text/css">
.svgdiv { border: 1px solid #3c8243; }
#svgintr0 { float: right; width: 150px; height: 150px;
margin-right: 30px; background: #fff; border: 1px solid #3c8243; }
.svgsample { float: left; width: 46%; margin: 1%;
overflow: scroll; border: 1px solid #3c8243; padding: 5px; }
.drawOpt { float: left; width: 25%; }
```

---

### 3° TERCER PASO

- Desde la consola de comando como usuario (**normal**), en los directorios (**intranet** y **media**) remplazamos el (**/home/cenditel/**) por el (**/home/usuario**) de nuestra maquina de trabajo y remplazamos (**http://192.168.12.175/**) por la de nuestro (**http://localhost/**), con el comando **rp -R (“ruta” “remplazo”)** \*, como se muestra en el siguiente código:

```
/var/www/ $ rpl -R "/home/cenditel/" "/home/usuario/" *
/var/www/ $ rpl -R "http://192.168.12.175/" "http://localhost/" *
```

---

**Nota: Con rpl:**

- El “localhost” se tiene que sustituir por el **nombre o IP** del servidor en el caso que se vaya a utilizar de forma **remota**, es decir desde otra máquina.
- 

**4° CUARTO PASO**

- Desde la consola de comando como usuario (**normal**), observamos los cambios en el archivo (safetconfig.py) y demás, que esta en el directorio (**intranet**) con un editor de texto (**gedit,vi,nano etc**), como se muestra en el siguiente código:

```
/var/www/ $ gedit intranet/safetconfig.py
```

---

**Nota: Vemos los cambios realizados:**

```
#-- coding: utf-8 --

# websafet
# archivo para normalizacion de directorios
#y rutas URL utilizadas en websafet

# Cambiar realizados:

SERVER_URL      = u"http://localhost/intranet"
LOGIN_URL       = u"http://localhost/intranet/login"
TEMPLATES_PATH  = u"/var/www/media/templates"
TEMPLATES_URL   = u"http://localhost/media/templates/"
HOMESAFET_PATH  = u"/home/usuario"
MEDIA_PATH       = u"/var/www/media"
MEDIA_URL        = u"http://localhost/media"

TEMPLATES = (
    TEMPLATES_PATH + u'/' + u'index.html',
    TEMPLATES_PATH + u'/' + u'consoperations.html',
    TEMPLATES_PATH + u'/' + u'resultgeneration.html',
```

---

**5° QUINTO PASO**

- Desde la consola de comando como usuario (**root**), damos permisos de acceso al directorio (**/var/www/**) con el comando **chown -R (www-data:www-data)**, como se muestra en el siguiente código:

```
# chown -R www-data:www-data /var/www/
```

### 8.1.5 E.- Configuración de WebSafet en el directorio (\$HOME/.safet)

#### 1° PRIMER PASO

- Desde la consola de comando como usuario (**normal**), abrimos con un editor de texto (**gedit,vi,nano etc**) el archivo (**safet.conf**) que encuentra en el directorio (**\$HOME/.safet/**), insertamos el siguiente código para poder ver los datos en la tabla de manera **Web**:

[Result]

```
result.output.type = VariableData
```

---

**Nota:** El código se coloca debajo de los valores para calculo de estadísticas, como se muestra a continuación:

```
#Valores para el calculo de estadísticas  
[Stats]
```

```
#Activar la recoleccion de estadisticas  
stats.actived = off  
# Fecha de inicio de calculo de estadística,  
# * significa que no hay fecha colocada  
stats.fromdate = *  
#Fecha de fin de calculo de estadística,  
#* significa que no hay fecha colocada  
stats.todate = *
```

[Result]

```
result.output.type = VariableData
```

---

#### 2° SEGUNDO PASO

- Desde la consola de comando como usuario (**root**), damos permisos de acceso al directorio (**<HOME>.safet/**) y el directorio (**<HOME>tmp**) con el comando **chown -R (www-data:www-data)**, como se muestra en el siguiente código:

```
# chown -R www-data:www-data /home/cenditel/.safet  
# chown -R www-data:www-data /home/cenditel/tmp
```

#### 3° TERCER PASO

- Desde la consola de comando como usuario (**root**), reiniciamos el **servidor apache2** como se muestra en el siguiente código:

```
# /etc/init.d/apache2 restart
```

## 8.1.6 F.- Probamos WebSafet en nuestro servidor web

### 1° PRIMER PASO

- Abrimos nuestro navegador **Web** y colocamos la **url** (<http://localhost/intranet/>), como se muestra en la siguiente *Figura 345: Navegador Web*:

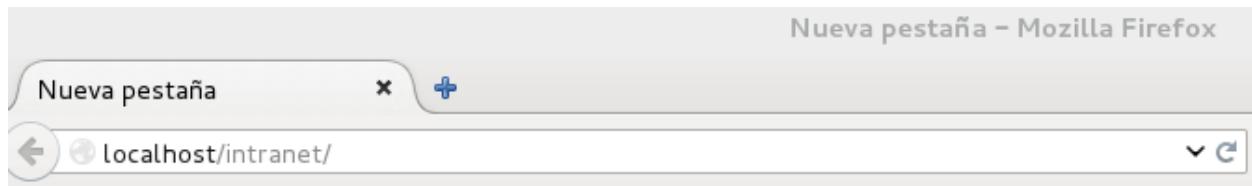


Figura 8.3: **Figura 345: Navegador Web**

### 2° SEGUNDO PASO

- Damos click a (**Entrar a safet**), como se muestra en la siguiente *Figura 346: Entrar a safet*:

### 3° PRIMER PASO

- Estamos en la entrada del sistema **safet**, como se muestra en la siguiente *Figura 347: Autenticación*:

#### **Nota: Plantillas:**

**1.-** Las plantillas por defecto corresponden a los utilizados por la **Fundación Cenditel**. Para adaptarlas a otra organización, debe reemplazar o modificar las plantillas que se encuentran el directorio (**/var/www/media/templates**), incluyendo las imágenes vista en el despliegue **web**.

**Por ejemplo la imagen de título del sistema se encuentra en la ruta:**

`$ /var/www/media/templates/images/cintillo.png`

**2.-** La interfaz **web** utiliza la librería jquery javascript ([jquery.org](http://jquery.org)) y algunos de sus pluigns (**componentes**), por lo tanto es importante en caso de cambios, considerar que debe mantener la dependencia.

## Documentación de PySafet, Publicación

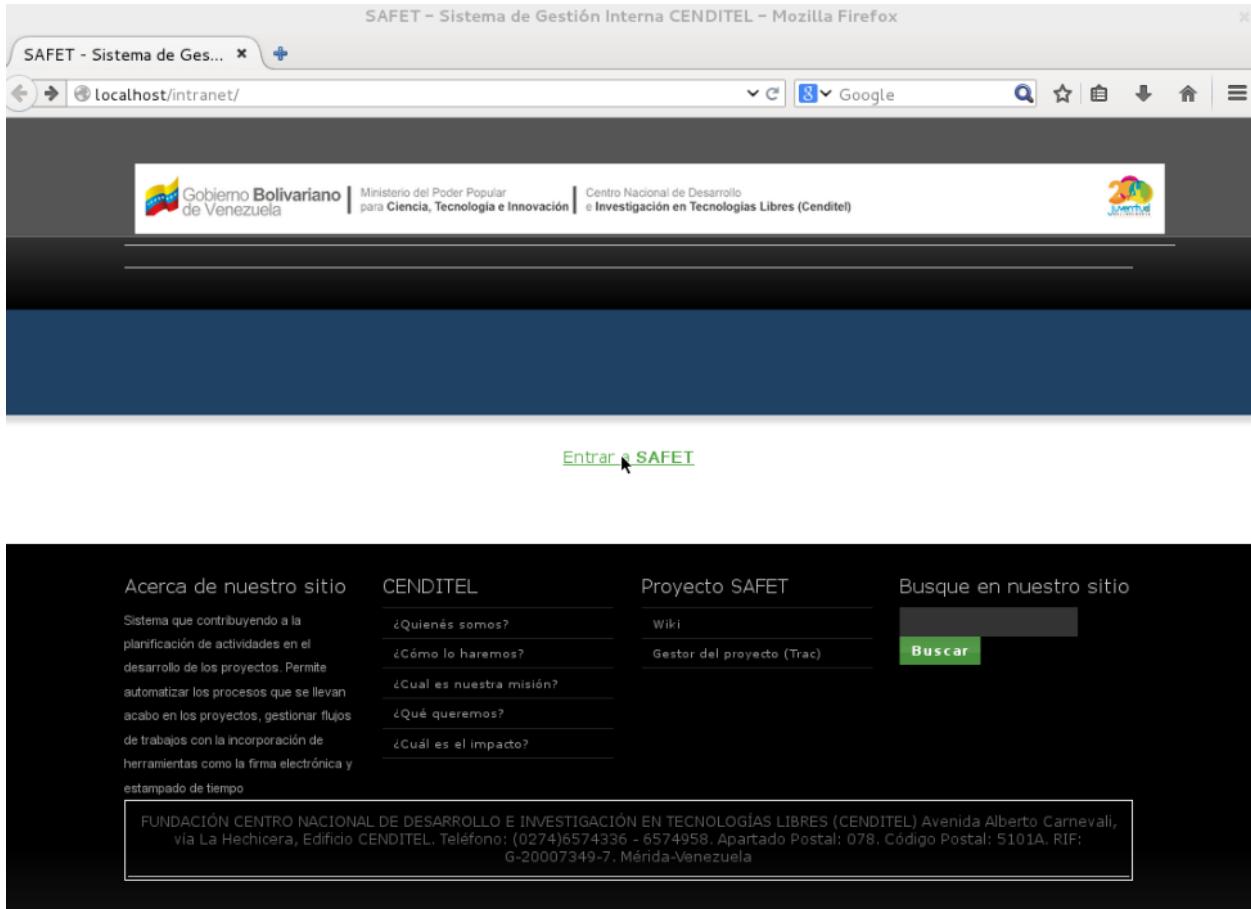


Figura 8.4: **Figura 346: Entrar a safet**

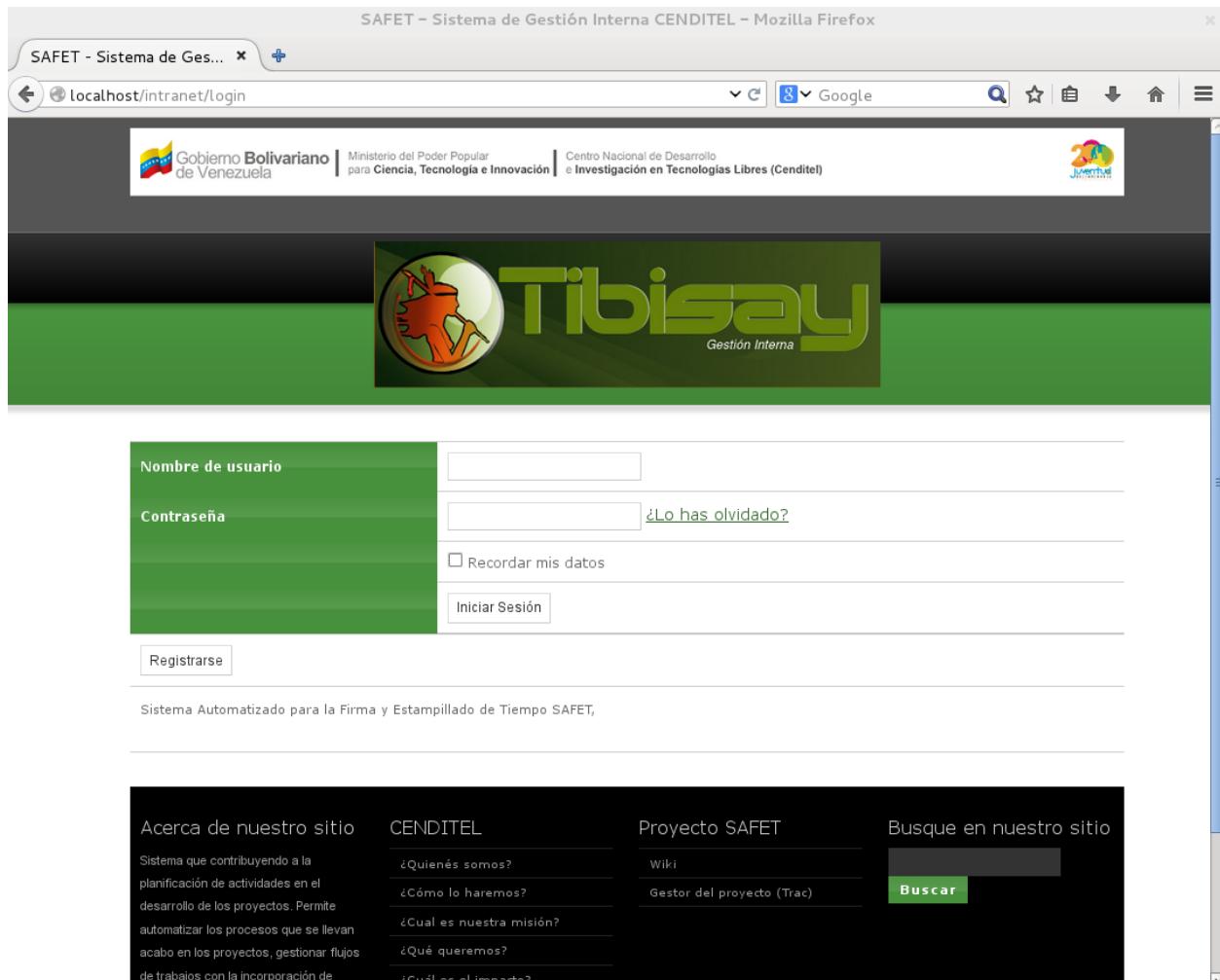


Figura 8.5: Figura 347: Autenticación

## 8.2 Ejecución de operaciones CRUD+

### 8.2.1 A.- Entramos al sistema y buscamos la opción (Ingresar o modificar información) o (Gestión)

#### 1° PRIMER PASO

- Abrimos nuestro navegador **Web** y colocamos la **url** (<http://localhost/intranet/>), como se muestra en la siguiente *Figura 348: Navegador Web*:

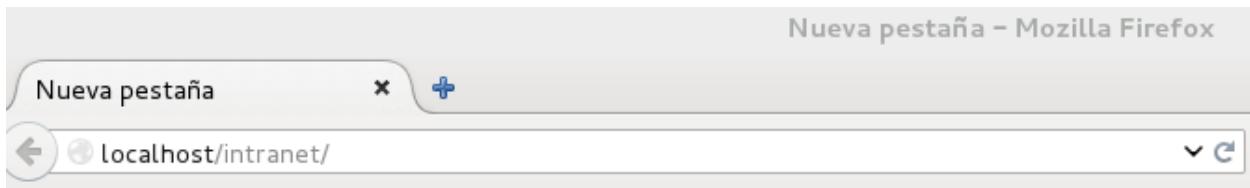


Figura 8.6: **Figura 348: Navegador Web**

#### 2° SEGUNDO PASO

- Damos click a **Entrar a safet**, como se muestra en la siguiente *Figura 349: Entrar a safet*:

#### 3° PRIMER PASO

- Colocamos el usuario/password (**admin/admin**) y pulsamos en el botón (**Iniciar Sesión**) para entrar al sistema, como se muestra en la siguiente *Figura 350: Autenticación de Safet*:

#### 4° CUARTO PASO

- Pulsamos el la opción (**Ingreso o modificar información**), como se muestra en la siguiente *Figura 351: Operaciones*:

### 8.2.2 B.- PRIMERA OPERACIÓN GRUD+ (Agregar o registrar producto)

#### 1° PRIMER PASO

- Pulsamos en la primera operación (**Agregar producto**), como se muestra en la siguiente *Figura 352: Operación (Agregar producto)*:

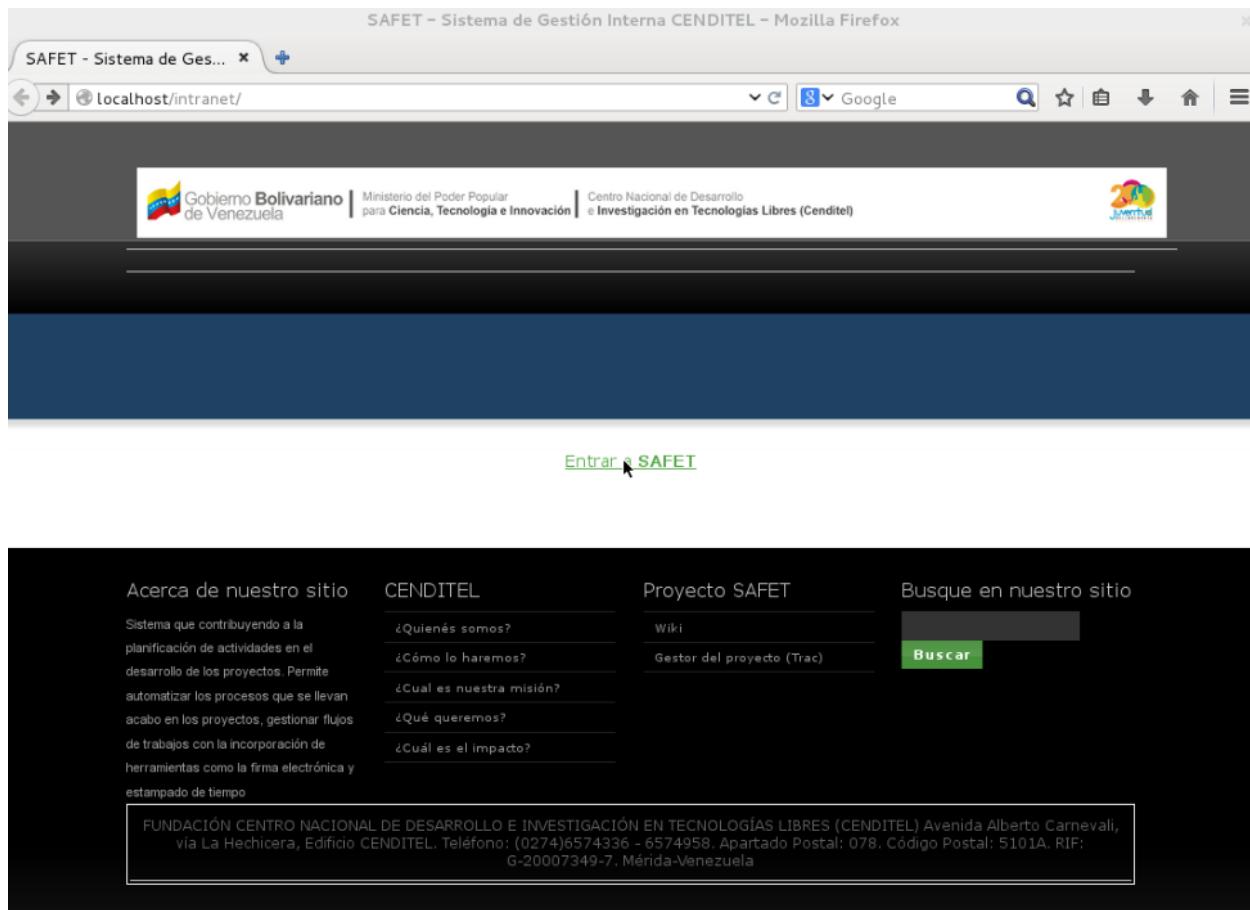


Figura 8.7: **Figura 349: Entrar a safet**

## Documentación de PySafet, Publicación

Nombre de usuario: admin

Contraseña: .....

Recordar mis datos

Iniciar Sesión

Registrarse

Sistema Automatizado para la Firma y Estampillado de Tiempo SAFET,

Figura 8.8: Figura 350: Autenticación de Safet

### 2° SEGUNDO PASO

- Agregamos el nombre del producto por ejemplo (**Jabón ariel**) y pulsamos en el botón (**Enviar**) para finalizar con la operación , como se muestra en la siguiente *Figura 353: botón (Enviar)*:

### 3° TERCER PASO

**Nota:** Si deseas agregar otro producto, solo pulsamos en la opción (**Regresar a formulario**), como se muestra en la siguiente *Figura 354: Regresar a formularios*:

- 
- **Se nos devuelve al formulario**, Obseven la siguiente *Figura 355: Formulario*:
-

Gobierno Bolivariano de Venezuela | Ministerio del Poder Popular para Ciencia, Tecnología e Innovación | Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) | Juventud

INICIO    GESTIÓN    CONSULTAS    CRÉDITOS    CERRAR SESIÓN    Administración - admin/Administrador

## Tibisay - Opciones Generales

Sistema Automatizado para la Firma Electrónica y el Estampillado de tiempo SAFET

Usted desea realizar una de las siguientes opciones:

[Ingreso o modificación de información](#)

[Generar reportes o gráficos de consulta](#)

[Cerrar Sesión](#)

**Acerca de nuestro sitio**  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos

**CENDITEL**  
[¿Quiénes somos?](#)  
[¿Cómo lo haremos?](#)  
[¿Cuál es nuestra misión?](#)  
[¿Qué queremos?](#)

**Proyecto SAFET**  
[Wiki](#)  
[Gestor del proyecto \(Trac\)](#)

**Busque en nuestro sitio**  
  
**Buscar**

Figura 8.9: Figura 351: Operaciones

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the CENDITEL logo, and a Youth logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an 'Administración - admin/Administrador' link. The main content area has a blue header with a pencil icon and the title 'Tibisay - Gestión'. Below this, the 'Productos' (Products) page is displayed, featuring a list of actions: Agregar producto, Realizar pedido producto, Modificar datos producto, Siguiente estado producto, and Borrar producto. The footer contains sections for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and a search bar.

Acerca de nuestro sitio

Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de

CENDITEL

¿Quiénes somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

Proyecto SAFET

Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio

Buscar

Figura 8.10: Figura 352: Operación (Agregar producto)

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Cenditel logo, and a Juventud logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right, it says 'Administración - admin/Administrador'. The main content area has a dark blue header with a magnifying glass icon and the text 'Tibisay - Gestión'. Below this, a form is displayed for adding a product. It has fields for 'Nombre \*' (with 'Jabon ariel' entered) and two buttons: 'Enviar' and 'Limpiar'. At the bottom, there is a footer section with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio'.

Acerca de nuestro sitio	CENDITEL	Proyecto SAFET	Busque en nuestro sitio
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan	<a href="#">¿Quiénes somos?</a> <a href="#">¿Cómo lo haremos?</a> <a href="#">¿Cuál es nuestra misión?</a>	<a href="#">Wiki</a> <a href="#">Gestor del proyecto (Trac)</a>	<input type="text"/> <input type="button" value="Buscar"/>

Figura 8.11: Figura 353: botón (Enviar)

The screenshot shows a web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the text "Tibisay - Resultado de la operación" and a small icon of a globe with a flame. Below this, there is a green button with a checkmark icon and the text "Se realizó la operación correctamente!". At the bottom of the page, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and a search bar.

Acerca de nuestro sitio

Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de herramientas como la firma electrónica

CENDITEL

¿Quién es somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

Proyecto SAFET

Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio

Buscar

Figura 8.12: Figura 354: Regresar a formularios

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header with the Venezuelan Government logo, the text 'Gobierno Bolivariano de Venezuela', 'Ministerio del Poder Popular para Ciencia, Tecnología e Innovación', 'Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)', and the 'Juventud' logo. Below the header, a navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', 'CERRAR SESIÓN', and 'Administración - admin/Administrador'. The main content area has a dark blue header with the text 'Tibisay - Gestión' and a small globe icon. Below this, a form titled 'Agregar producto' contains a text input field labeled 'Nombre \*' with the placeholder 'Escribir el nombre del producto'. There are two buttons at the bottom of the form: 'Enviar' and 'Limpiar'. At the bottom of the page, there is a footer section with four columns: 'Acerca de nuestro sitio' (describing the system as a planning tool for projects), 'CENDITEL' (with links to '¿Quiénes somos?', '¿Cómo lo haremos?', '¿Cuál es nuestra misión?', '¿Qué queremos?', and '¿Cuál es el impacto?'), 'Proyecto SAFET' (with links to 'Wiki' and 'Gestor del proyecto (Trac)'), and 'Busque en nuestro sitio' (with a search input field and a green 'Buscar' button).

Figura 8.13: Figura 355: Formulario

### 4° CUARTO PASO

- Para salir de la operación pulsamos en el menu en la opción (**Gestión**), como se muestra en la siguiente *Figura 356: Opción (Gestión)*:



Figura 8.14: Figura 356: Opción (Gestión)

### 8.2.3 C.- SEGUNDA OPERACIÓN GRUD+ (Realizar pedido de un producto)

#### 1° PRIMER PASO

- Pulsamos en la primera operación (**Realizar pedido producto**), como se muestra en la siguiente *Figura 357: Operación (Realizar pedido producto)*:

#### 2° SEGUNDO PASO

- Seleccionamos el producto a pedir en este caso **Jabon ariel**, como se muestra en la siguiente *Figura 358: Selección de producto*:

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cendite), and the Juventud Venezuela logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an 'Administración - admin/Administrador' link. The main content area has a dark blue header with a pencil icon and the text 'Tibisay - Gestión'. Below this, the 'Productos' (Products) page is displayed, featuring a list of actions: Agregar producto, Realizar pedido producto, Modificar datos producto, Siguiente estado producto, and Borrar producto. The bottom of the page contains footer sections for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and a search bar.

Agregar producto  
Realizar pedido producto  
Modificar datos producto  
Siguiente estado producto  
Borrar producto

Acerca de nuestro sitio  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de

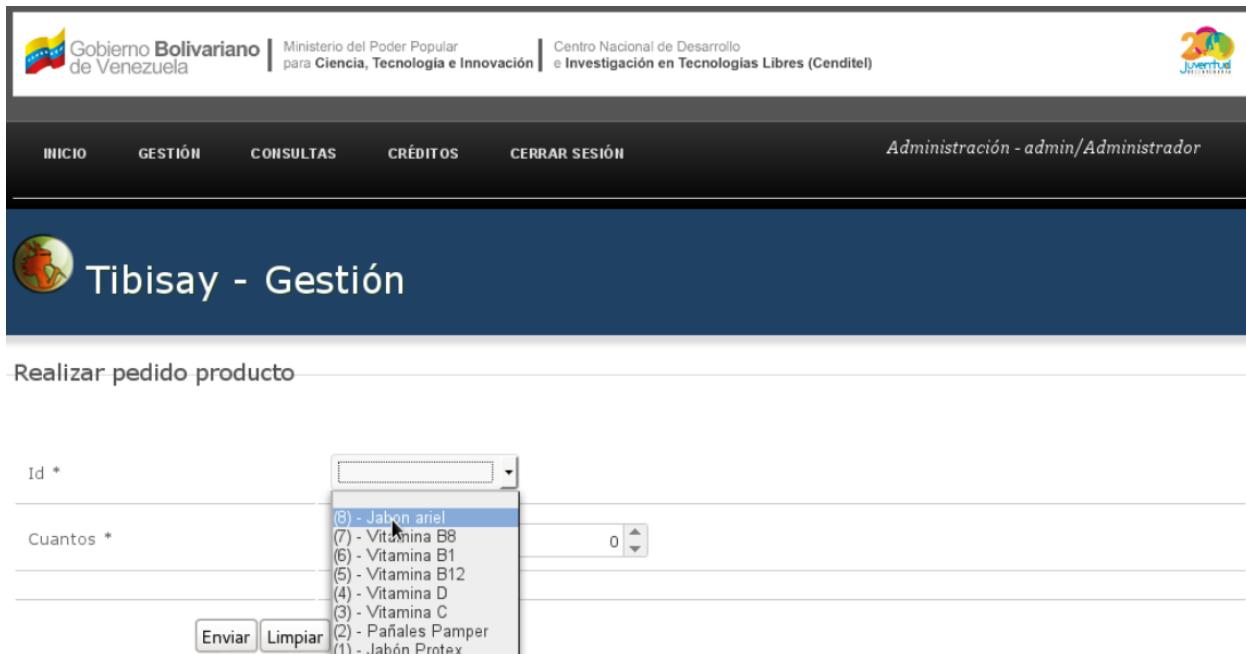
CENDITEL  
¿Quiénes somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

Proyecto SAFET  
Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio  
Buscar

Figura 8.15: Figura 357: Operación (Realizar pedido producto)

## Documentación de PySafet, Publicación



The screenshot shows a web-based application interface. At the top, there is a header with the logo of the Government of Venezuela, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel). Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Administración - admin/Administrador". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small icon. Below this, a sub-header says "Realizar pedido producto". The form contains fields for "Id \*" (with a dropdown menu open showing a list of products), "Cuantos \*" (with a numeric input field set to 0), and two buttons "Enviar" and "Limpiar". A dropdown menu is open over the "Id \*" field, listing various items such as Jabón ariel, Vitamina B8, Vitamina B1, Vitamina B12, Vitamina D, Vitamina C, Pañales Pamper, and Jabón Protex. The number 8 is highlighted in the list. At the bottom of the page, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and a search bar.

Figura 8.16: Figura 358: Selección de producto

### 3° TERCER PASO

- Agreamos la cantidad a **pedir** por ejemplo (300) ,como se muestra en la siguiente *Figura 359: Cantidad a pedir:*

### 4° CUARTO PASO

- Pulsamos en el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *Figura 360: botón (Enviar):*

### 5° QUINTO PASO

**Nota:** Si deseas realizar otro pedido, solo pulsamos en la opción (**Regresar a formulario**),

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. To the right of the navigation bar, it says 'Administración - admin/Administrador'. The main content area has a blue header with a circular icon and the text 'Tibisay - Gestión'. Below this, a form titled 'Realizar pedido producto' contains fields for 'Id \*' (with a dropdown menu showing '(8) - Jabon ariel'), 'Cuantos \*' (with a text input field containing '300' and an upward arrow), and two buttons 'Enviar' and 'Limpiar'. At the bottom of the page, there is a footer with sections for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio'.

Acerca de nuestro sitio  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan

CENDITEL  
¿Quienés somos?  
¿Cómo lo faremos?  
¿Cuál es nuestra misión?

Proyecto SAFET  
Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio  
Buscar

Figura 8.17: Figura 359: Cantidad a pedir

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the text "Gobierno Bolivariano de Venezuela", "Ministerio del Poder Popular para Ciencia, Tecnología e Innovación", "Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)", and the "Juventud" logo. Below the header, a navigation bar includes links for "INICIO", "GESTIÓN", "CONSULTAS", "CRÉDITOS", "CERRAR SESIÓN", and "Administración - admin/Administrador". The main content area has a blue header with the text "Tibisay - Gestión" and a small icon. Below this, a form titled "Realizar pedido producto" contains fields for "Id \*" (with a dropdown menu showing "(8) - Jabon ariel"), "Cuantos \*" (with a numeric input field showing "300" and up/down arrows), and two buttons: "Enviar" and "Limpiar". At the bottom, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio", along with various links and a search bar.

Realizar pedido producto

Id \* (8) - Jabon ariel

Cuantos \* 300

Enviar Limpiar

Acerca de nuestro sitio

Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan

CENDITEL

¿Quienés somos?  
¿Cómo lo haremos?  
¿Cual es nuestra misión?

Proyecto SAFET

Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio

Buscar

Figura 8.18: Figura 360: botón (Enviar)

como se muestra en la siguiente *Figura 361: Regresar a formulario:*

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the text "Gobierno Bolivariano de Venezuela", "Ministerio del Poder Popular para Ciencia, Tecnología e Innovación", "Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)", and the "Joven" logo. Below the header is a navigation bar with links for "INICIO", "GESTIÓN", "CONSULTAS", "CRÉDITOS", and "CERRAR SESIÓN". To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the title "Tibisay - Resultado de la operación". Below the header, there is a green box containing a checkmark icon and the text "Se realizó la operación correctamente!". At the bottom of the page, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.19: Figura 361: Regresar a formulario

■ **Se nos devuelve al formulario,** Obseven la siguiente *Figura 362: Formulario:*

## 6° SEXTO PASO

- Para salir de la operación pulsamos en el menu en la opción (**Gestión**), como se muestra en la siguiente *Figura 363: Opción (Gestión)*:

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, and the Cenditel logo. The navigation menu includes 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', 'CERRAR SESIÓN', and 'Administración - admin/Administrador'. Below the header, a blue banner features the 'Tibisay' logo and the text 'Realizar pedido producto'. The main form has fields for 'Id \*' (with a dropdown menu), 'Cuantos \*' (with a numeric input field showing '0' and up/down arrows), and buttons for 'Enviar' and 'Limpiar'. At the bottom, there is a footer with sections for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio'.

**Acerca de nuestro sitio**  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan

**CENDITEL**  
¿Quienés somos?  
¿Cómo lo haremos?  
¿Cual es nuestra misión?

**Proyecto SAFET**  
Wiki  
Gestor del proyecto (Trac)

**Busque en nuestro sitio**  
 **Buscar**

Figura 8.20: Figura 362: Formulario

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN (which is highlighted), CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the text "Tibisay - Resultado de la operación" and a small icon of a person with a checkmark. Below this, a green rounded rectangle contains the message "Se realizó la operación correctamente!" with a checkmark icon. At the bottom left, there is a link "Regresar al Formulario". The footer section contains four columns: "Acerca de nuestro sitio" (with text about contributing to project planning), "CENDITEL" (with links to Who we are, How we do it, Our mission, What we want, and Impact), "Proyecto SAFET" (with links to Wiki and Project manager (Trac)), and "Busque en nuestro sitio" (with a search input field and a "Buscar" button).

Figura 8.21: Figura 363: Opción (Gestión)

### 8.2.4 D.- TERCERA OPERACIÓN GRUD+ (Modificar datos del producto)

#### 1° PRIMER PASO

- Pulsamos en la primera operación (**Modificar datos producto**), como se muestra en la siguiente *Figura 364: Operación (Modificar datos producto)*:

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Juventud 2030 logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Administración - admin/Administrador". The main content area has a blue header with a globe icon and the text "Tibisay - Gestión". Below this, there is a section titled "Productos" with a pencil icon. A sidebar on the left lists options: Agregar producto, Realizar pedido producto, Modificar datos producto (highlighted in red), Siguiente estado producto, and Borrar producto. At the bottom, there is a footer with sections for Acerca de nuestro sitio, CENDITEL, Proyecto SAFET, and Busque en nuestro sitio.

<b>Acerca de nuestro sitio</b> Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de	<b>CENDITEL</b> ¿Quiénes somos? ¿Cómo lo haremos? ¿Cuál es nuestra misión? ¿Qué queremos? ¿Cuál es el impacto?	<b>Proyecto SAFET</b> Wiki Gestor del proyecto (Trac)	<b>Busque en nuestro sitio</b> <input type="text"/> <b>Buscar</b>
---	---	---	---

Figura 8.22: Figura 364: Operación (Modificar datos producto)

#### 2° SEGUNDO PASO

- Seleccionamos el **id** del producto a modificar por ejemplo ((8) **Jabón ariel**), como se muestra en la siguiente *Figura 365: Seleccionar producto*:

**Nota:** Al seleccionar el producto automáticamente se nos cargara los datos del producto. Esto se debe a los parametros () y () se archivo destrac.xml en el directorio (<HOME>.safet/input/). Observen la siguiente *Figura 366: Datos cargados en el formulario*:

The screenshot shows a web-based application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the 20th Anniversary of the Youth logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with the text "Administración - admin/Administrador". The main title "Tibisay - Gestión" is displayed above a form titled "Modificar datos producto". The form contains fields for Id \* (with a dropdown menu open showing options like "8 - Jabón ariel", "7 - Vitamina B8", etc.), Nombre, Cuantos, Categoría (with a dropdown menu open showing options like "8 - Jabón ariel", "7 - Vitamina B8", etc.), and Status. At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Figura 8.23: Figura 365: Seleccionar producto

### Modificar datos producto

Id *	8 - Jabon ariel
Nombre	Jabon ariel
Cuantos	300
Categoría	no_clasificado
Status	Pedido
<a href="#">Enviar</a>	<a href="#">Limpiar</a>

Figura 8.24: *Figura 366: Datos cargados en el formulario*

### 3° TERCER PASO

- Modificamos los campos que deseamos y pulsamos en el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *Figura 367: botón (Enviar)*:

### 4° CUARTO PASO

---

**Nota:** Si deseas realizar otro pedido, solo pulsamos en la opción (**Regresar a formulario**), como se muestra en la siguiente *Figura 368: Regresar a formulario*:

- Se nos devuelve al formulario, Obsevén la siguiente *Figura 369: Formulario*:

The screenshot shows a web application interface for managing products. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session indicator. The main title "Tibisay - Gestión" is displayed above a form titled "Modificar datos producto". The form contains fields for Id \* (set to 8 - Jabón Ace), Nombre (Jabón Ace), Cuantos (100), Categoría (Lavar ropa), and Status (Pedido). At the bottom of the form are two buttons: "Enviar" (highlighted with a yellow border) and "Limpiar". The footer of the page includes links for Acerca de nuestro sitio, CENDITEL, Proyecto SAFET, and Busque en nuestro sitio, along with a "Wiki" link.

Id *	8 - Jabón Ace
Nombre	Jabón Ace
Cuantos	100
Categoría	Lavar ropa
Status	Pedido

**Enviar** **Limpiar**

Acerca de nuestro sitio      CENDITEL      Proyecto SAFET      Busque en nuestro sitio  
Sistema que contribuyendo a la      ¿Quiénes somos?      Wiki

Figura 8.25: Figura 367: botón (Enviar)

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an 'Administración' link. The main content area has a green header with the title 'Tibisay - Resultado de la operación'. A green message box contains the text 'Se realizó la operación correctamente!' (The operation was successfully performed!). Below this, a link 'Regresar al Formulario' (Return to the form) is visible. The footer contains sections for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and a search bar.

**Acerca de nuestro sitio**

Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de herramientas como la firma electrónica

**CENDITEL**

¿Quién es somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

**Proyecto SAFET**

Wiki  
Gestor del proyecto (Trac)

**Busque en nuestro sitio**

**Buscar**

Figura 8.26: Figura 368: Regresar a formulario

The screenshot shows a web application interface for managing products. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Cenditel logo, and a Youth Ministry logo. The navigation menu includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. The current user is identified as 'Administración - admin/Administrador'. Below the header, the title 'Tibisay - Gestión' is displayed next to a globe icon. The main content area is titled 'Modificar datos producto'. It contains a form with fields for 'Id \*' (with a dropdown menu), 'Nombre' (with placeholder 'Escribir el nombre'), 'Cuantos' (with placeholder 'Escribir cantidad de producto'), 'Categoria' (with placeholder 'Escribir la categoría'), and 'Status' (with placeholder 'Escribir status del producto'). At the bottom of the form are two buttons: 'Enviar' and 'Limpiar'. At the very bottom of the page, there is a footer with sections for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio', along with links for 'Wiki' and 'Gestor del proyecto (Trac)'. There is also a 'Buscar' button.

Figura 8.27: Figura 369: Formulario

### 5° QUINTO PASO

- Para salir de la operación pulsamos en el menu en la opción (**Gestión**), como se muestra en la siguiente *Figura 370: Opción (Gestión)*:

The screenshot shows a web application interface. At the top, there's a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN (which is highlighted in a black box), CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the text "Tibisay - Resultado de la operación" and a small icon of a person holding a torch. Below this, a green box contains a checkmark icon and the text "Se realizó la operación correctamente!". At the bottom of the page, there are footer sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and a search bar labeled "Busque en nuestro sitio".

Figura 8.28: **Figura 370: Opción (Gestión)**

### 8.2.5 E.- CUARTA OPERACIÓN GRUD+ (Siguiente estado del producto)

#### 1° PRIMER PASO

- Pulsamos en la primera operación (**Siguiente estado producto**), como se muestra en la siguiente *Figura 371: Operación (Siguiente estado producto)*:

#### 2° SEGUNDO PASO

- Seleccionamos el **id** del producto cambiar de **estado**, como se muestra en la siguiente *Figura 372: Seleccionar producto*:

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Juventud 20 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the navigation bar, it says "Administración - admin/Administrador". The main content area has a dark blue header with a globe icon and the text "Tibisay - Gestión". Below this, the "Productos" page is displayed, featuring a pencil icon and the heading "Productos". A sidebar on the left lists actions: Agregar producto, Realizar pedido producto, Modificar datos producto, Siguiente estado producto (with a cursor arrow pointing to it), and Borrar producto. At the bottom, there is a footer with sections for Acerca de nuestro sitio, CENDITEL, Proyecto SAFET, and Busque en nuestro sitio.

**Acerca de nuestro sitio**  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de

**CENDITEL**  
¿Quiénes somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

**Proyecto SAFET**  
Wiki  
Gestor del proyecto (Trac)

**Busque en nuestro sitio**  
  
**Buscar**

Figura 8.29: Figura 371: Operación (Siguiente estado producto)

## Documentación de PySafet, Publicación

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel), and the Juventud 20 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session indicator "Administración - admin/Administrador". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small icon. Below this, a form section titled "Siguiente estado producto" contains fields for "Id \*" and "Estado producto \*". A dropdown menu is open, listing various product states: (8) Jabón Ace, (7) Vitamina B8, (6) Vitamina B1, (5) Vitamina B12, (4) Vitamina D, (3) Vitamina C, (2) Pañales Pamper, and (1) Jabón Protex. At the bottom of the form are two buttons: "Enviar" and "Limpiar". In the footer, there are sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio", each with links to further information. There is also a search bar with a "Buscar" button.

Figura 8.30: Figura 372: Seleccionar producto

### 3° TERCER PASO

- Seleccionamos el **estado del producto** por el cual puede pasar o ir. En este caso el producto esta en pedido solo puede ir a (Disponible o Por\_llegar), esto quiere decir que el producto ya llego o esta por llegar. En este ejemplo seleccionamos el estado (**Disponible**), mencionando que el producto que pedimos (**Jabón Ariel**) ya llego, como se muestra en la siguiente *Figura 373: Seleccionar estado*:

### 4° CUARTO PASO

- Pulsamos el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *Figura 374: botón (Enviar)*:

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. The navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', 'CERRAR SESIÓN', and 'Administración - admin/Administrador'. Below the header, the main title 'Tibisay - Gestión' is displayed next to a circular icon. The main content area has a dark blue header with the text 'Siguiente estado producto'. Below this, there are two input fields: 'Id \*' with the value '(8)Jabón Ace' and 'Estado producto \*' with a dropdown menu open. The dropdown menu contains two options: 'Disponible' (highlighted in blue) and 'Pendiente'. At the bottom of the form are two buttons: 'Enviar' and 'Limpiar'. At the very bottom of the page, there is a footer with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio', along with a 'Wiki' link.

Figura 8.31: Figura 373: Seleccionar estado

The screenshot displays a web-based application interface for product management. At the top, there is a header bar with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation (CENDITEL) logo, and the Youth Ministry logo. Below the header, a navigation menu includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it shows the user is logged in as 'Administración - admin/Administrador'. The main content area has a blue header titled 'Tibisay - Gestión' featuring a small icon of a hand holding a plant. Below this, there is a form for managing a product entry. The form fields include 'Id \*' with the value '(8)Jabón Ace', 'Estado producto \*' set to 'Disponible', and two buttons at the bottom: 'Enviar' (highlighted in yellow) and 'Limpiar'. At the bottom of the page, there is a footer section with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and a search bar labeled 'Busque en nuestro sitio'.

Siguiente estado producto

Id \* (8)Jabón Ace

Estado producto \* Disponible

Enviar Limpiar

Acerca de nuestro sitio  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan

CENDITEL  
¿Quiénes somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?

Proyecto SAFET  
Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio

Figura 8.32: Figura 374: botón (Enviar)

## 5° QUINTO PASO

**Nota:** Si deseas realizar otro pedido, solo pulsamos en la opción (**Regresar a formulario**), como se muestra en la siguiente *Figura 375: Regresar a formulario*:

The screenshot shows a web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Administración - admin/Administrador". The main content area has a green header bar with the text "Tibisay - Resultado de la operación". Below this, there is a green rounded rectangle containing a checkmark icon and the text "Se realizó la operación correctamente!". At the bottom of the page, there is a footer section with four columns: "Acerca de nuestro sitio" (with text about contributing to project planning), "CENDITEL" (with links to Who we are, How we do it, Our mission, What we want, and Impact), "Proyecto SAFET" (with links to Wiki and Project manager (Trac)), and "Busque en nuestro sitio" (with a search input field and a "Buscar" button). A red arrow points to the "Regresar al Formulario" link at the bottom left of the page.

Figura 8.33: Figura 375: Regresar a formulario

**Se nos devuelve al formulario,** Obseven la siguiente *Figura 376: Formulario*:

## 6° SEXTO PASO

- Para salir de la operación pulsamos en el menu en la opción (**Gestión**), como se muestra en la siguiente *Figura 377: Opción (Gestión)*:

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header bar with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, the Cenditel logo, and a 'Juventud' logo. Below the header, a navigation menu includes 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', 'CERRAR SESIÓN', and 'Administración - admin/Administrador'. The main title 'Tibisay - Gestión' is displayed above a form. The form contains fields for 'Id \*' (with a dropdown menu), 'Estado producto \*' (with a dropdown menu), and two buttons: 'Enviar' and 'Limpiar'. At the bottom of the page, there is a footer section with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and a search bar.

Siguiente estado producto

Id \*

Estado producto \*

Enviar Limpiar

Acerca de nuestro sitio  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan acabo en los proyectos, gestionar flujos

CENDITEL  
¿Quienés somos?  
¿Cómo lo haremos?  
¿Cual es nuestra misión?  
¿Qué queremos?

Proyecto SAFET  
Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio  
Buscar

Figura 8.34: Figura 376: Formulario

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN (which is highlighted), CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the title "Tibisay - Resultado de la operación". Below this, a green box contains the message "Se realizó la operación correctamente!" (The operation was performed correctly!). At the bottom left, there is a link "Regresar al Formulario". The footer section contains four columns: "Acerca de nuestro sitio" (About our site) with text about project management; "CENDITEL" with links to Who we are, How we do it, Our mission, What we want, and Impact; "Proyecto SAFET" with links to Wiki and Project manager (Trac); and "Busque en nuestro sitio" (Search our site) with a search input field and a "Buscar" button.

Figura 8.35: Figura 377: Opción (Gestión)

### 8.2.6 F.- QUINTA OPERACIÓN GRUD+ (Borrar producto del inventario)

#### 1° PRIMER PASO

- Pulsamos en la primera operación (**Borrar producto**), como se muestra en la siguiente *Figura 378: Operación (Borrar producto)*:

The screenshot shows a web-based management system. The top navigation bar includes the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the CENDITEL logo, and a user session indicator. The main header is 'Tibisay - Gestión'. The central content area is titled 'Productos' with a pencil icon. On the left, a sidebar lists several actions: 'Agregar producto', 'Realizar pedido producto', 'Modificar datos producto', 'Siguiente estado producto', and 'Borrar producto'. A cursor is hovering over the 'Borrar producto' link. At the bottom, there are four footer sections: 'Acerca de nuestro sitio' (with text about contributing to project planning), 'CENDITEL' (with links to '¿Quiénes somos?', '¿Cómo lo haremos?', '¿Cuál es nuestra misión?', '¿Qué queremos?', and '¿Cuál es el impacto?'), 'Proyecto SAFET' (with links to 'Wiki' and 'Gestor del proyecto (Trac)'), and a search bar labeled 'Busque en nuestro sitio'.

Figura 8.36: Figura 378: Operación (Borrar producto)

#### 2° SEGUNDO PASO

- Seleccionamos el producto a **borrar**, en este ejemplo vamos a seleccionar el producto ((8) **Jabón Ace**), como se muestra en la siguiente *Figura 379: Seleccionar producto*:

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main title "Tibisay - Gestión" is displayed above a form. The form has a field labeled "Id \*" with a dropdown menu open, showing a list of items: (1) - Jabón Protex, (2) - Pañales Pamper, (3) - Vitamina C, (4) - Vitamina D, (5) - Vitamina B12, (6) - Vitamina B1, (7) - Vitamina B8, and (8) - Jabón Ace. The item "(8) - Jabón Ace" is highlighted with a blue selection bar. Below the form, there are footer sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.37: Figura 379: Seleccionar producto

### 3° TERCER PASO

- Pulsamos en el botón (Enviar) para finalizar con la operación, como se muestra en la siguiente *Figura 380: botón (Enviar)*:

The screenshot shows a web interface for managing products. At the top, there's a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an 'Administración - admin/Administrador' link. The main content area is titled 'Tibisay - Gestión' and contains a sub-section for deleting a product. It shows an input field with the value '(8) - Jabón Ace', a 'Borrar producto' button, and two action buttons: 'Enviar' (highlighted with a cursor) and 'Limpiar'. The footer features links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and a search bar.

Figura 8.38: **Figura 380: botón (Enviar)**

### 4° CUARTO PASO

**Nota:** Si deseas realizar otro pedido, solo pulsamos en la opción (Regresar a formulario), como se muestra en la siguiente *Figura 381: Regresar a formulario*:

■ Se nos devuelve al formulario, Obseven la siguiente *Figura 382: Formulario*:

The screenshot shows a web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session indicator. The main content area has a green header titled "Tibisay - Resultado de la operación". A green success message box contains the text "Se realizó la operación correctamente!". Below this, a link "Regresar al Formulario" is visible. The footer contains sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and a search bar.

**Acerca de nuestro sitio**  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan acabo en los proyectos, gestionar flujos de trabajos con la incorporación de herramientas como la firma electrónica

**CENDITEL**  
¿Quién es somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

**Proyecto SAFET**  
Wiki  
Gestor del proyecto (Trac)

**Busque en nuestro sitio**  
  
**Buscar**

Figura 8.39: Figura 381: Regresar a formulario

## Documentación de PySafet, Publicación

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header bar with the Venezuelan Government logo, the text 'Gobierno Bolivariano de Venezuela', 'Ministerio del Poder Popular para Ciencia, Tecnología e Innovación', 'Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)', and the 'Juventud' logo. Below the header, a navigation menu includes 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', 'CERRAR SESIÓN', and 'Administración - admin/Administrador'. The main content area has a dark blue header with the text 'Tibisay - Gestión' and a small globe icon. Below this, there is a form with a dropdown menu labeled 'Id \*', two buttons 'Enviar' and 'Limpiar', and a search bar with a placeholder 'Busque en nuestro sitio' and a green 'Buscar' button.

Acerca de nuestro sitio

Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan acabo en los proyectos, gestionar flujos de trabajos con la incorporación de

CENDITEL

¿Quién es somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

Proyecto SAFET

Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio

Buscar

Figura 8.40: Figura 382: Formulario

## 5° QUINTO PASO

- Para salir de la operación pulsamos en el menu en la opción (**Gestión**), como se muestra en la siguiente *Figura 383: Opción (Gestión)*:

The screenshot shows a web application interface. At the top, there is a header with the logo of the Government of Venezuela, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel). Below the header, a navigation bar includes links for INICIO, GESTIÓN (which is highlighted in a black box), CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the title "Tibisay - Resultado de la operación". Below this, a green box contains a checkmark icon and the text "Se realizó la operación correctamente!". At the bottom of the page, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and a search bar.

<b>Acerca de nuestro sitio</b>	<b>CENDITEL</b>	<b>Proyecto SAFET</b>	<b>Busque en nuestro sitio</b>
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de herramientas como la firma electrónica	<a href="#">¿Quiénes somos?</a> <a href="#">¿Cómo lo haremos?</a> <a href="#">¿Cuál es nuestra misión?</a> <a href="#">¿Qué queremos?</a> <a href="#">¿Cuál es el impacto?</a>	<a href="#">Wiki</a> <a href="#">Gestor del proyecto (Trac)</a>	<input type="text"/> <input type="button" value="Buscar"/>

Figura 8.41: *Figura 383: Opción (Gestión)*

**Nota:** Se nos devuelve a las opciones de (CRUD+). Observen la siguiente *Figura 384: opciones de (CRUD+)*:

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Juventud 2040 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Administración - admin/Administrador". The main content area has a dark blue header with a globe icon and the text "Tibisay - Gestión". Below this, a section titled "Productos" features a pencil icon and links for "Agregar producto", "Realizar pedido producto", "Modificar datos producto", "Siguiente estado producto", and "Borrar producto". At the bottom of the page is a footer with four columns: "Acerca de nuestro sitio" (with text about contributing to project planning), "CENDITEL" (with links to "¿Quiénes somos?", "¿Cómo lo haremos?", "¿Cuál es nuestra misión?", "¿Qué queremos?", and "¿Cuál es el impacto?"), "Proyecto SAFET" (with links to "Wiki" and "Gestor del proyecto (Trac)"), and "Busque en nuestro sitio" (with a search input field and a "Buscar" button).

Figura 8.42: Figura 384: opciones de (CRUD+)

## 8.3 Ejecución de listados de reportes

### 8.3.1 A.- Entramos al sistema y buscamos la opción (Ingresar o modificar información) o (Gestion)

#### 1° PRIMER PASO

- Abrimos nuestro navegador Web y colocamos la url (<http://localhost/intranet/>), como se muestra en la siguiente *Figura 385: Navegador Web*:

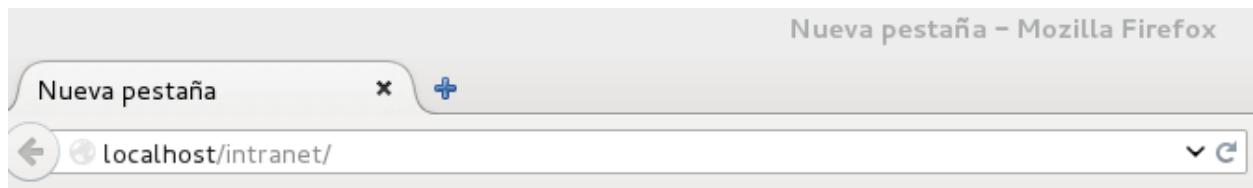


Figura 8.43: **Figura 385: Navegador Web**

#### 2° SEGUNDO PASO

- Damos click a **Entrar a safet**, como se muestra en la siguiente *Figura 386: Entrar a safet*:

#### 3° PRIMER PASO

- Colocamos el usuario/password (**admin/admin**) y pulsamos en el botón (**Iniciar Sesión**) para entrar al sistema, como se muestra en la siguiente *Figura 387: Autenticación de Safet*:

#### 4° CUARTO PASO

- Pulsamos el la opción (**Generar reporte o gráficos de consulta**), como se muestra en la siguiente *Figura 388: Opción (Generar reporte o gráficos de consulta)*:

#### 5° QUINTO PASO

- Pulsamos en la operación (**Listar datos**), como se muestra en la siguiente *Figura 389: Operación (Listar datos)*:

## Documentación de PySafet, Publicación

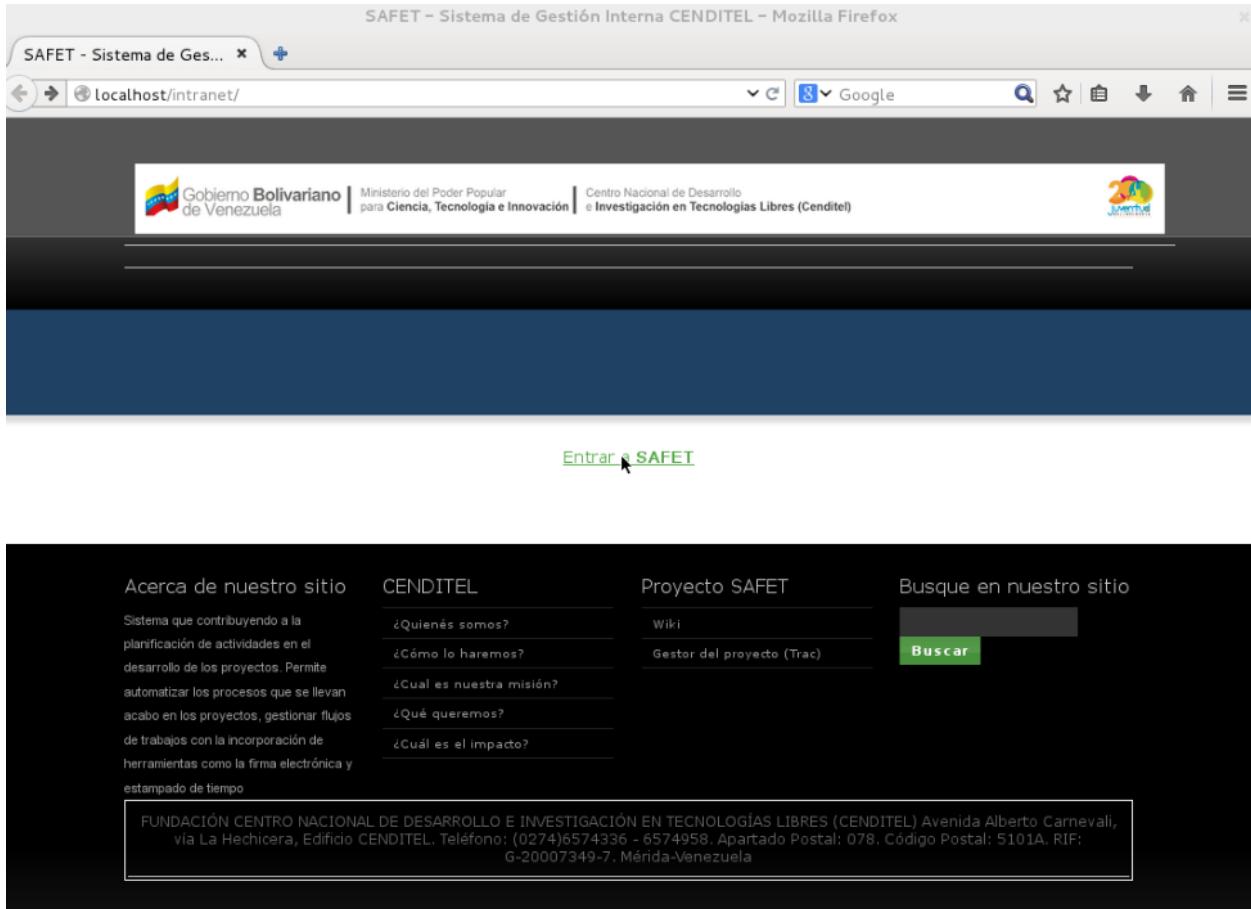


Figura 8.44: Figura 386: Entrar a safet



Figura 8.45: Figura 387: Autenticación de Safet

## 6° SEXTO PASO

- Seleccionamos el archivo (**XML**) en este ejemplo sería el archivo (**productos.xml**), como se muestra en la siguiente *Figura 390: Selección del archivo (xml)*:

## 7° SEPTIMO PASO

- Seleccionamos el variable a listar, en este ejemplo listaremos los datos de la variable (**Vregistrado**), como se muestra en la siguiente *Figura 391: Selección de la variable*:

## 8° OCTAVO PASO

- Pulsamos en el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *Figura 392: botón (Enviar)*:

**Nota:** Se nos muestra una tabla con los datos y tu título del reporte que seleccionamos anterior (vRegistrado), como se muestra en la siguiente *Figura 393: Reporte de la variable vRegistrado*:

The screenshot shows the Tibisay - Opciones Generales page of the SAFET system. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a blue header with the text "Tibisay - Opciones Generales" and a small globe icon. Below the header, it says "Sistema Automatizado para la Firma Electrónica y el Estampillado de tiempo SAFET" and "Usted desea realizar una de las siguientes opciones:". There are four options listed: "Ingreso o modificación de información" (highlighted in green), "Generar reportes" (with a small arrow icon), "gráficos de consulta" (highlighted in green), and "Cerrar Sesión". At the bottom, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Acerca de nuestro sitio

Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de

CENDITEL

¿Quiénes somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

Proyecto SAFET

Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio

Buscar

Figura 8.46: Figura 388: Opción (Generar reporte o gráficos de consulta)

The screenshot shows the Tibisay - Consultas application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the text "Tibisay - Consultas" and a small icon. Below this, there is a section titled "Opciones de listado" with a sub-section titled "Listar datos". Under "Listar datos", there are three options: "Listar datos", "Listar datos con autofiltro", and "Listar datos con filtrorecursivo".

## Gráficos básicos

Generar gráfico coloreado  
Generar gráfico para dave

## Gráficos con filtro

Generar gráfico con autofiltro  
Generar gráfico con filtrorecursivo

A footer navigation bar with links for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.47: Figura 389: Operación (Listar datos)

The screenshot shows a web application interface for 'Tibisay - Consultas'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel). Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an 'Administración' link for the user 'admin/Administrador'. The main title 'Tibisay - Consultas' is displayed above a green header bar. Below this, a section titled 'Listar datos' contains two input fields: 'Cargar archivo flujo \*' and 'Variable \*'. The 'Variable' field is a dropdown menu showing two options: '/home/cenditel/.safet/flowfiles/productos.xml' and '/home/cenditel/.safet/flowfiles/productos\_sig\_Estado.xml'. Below these fields are buttons for 'Enviar' and 'Limpiar', and a 'Parámetros' button. At the bottom of the page, there is a footer with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio', along with a search bar.

Figura 8.48: Figura 390: Selección del archivo (xml)

The screenshot shows the Tibisay - Consultas application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Administración - admin/Administrador". The main title "Tibisay - Consultas" is displayed above a green banner. Below the banner, the text "Listar datos" is visible. The form area contains fields for "Cargar archivo flujo \*" with a value of "/home/cenditel/.safet/flowfiles/productos.xml", a dropdown menu for "Variable \*", and buttons for "Enviar" and "Limpiar". A "Parámetros" button is also present. The dropdown menu is open, showing options: vPor\_agotarse, vPor\_llegar, vRegistrado (which is highlighted), vDisponible, vPedido, and vAgotado. At the bottom of the page, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.49: Figura 391: Selección de la variable

## Documentación de PySafet, Publicación



Listar datos



Figura 8.50: Figura 392: botón (Enviar)

A screenshot of a report page titled "Tibisay - Reportes de los registros". The main title is "Listado de los registros del proyecto: vRegistrado". Below the title, there's a pagination control with links for "primero", "anterior", "1", "siguiente", and "último" followed by a page number "25" and a refresh icon. A table follows, showing a list of registered items with columns for "id", "nombre", "cantidad", "status", and "categoria". The items listed are Jabón Protex, Pañales Pamper, Vitamina C, Vitamina D, Vitamina B12, Vitamina B1, and Vitamina B8. All items have a status of "Registrado" and a category of "no\_clasificado". At the bottom, there's another pagination control with the same links and page number.

Figura 8.51: Figura 393: Reporte de la variable vRegistrado

## 9° NOVENO PASO

- Pulsamos en el menú la opción consulta para salir de la operación, como se muestra en la siguiente *Figura 394: Salir de la operación:*

id	nombre	car	status	categoría
1	Jabón Protex	0	Registrado	no_clasificado
2	Pañales Pamper	0	Registrado	no_clasificado
3	Vitamina C	0	Registrado	no_clasificado
4	Vitamina D	0	Registrado	no_clasificado
5	Vitamina B12	0	Registrado	no_clasificado
6	Vitamina B1	0	Registrado	no_clasificado
7	Vitamina B8	0	Registrado	no_clasificado

Figura 8.52: Figura 394: Salir de la operación

**Nota:** Al pulsar en la opción consulta se nos mostrara las operaciones de consulta, como se muestra en la siguiente *Figura 395: Operación consulta:*

## 8.4 Ejecución de gráficos de flujos de trabajo

### 8.4.1 A.- Ingresamos al sistema y buscamos la opción (Ingresar o modificar información) o (Gestion)

#### 1° PRIMER PASO

- Abrimos nuestro navegador Web y colocamos la url (<http://localhost/intranet/>), como se muestra en la siguiente *Figura 396: Navegador Web:*

The screenshot shows the Tibisay - Consultas web application. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the Tibisay logo and the text "Tibisay - Consultas". The main content area contains sections for "Opciones de listado", "Gráficos básicos", "Gráficos de seguimiento", and "Gráficos con filtro", each with associated icons and links.

The screenshot shows the footer of the Project SAFET website. It includes sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and a search bar. The "Acerca de nuestro sitio" section contains text about the system's purpose. The "CENDITEL" section lists links to "¿Quién es somos?", "¿Cómo lo haremos?", "¿Cuál es nuestra misión?", and "¿Qué queremos?". The "Proyecto SAFET" section lists "Wiki" and "Gestor del proyecto (Trac)". The search bar has a "Buscar" button.

Figura 8.53: Figura 395: Operación consulta

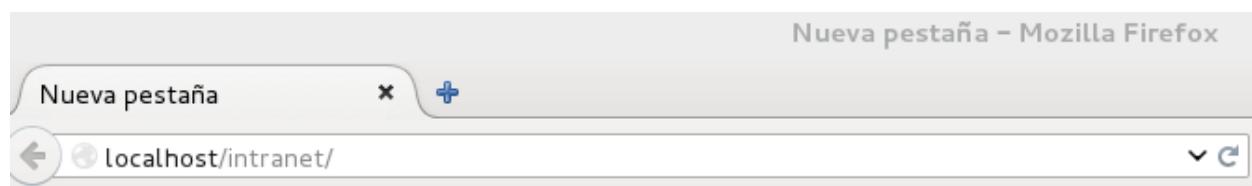


Figura 8.54: Figura 396: Navegador Web

## 2° SEGUNDO PASO

- Damos click a **Entrar a safet**, como se muestra en la siguiente *Figura 397: Entrar a safet*:

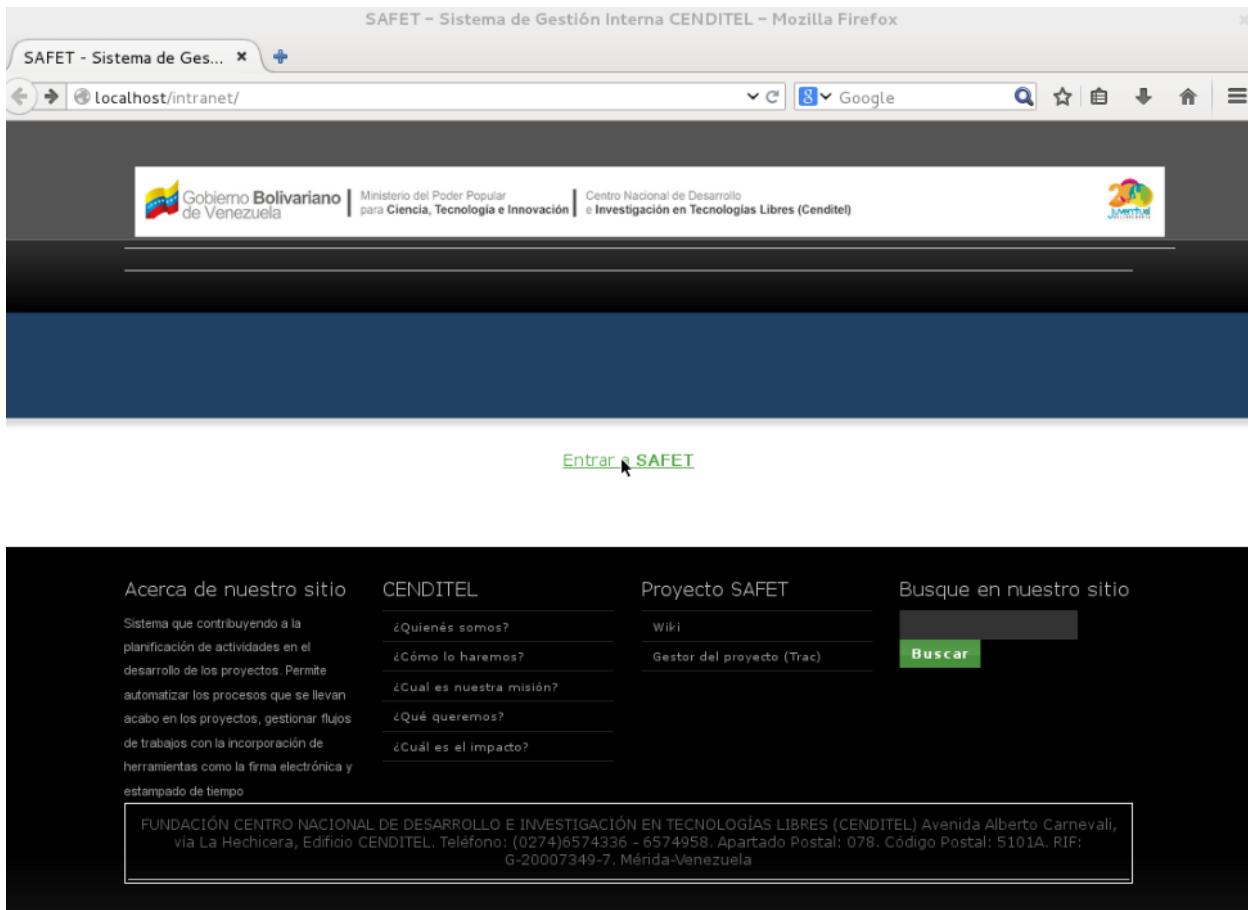


Figura 8.55: **Figura 397: Entrar a safet**

## 3° PRIMER PASO

- Colocamos el usuario/password (**admin/admin**) y pulsamos en el botón (**Iniciar Sesión**) para entrar al sistema, como se muestra en la siguiente *Figura 398: Autenticación de Safet*:

## 4° CUARTO PASO

- Pulsamos el la opción (**Generar reporte o gráficos de consulta**), como se muestra en la siguiente *Figura 399: Opción (Generar reporte o gráficos de consulta)*:

## Documentación de PySafet, Publicación

Figura 8.56: Figura 398: Autenticación de Safet

### 8.4.2 B.- Gráficos general

#### 1° PRIMERA PASO

- Pulsamos en la operación (**Generar gráfico coloreado**), como se muestra en la siguiente *Figura 400: Operación (Generar gráfico coloreado)*:

#### 2° SEGUNDO PASO

- Seleccionamos el archivo (**.xml**) en este caso se llama (**productos.xml**), como se muestra en la siguiente *Figura 401: Selección de archivo (XML)*:

#### 3° PRIMER PASO

- Pulsamos en el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *Figura 402: Botón (Enviar)*:

**Nota:** Se nos muestra todo los productos que existen en cada procesos, como se muestra en la

Gobierno Bolivariano de Venezuela | Ministerio del Poder Popular para Ciencia, Tecnología e Innovación | Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)

INICIO    GESTIÓN    CONSULTAS    CRÉDITOS    CERRAR SESIÓN    Administración - admin/Administrador

## Tibisay - Opciones Generales

Sistema Automatizado para la Firma Electrónica y el Estampillado de tiempo SAFET  
Usted desea realizar una de las siguientes opciones:

Ingreso o modificación de información  
Generar reportes > gráficos de consulta  
Cerrar Sesión

Acerca de nuestro sitio	CENDITEL	Proyecto SAFET	Busque en nuestro sitio
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de	<a href="#">¿Quiénes somos?</a> <a href="#">¿Cómo lo haremos?</a> <a href="#">¿Cuál es nuestra misión?</a> <a href="#">¿Qué queremos?</a> <a href="#">¿Cuál es el impacto?</a>	<a href="#">Wiki</a> <a href="#">Gestor del proyecto (Trac)</a>	<input type="text"/> <b>Buscar</b>

Figura 8.57: Figura 399: Opción (Generar reporte o gráficos de consulta)

The screenshot shows a web application interface. At the top, there is a header bar with the Venezuelan government logo, the text "Gobierno Bolivariano de Venezuela", "Ministerio del Poder Popular para Ciencia, Tecnología e Innovación", "Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)", and the "Juventud" logo. Below the header, a navigation bar includes links for "INICIO", "GESTIÓN", "CONSULTAS", "CRÉDITOS", and "CERRAR SESIÓN". On the right side of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the text "Tibisay - Consultas" and a small icon. Below this, there are two sections: "Opciones de listado" with three options ("Listar datos", "Listar datos con autofiltro", "Listar datos con filtrorecursivo") and "Gráficos básicos" with two options ("Generar gráfico coloreado", "Generar gráfico para clave").

Figura 8.58: Figura 400: Operación (Generar gráfico coloreado)

The screenshot shows a web application interface for 'Tibisay - Consultas'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. To the right of the navigation bar, it says 'Administración - admin/Administrador'. The main content area has a green header with the title 'Tibisay - Consultas' and a small icon. Below this, there is a form field labeled 'Generar gráfico coloreado' with a dropdown menu for selecting an XML file. The dropdown menu shows two options: '/home/cenditel/.safet/flowfiles/productos.xml' and '/home/cenditel/.safet/flowfiles/productos\_sig\_Estado.xml'. There are also 'Enviar' and 'Limpiar' buttons. A 'Parámetros' button is present, along with a note: 'Cargue de nuevo la página para ver los parámetros'. At the bottom, there is a footer with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', and 'Busque en nuestro sitio'.

Figura 8.59: Figura 401: Selección de archivo (XML)

The screenshot shows the Tibisay - Consultas web application. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and the Juventud 20 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an 'Administración - admin/Administrador' link. The main content area has a green header with the title 'Tibisay - Consultas' and a small icon. Below this, a section titled 'Generar gráfico coloreado' contains a file input field set to '/home/cenditel/.safet/flowfiles/productos.xml'. There are 'Enviar' and 'Limpiar' buttons. A 'Parámetros' button is present, with a note below it stating 'Cargue de nuevo la página para ver los parámetros'. A 'Nombre' input field is also shown. At the bottom, there is a footer with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', 'Busque en nuestro sitio', and search fields for 'Sistema que contribuyendo a la' and '¿Quiénes somos?'. There is also a 'Wiki' link.

Figura 8.60: **Figura 402: Botón (Enviar)**

siguiente *Figura 403: Productos*:

---

### 8.4.3 C.- Gráficos por clave

#### 1° PRIMERA PASO

- Pulsamos en la operación (**Generar gráfico para clave**), como se muestra en la siguiente *Figura 405: Operación (Generar gráfico para clave)*:

#### 2° SEGUNDO PASO

- Seleccionamos el archivo (**.xml**) en este caso se llama (**productos.xml**), como se muestra en la siguiente *Figura 406: Selección del archivo (XML)*:

#### 3° TERCER PASO

- Seleccionamos el producto en específico, como se muestra en la siguiente *Figura 407: Seleccionar producto*:

#### 4° CUARTO PASO

- Pulsamos en el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *Figura 408: Botón (Enviar)*:

---

**Nota:** Se nos gráficamente el producto donde esta ubicado en el procesos, como se muestra en la siguiente *Figura 409: Producto*:

---

## 8.5 Ejecución de gráficos de flujos de trabajo de seguimiento

### 8.5.1 A.- Descarga del archivo xml(**productos\_de\_seguimiento.xml**)

#### 1° PRIMER PASO

- Damos click como se muestra en la siguiente imagen para la de carga del archivo:

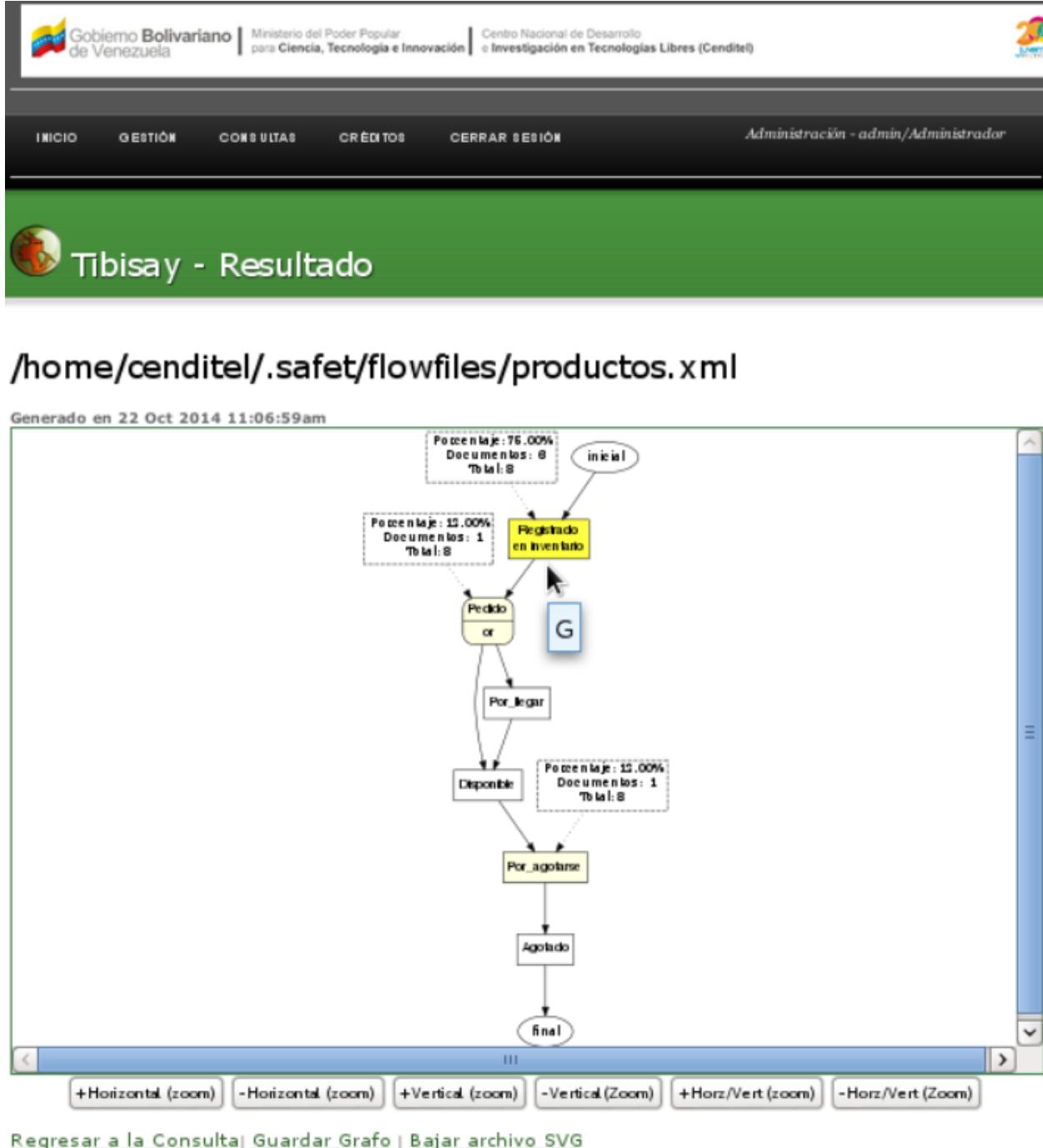


Figura 8.61: Figura 403: Productos

The screenshot shows a web-based application interface. At the top, there is a header with the Venezuelan Government logo, the text "Gobierno Bolivariano de Venezuela", the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel), and the Juventud 20 logo. Below the header is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with the text "Tibisay - Consultas" and a small icon of a document with a graph. Below this, there are two sections: "Opciones de listado" with links to "Listar datos", "Listar datos con autofiltro", and "Listar datos con filtrorecursivo"; and "Gráficos básicos" with links to "Generar gráfico coloreado" and "Generar gráfico para clave". There is also a section for "Gráficos con filtro" with links to "Generar gráfico con autofiltro" and "Generar gráfico con filtrorecursivo". A large black horizontal bar is at the bottom of the page.

Figura 8.62: Figura 405: Operación (Generar gráfico para clave)

The screenshot shows a web application interface for generating a flowchart from an XML file. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the CENDITEL logo, and a Jovenes Unidos logo. The menu bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right, it says "Administración - admin/Administrador". Below the header, the title "Tibisay - Consultas" is displayed next to a small icon. A green banner below the title contains the text "Generar gráfico para clave". The main form has fields for "Cargar archivo flujo \*" (with a dropdown menu showing "/home/cenditel/safet/flowfiles/productos.xml" and "/home/cenditel/safet/flowfiles/productos\_sig\_Estado.xml"), "Clave \*" (with two options), and buttons for "Enviar" and "Limpiar". A "Parámetros" button is present. Below the form, a message says "Cargue de nuevo la página para ver los parámetros". At the bottom, there is a footer with links for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio". The "CENDITEL" section includes "Sistema que contribuyendo a la" and "¿Quiénes somos?". The "Proyecto SAFET" section includes "Wiki".

Figura 8.63: Figura 406: Selección del archivo (XML)

The screenshot shows a web application interface for 'Tibisay - Consultas'. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with an administrator login status ('Administración - admin/Administrador'). The main content area has a green header with the application name 'Tibisay - Consultas' and a small icon. Below this, a form is displayed for generating a flowchart for a key. The form includes fields for 'Cargar archivo flujo \*' (with a value of '/home/cenditel/.safet/flowfiles/productos.xml'), 'Clave \*' (with a dropdown menu showing a list of products), and buttons for 'Enviar' and 'Limpiar'. A 'Parámetros' button is also present. A message at the bottom says 'Cargue de nuevo la página para ver los parámetros'. At the bottom of the page, there is a footer with links for 'Acerca de nuestro sitio', 'CENDITEL', 'Proyecto SAFET', 'Busque en nuestro sitio', and sections for 'Sistema que contribuyendo a la' and '¿Quiénes somos?'. There is also a 'Wiki' link.

Generar gráfico para clave

Cargar archivo flujo \*

/home/cenditel/.safet/flowfiles/productos.xml

Clave \*

(1) - Jabón Protex  
(2) - Pañales Pamper  
(3) - Vitamina C  
(4) - Vitamina D  
(5) - Vitamina B12  
(6) - Vitamina B1  
(7) - Vitamina B8

Enviar Limpiar

Parámetros

Cargue de nuevo la página para ver los parámetros

Nombre

Acerca de nuestro sitio

Sistema que contribuyendo a la

¿Quiénes somos?

CENDITEL

Proyecto SAFET

Wiki

Busque en nuestro sitio

Figura 8.64: Figura 407: Seleccionar producto

The screenshot shows a web application interface for generating a flowchart for a key. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, there is a navigation menu with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Administración - admin/Administrador". The main title "Tibisay - Consultas" is displayed with a small icon. Below the title, there is a form field labeled "Generar gráfico para clave" with a placeholder "Cargar archivo flujo \*". A dropdown menu shows the path "/home/cenditel/.safet/flowfiles/productos.xml". Another dropdown menu shows "(1) - Jabón Protex". At the bottom of the form, there are two buttons: "Enviar" (highlighted with a cursor) and "Limpiar". Below the form, there is a link "Parámetros" and a note "Cargue de nuevo la página para ver los parámetros". There is also a "Nombre" input field. At the very bottom of the page, there is a footer with links for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.65: Figura 408: Botón (Enviar)

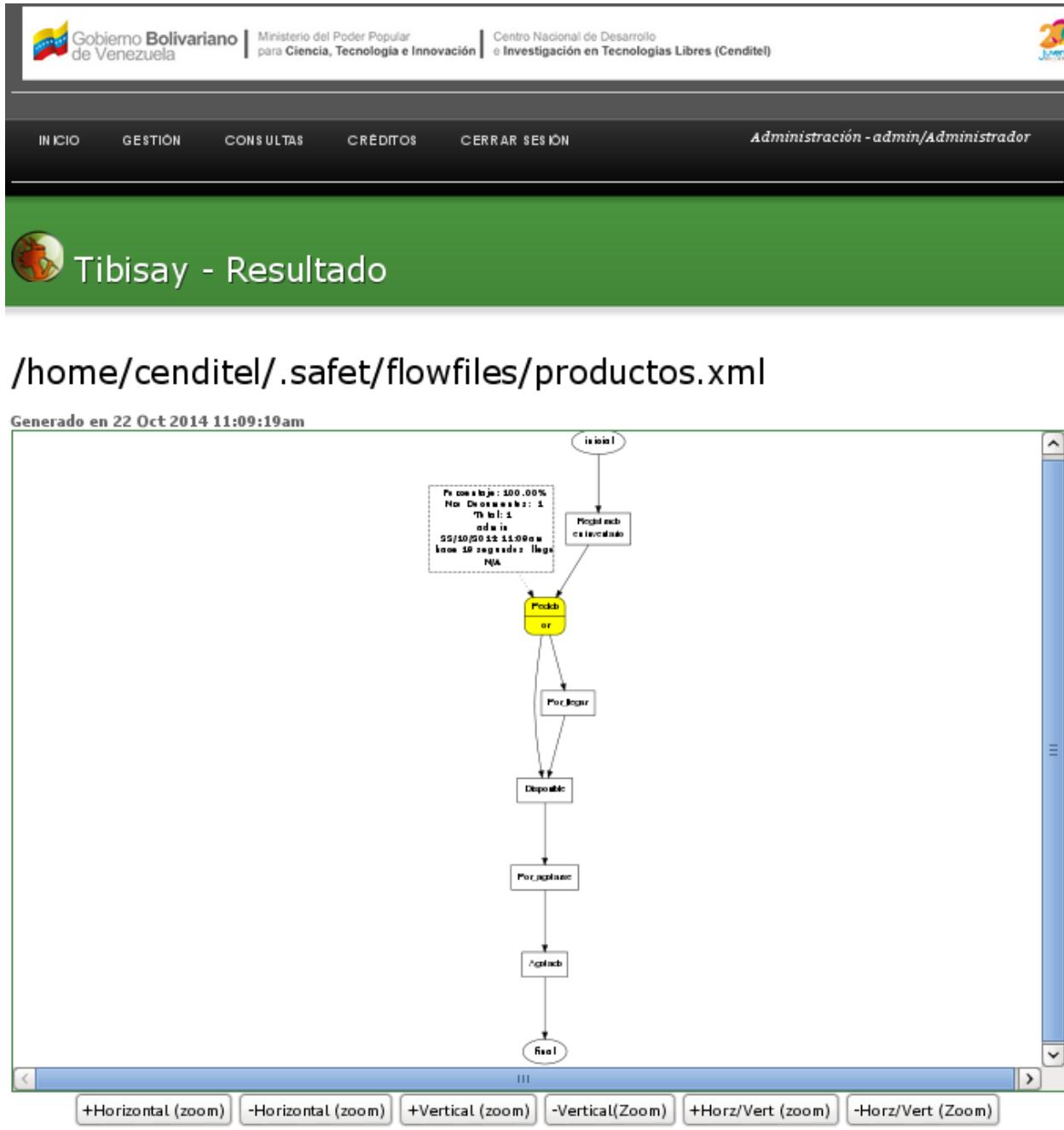


Figura 8.66: Figura 409: Producto



Figura 8.67: Click aquí (productos\_de\_seguimiento.xml)

## 2° SEGUNDO PASO

- Descomprimimos el archivo **.tar.gz** que descargamos en el paso anterior y guardamos el archivo (**productos\_de\_seguimiento.xml**) en el directorio **/home/usuario/.safet/flowfiles/**.

### 8.5.2 B.- Modificación del archivo xml

---

**Nota:** En el archivo **productos\_de\_seguimiento.xml** del directorio **/home/usuario/.safet/flowfiles/** se le agrego el parámetro (**InfoText,InfoDate**), como se muestra a continuación:

```
<parameter
    title="Campos"
    options=""
    configurekey="Plugins.Graphviz/plugins.graphviz.extrainfo.show"
    type="string"
    mandatory="no"
    defaultvalue="InfoText,InfoDate"
/>
```

---

**Nota:** En el archivo **productos\_de\_seguimiento.xml** del directorio **/home/usuario/.safet/flowfiles/** se puede cambiar o manejar valores de **texto y fecha** en cada ficha. Dentro de la etiqueta (**<variable>**) se encuentra el atributos (**rolfield**) y (**timestampfield**) la cual se puedes concatenar los valores de cada ficha.

---

### 1° Concatenación de los atributos (**rolfield**) y (**timestampfield**)

1.- Observen el siguiente código, como en el atributo (**rolfield**) y en el atributo (**timestampfield**) no estan concatenados:

```

<variable config="1"
documentsource="select nombre,cantidad,status,categoria from productos"
type="sql" tokenlink=""
id="vRegistrado" scope="task"

rolfield=" (select rol from productos_registro
where productoid=productos.id and regstatus='Registrado') as rol"

timestampfield="(select fecha from productos_registro
where productoid=productos.id and regstatus='Registrado') as fecha"
/>

```

**2.- Observen el siguiente código, como en el atributo (rolfield) el rol y en el atributo (timestampfield) la fecha estan concatenados:**

```

<variable config="1"
documentsource="select nombre,cantidad,status,categoria from productos"
type="sql" tokenlink=""
id="vRegistrado" scope="task"

rolfield=" (select 'Rol:' || rol from productos_registro
where productoid=productos.id and regstatus='Registrado') as rol"

timestampfield="(select fecha-16200 from productos_registro
where productoid=productos.id and regstatus='Registrado') as fecha"
/>

```

## 2º Visulazación de la base de datos

Observamos en nuestra base de datos sqlite (**mydb.db**), demostrando con un producto los valores de seguimiento de (**rolfield** y **timestampfield**), como se muestra a continuación:

**1.-** Abrimos nuestra base de datos desde el navegador **web** con la herramienta **sqlite-manager** y seleccionamos un producto de la tabla (**productos**) para verle el seguimiento en los procesos. En este caso tomaremos como ejemplo el producto (**Soflan super**), como se muestra en la siguiente *figura411*:

**2.-** En la tabla (**productos\_registro**) observamos el seguimiento del producto que seleccionamos (**Soflan super**), donde nos muestra los valores del **productoid(id del producto)**,**rolfield(roles)**,**timestampfield(fecha)** y **el status(Por los procesos donde a recorrido o pasado)**, como se muestra a continuación *figura412*:

**3.-** En el archivo **safet.conf** del directorio **/home/usuario/.safet/flowfiles/** se sebe activa de **(on) a (off)** el plugins (**plugins.graphviz.extrainfo.showwheretokens**) para que muestre la información de cada tokens o ficha,como se muestra a continuación:

```
# Mostrar la información extra solo donde existan tokens (fichas) (on/off)
plugins.graphviz.extrainfo.showwheretokens = off
```

## Documentación de PySafet, Publicación

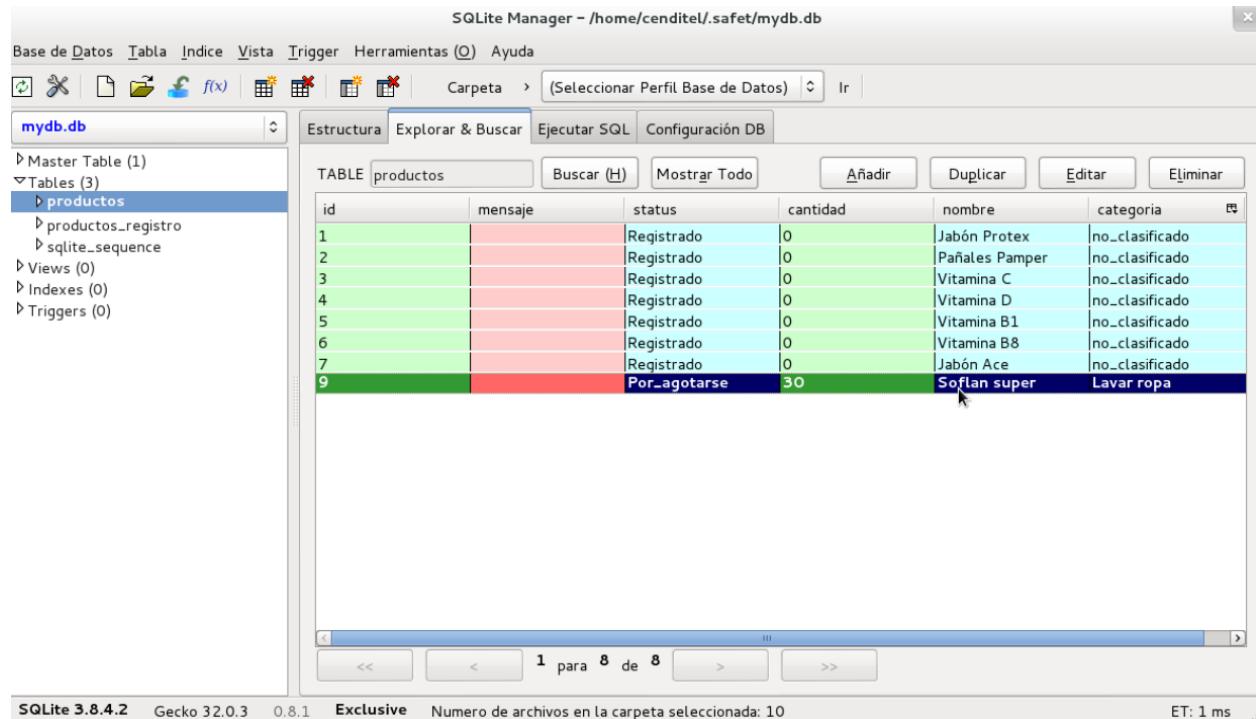


Figura 8.68: Figura 411: Selección de producto

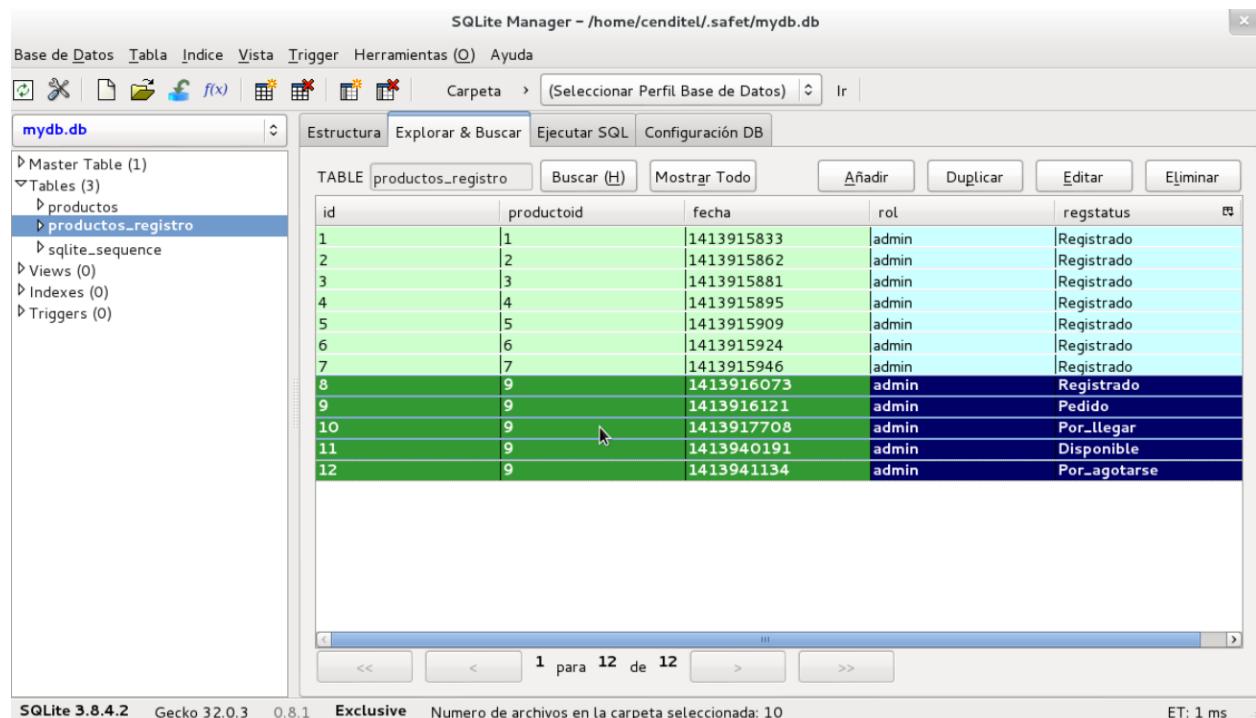


Figura 8.69: Figura 412: Seguimiento del producto

### 8.5.3 C.- Ingresamos al sistema y buscamos la opción (Ingresar o modificar información) o (Gestion)

#### 1° PRIMER PASO

- Abrimos nuestro navegador **Web** y colocamos la **url** (<http://localhost/intranet/>), como se muestra en la siguiente *figura413*:

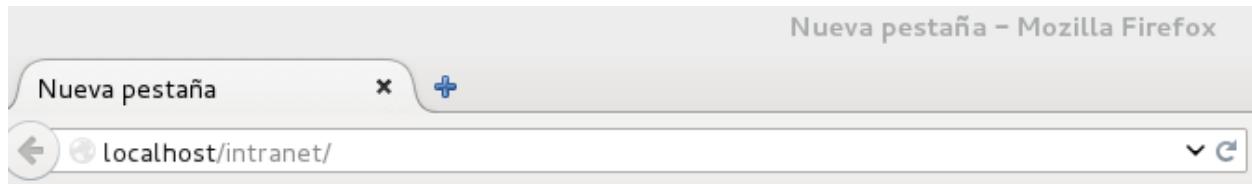


Figura 8.70: **Figura 413: Navegador Web**

#### 2° SEGUNDO PASO

- Damos click a **Entrar a safet**, como se muestra en la siguiente *figura414*:

#### 3° PRIMER PASO

- Colocamos el usuario/password (**admin/admin**) y pulsamos en el botón (**Iniciar Sesión**) para entrar al sistema, como se muestra en la siguiente *figura415*:

#### 4° CUARTO PASO

- Pulsamos el la opción (**Generar reporte o gráficos de consulta**), como se muestra en la siguiente *figura416*:

### 8.5.4 D.- Gráficos de seguimiento

#### 1° PRIMERA PASO

- Pulsamos en la operación (**Generar gráfico de seguimiento**), como se muestra en la siguiente *figura417*:

## Documentación de PySafet, Publicación

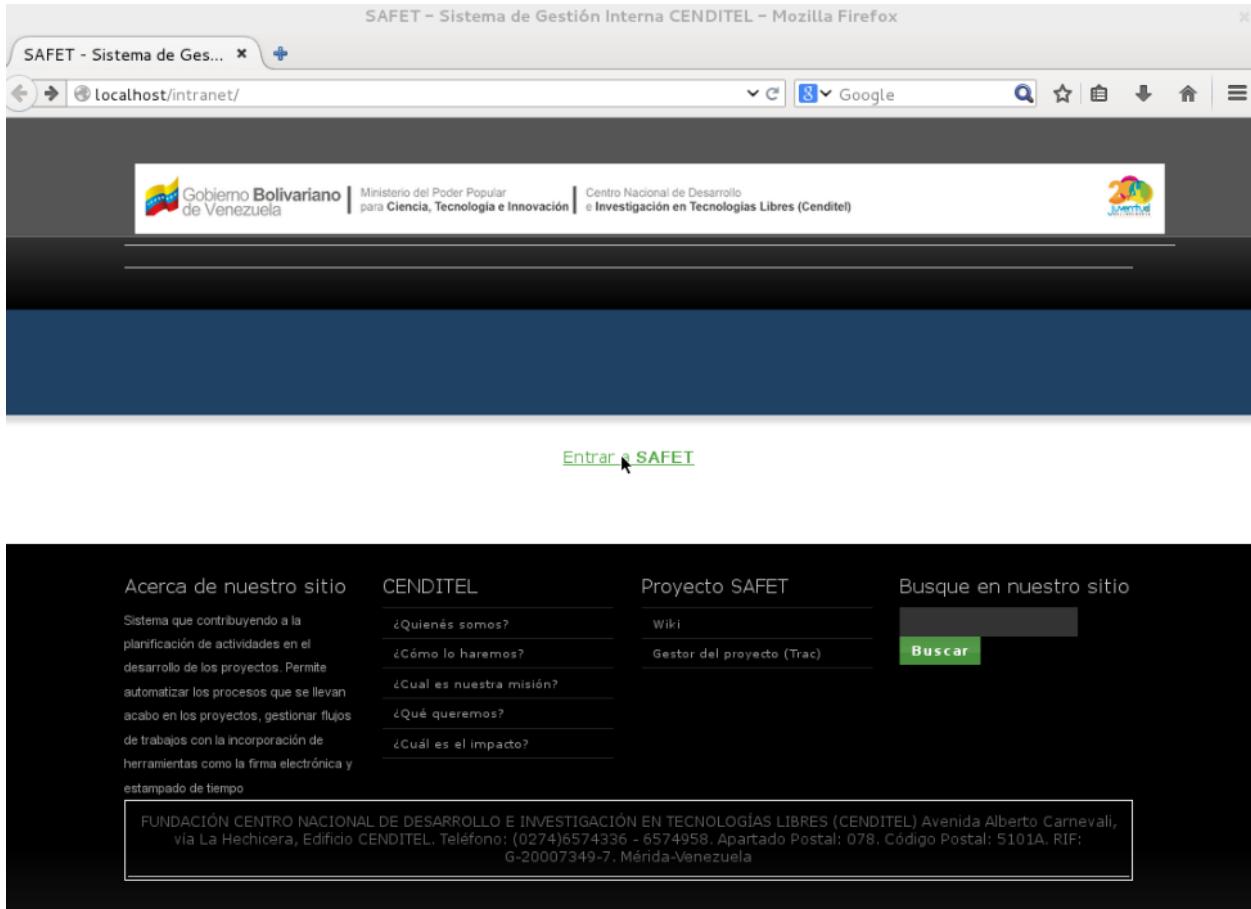


Figura 8.71: Figura 414: Entrar a safet



Figura 8.72: Figura 415: Autenticación de Safet

## 2° SEGUNDO PASO

- Seleccionamos el archivo (.xml) en este caso se llama (**productos.xml**), como se muestra en la siguiente *figura418*:

## 3° TERCER PASO

- Seleccionamos el producto en específico, como se muestra en la siguiente *figura420*:

## 4° CUARTO PASO

- Pulsamos en el botón (**Enviar**) para finalizar con la operación, como se muestra en la siguiente *figura421*:

## 5° QUINTO PASO

- Se nos gráficamente el producto por donde a pasado en cada proceso, mostrando una un mensajes del **rol, fecha, y la hora** cuando llegó la tarea o proceso, como se muestra en la siguiente *figura422* y la

The screenshot shows the Tibisay - Opciones Generales page of the SAFET system. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Administración - admin/Administrador". The main content area has a blue header with the text "Tibisay - Opciones Generales" and a small globe icon. Below the header, it says "Sistema Automatizado para la Firma Electrónica y el Estampillado de tiempo SAFET" and "Usted desea realizar una de las siguientes opciones:". There are four options listed: "Ingreso o modificación de información" (highlighted in green), "Generar reportes o gráficos de consulta" (highlighted in green), "Cerrar Sesión" (in green), and "Generar reportes o gráficos de consulta" (in green). At the bottom, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Acerca de nuestro sitio  
Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos, gestionar flujos de trabajos con la incorporación de

CENDITEL  
¿Quiénes somos?  
¿Cómo lo haremos?  
¿Cuál es nuestra misión?  
¿Qué queremos?  
¿Cuál es el impacto?

Proyecto SAFET  
Wiki  
Gestor del proyecto (Trac)

Busque en nuestro sitio  
Buscar

Figura 8.73: Figura 416: Opción (Generar reporte o gráficos de consulta)

The screenshot shows the Tibisay - Consultas application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, and the CENDITEL logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the navigation bar, it says "Administración - admin/Administrador". The main content area has a green header with a globe icon and the text "Tibisay - Consultas". Below this, there is a section titled "Opciones de listado" with a small icon of a document with arrows. Underneath, there are three options: "Listar datos", "Listar datos con autofiltro", and "Listar datos con filtrorecursivo".

## Opciones de listado

Listar datos  
Listar datos con autofiltro  
Listar datos con filtrorecursivo

## Gráficos básicos

Generar gráfico coloreado  
Generar gráfico para clave

## Gráficos de seguimiento

Generar gráfico de seguimiento

## Gráficos con filtro

Generar gráfico con autofiltro  
Generar gráfico con filtrorecursivo

The footer section of the Tibisay - Consultas application. It contains four main links: "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio". The "Acerca de nuestro sitio" link has a sub-link "Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite". The "CENDITEL" link has two sub-links: "¿Quién es somos?" and "¿Cómo lo haremos?". The "Proyecto SAFET" link has two sub-links: "Wiki" and "Gestor del proyecto (Trac)". To the right of these links is a search bar with a "Buscar" button.

Figura 8.74: Figura 417: Operación (Generar gráfico de seguimiento)

The screenshot shows a web application interface for generating a tracking graph. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right of the navigation bar, it says "Administración - admin/Administrador". The main title "Tibisay - Consultas" is displayed above a green header bar. Below the header, there is a form with fields for "Cargar archivo flujo \*" (Load flowfile \*), "Producto \*" (Product \*), and buttons for "Enviar" (Send) and "Limpiar" (Clear). A dropdown menu is open over the "Producto" field, showing three options: "/home/cenditel/.safet/flowfiles/productos.xml", "/home/cenditel/.safet/flowfiles/productos\_siguiente\_Estado.xml", and "/home/cenditel/.safet/flowfiles/productos\_de\_seguimiento.xml". The third option is highlighted with a blue selection bar. Below the form, there is a "Parámetros" button and a note: "Cargue de nuevo la página para ver los parámetros". At the bottom, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.75: Figura 418: Selección del archivo (XML)

The screenshot shows a web application interface for generating a follow-up graph. At the top, there is a header with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, and the CENDITEL logo. The navigation menu includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right, it says "Administración - admin/Administrador". Below the header, a green banner displays the logo of a jaguar and the text "Tibisay - Consultas". The main form has fields for "Cargar archivo flujo \*" (Upload flowchart file) with a value of "/home/cenditel/.safet/flowfiles/productos\_de\_seguimiento.xml", "Producto \*" (Product \*), which is a dropdown menu listing various items like Jabón Protex, Pañales Pampers, Vitamina C, etc., with "Sofla super" selected; there are "Enviar" and "Limpiar" buttons next to it; and a "Parámetros" button. A note below says "Cargue de nuevo la página para ver los parámetros". Below this, there is a "Nombre" (Name) input field. At the bottom, there is a footer with sections for "Acerca de nuestro sitio" (About our site), "CENDITEL" (with links to "¿Quiénés somos?" and "¿Cómo lo haremos?"), "Proyecto SAFET" (with links to "Wiki" and "Gestor del proyecto (Trac)"), and a search bar with a "Buscar" (Search) button.

Figura 8.76: Figura 420: Seleccionar producto

## Documentación de PySafet, Publicación

The screenshot shows a web-based application titled "Tibisay - Consultas". At the top, there is a header with the Venezuelan Government logo, the text "Gobierno Bolivariano de Venezuela", "Ministerio del Poder Popular para Ciencia, Tecnología e Innovación", and "Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)". Below the header is a navigation bar with links for "INICIO", "GESTIÓN", "CONSULTAS", "CRÉDITOS", "CERRAR SESIÓN", and "Administración - admin/Administrador". The main content area has a green header with the title "Tibisay - Consultas" and a sub-header "Generar gráfico de seguimiento". Below this, there is a form with fields for "Cargar archivo flujo \*", a dropdown menu showing "/home/cenditel/.safet/flowfiles/productos\_de\_seguimiento.xml", "Producto \*", a dropdown menu showing "(0) - Soflan super", and two buttons "Enviar" and "Limpiar". A "Parámetros" button is also present. Further down, there is a message "Cargue de nuevo la página para ver los parámetros" and a "Nombre" input field. At the bottom, there is a footer with sections for "Acerca de nuestro sitio", "CENDITEL", "Proyecto SAFET", and "Busque en nuestro sitio".

Figura 8.77: Figura 421: Botón (Enviar)

*figura423:*

---

**Nota: En la figura422 y en la figura423 se define lo siguiente:**

- Observamos los procesos o estados en color amarillo significando que el producto a pasado o recorrido por esos estados
  - Observamos en cada estado o procesos que se muestra un **mensaje**, indicando **rol** de quien ejecuto ese procesos. Aquí se muestra los valores del atributo (**rolfield**) concatenados de la base de datos.
  - En el **mensaje** se muestra la **fecha y hora** del día que se ejecuto el proceso. Aquí se muestra los valores del atributo (**timestampfield**) concatenados a la fecha del computador.
- 

## 5° QUINTO PASO

- Se nos mostrara una tabla, como se muestra en la siguiente *figura424*:

## 6° SEXTO PASO

- Pulsamos en el menú la opción consulta para represarnos a las operaciones, como se muestra en la siguiente *figura425*:

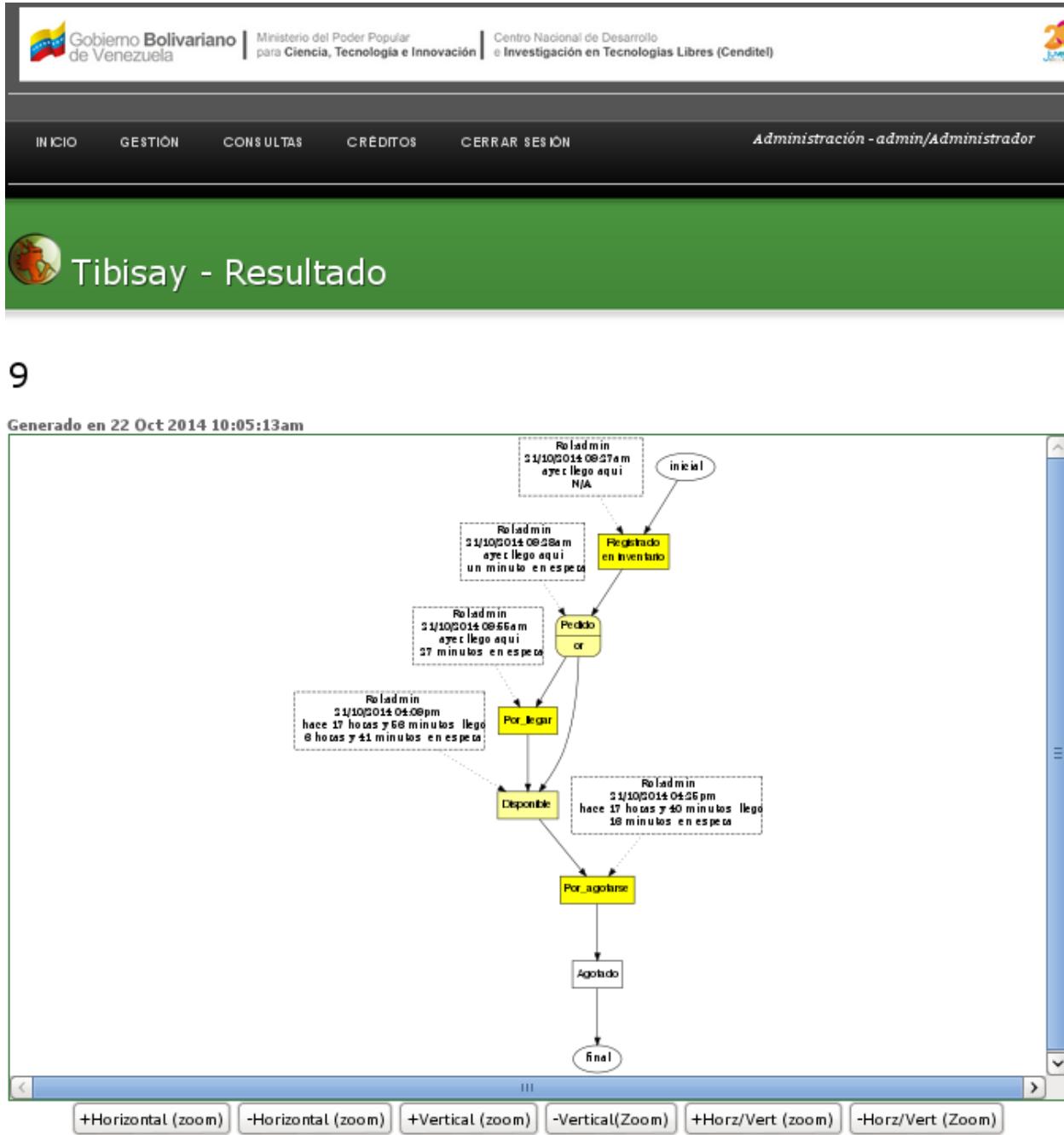


Figura 8.78: Figura 422: Gráfica de seguimiento del productos (Soflan super)

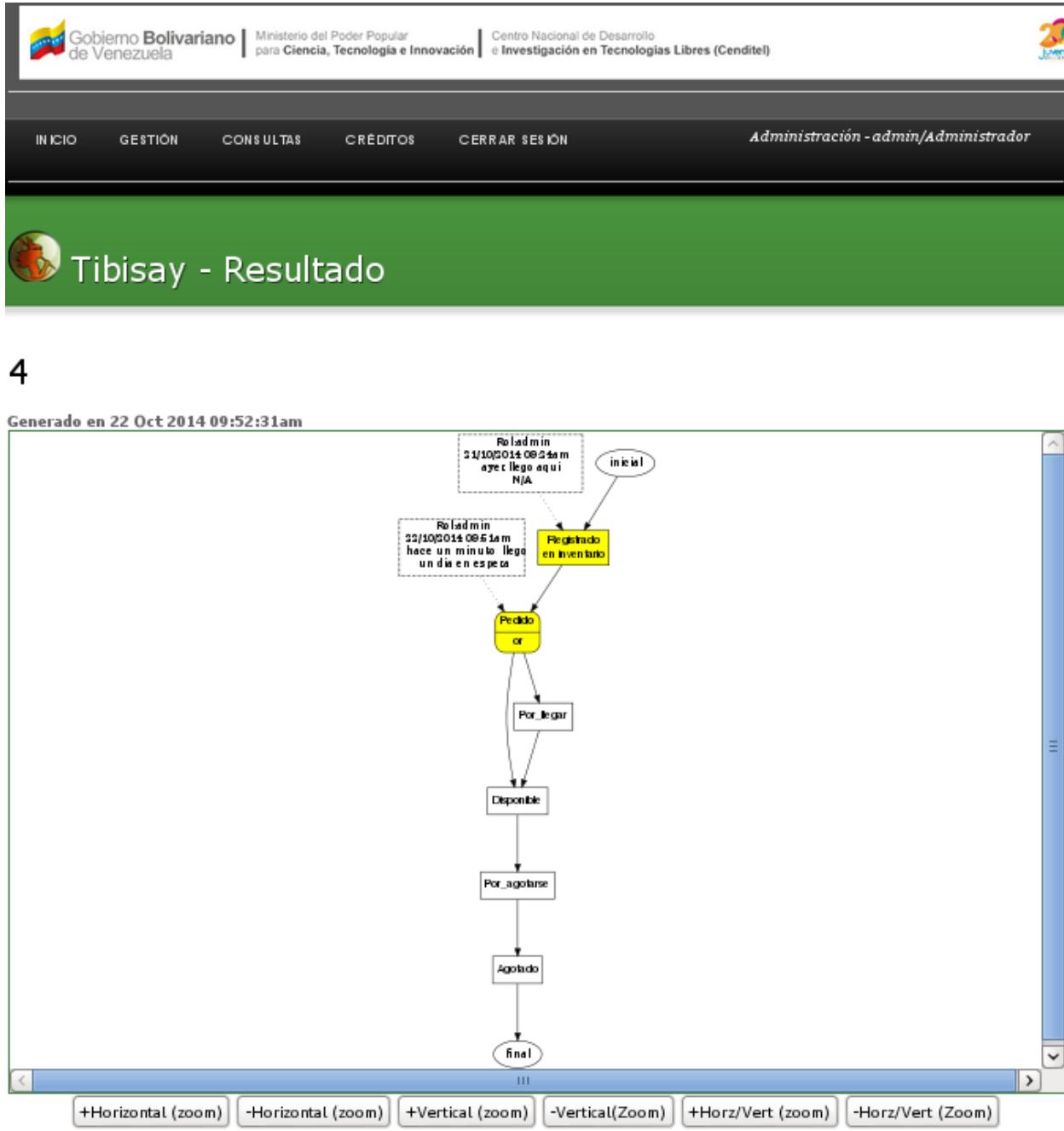


Figura 8.79: Figura 423: Gráfica de seguimiento del productos (Vitamina D)

## Documentación de PySafet, Publicación

Porcentaje (Tareas realizadas): 0%  
Porcentaje (Tareas faltantes): 0%

Search:

<b>Id</b>	<b>Tarea</b>	<b>Rol</b>	<b>Fecha</b>	<b>Tiempo</b>	<b>Espera</b>	<b>Porcentaje</b>	<b>Faltante</b>	<b>Nota</b>
0	Agotado		20/10/2014 02:41pm	n/a	n/a	0%		n/reporte
0	Disponible		20/10/2014 02:41pm	n/a	n/a	0%		n/reporte
0	Pedido		20/10/2014 02:41pm	n/a	n/a	0%		n/reporte
0	Por_agotarse		20/10/2014 02:41pm	n/a	n/a	0%		n/reporte
0	Por_llegar		20/10/2014 02:41pm	n/a	n/a	0%		n/reporte
0	Registrado		20/10/2014 02:41pm	n/a	n/a	0%		n/reporte

Showing 1 to 6 of 6 entries

Acerca de nuestro sitio

Sistema que contribuye a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos

CENDITEL

¿Quiénes somos?

¿Cómo lo haremos?

¿Cuál es nuestra misión?

Proyecto SAFET

Wiki

Gestor del proyecto (Trac)

Busque en nuestro sitio

Buscar

Figura 8.80: Figura 424: Tabla del porcentaje

The screenshot shows the Tibisay - Consultas application. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header is a dark navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar is the text "Administración - admin/Administrador". The main content area has a green header with a small icon and the text "Tibisay - Consultas". The main body contains sections for "Opciones de listado", "Gráficos básicos", "Gráficos de seguimiento", and "Gráficos con filtro", each with associated links.



## Opciones de listado

[Listar datos](#)  
[Listar datos con autofiltro](#)  
[Listar datos con filtro recursivo](#)

## Gráficos básicos

[Generar gráfico coloreado](#)  
[Generar gráfico para clave](#)

## Gráficos de seguimiento

[Generar gráfico de seguimiento](#)

## Gráficos con filtro

[Generar gráfico con autofiltro](#)  
[Generar gráfico con filtro recursivo](#)

The footer section includes four columns: "Acerca de nuestro sitio" (with a link to "Sistema que contribuyendo a la planificación de actividades en el desarrollo de los proyectos. Permite automatizar los procesos que se llevan a cabo en los proyectos."), "CENDITEL" (with links to "¿Quién es somos?", "¿Cómo lo haremos?", "¿Cuál es nuestra misión?", and "¿Qué queremos?"), "Proyecto SAFET" (with links to "Wiki" and "Gestor del proyecto (Trac)"), and a search bar labeled "Busque en nuestro sitio" with a "Buscar" button.

Figura 8.81: Figura 425: Opción consulta



---

## 9.- Tutorial de firma electrónica

---

### 9.1 Conceptos básico

### 9.2 Instalación de los componentes

#### 9.2.1 A.- Biblioteca Libfirmaxml

La biblioteca libfirmaxml provee funciones de firma electrónica sobre el formato BDOC. Es necesario instalar las bibliotecas Libdigidoc y Libdigidocpp en sus versiones 3.6.

Para descargar las bibliotecas Libdigidoc y Libdigidocpp puede ejecutar los siguientes comandos.

```
$ svn export
https://svn.eesti.ee/projektid/idkaart_public/branches/3.6/libdigidoc/
$ svn export
https://svn.eesti.ee/projektid/idkaart_public/branches/3.6/libdigidocpp/
```

#### 9.2.2 B.- Compilar e instalar libdigidoc

Dependencias: gcc, g++, cmake, subversion, libxml2-dev, libssl-dev

```
$ cd libdigidoc
$ cmake .
$ make
# make install (como root)
```

---

**Nota:** Los archivos de encabezado de libdigidoc se instalarán en /usr/local/include/libdigidoc y la biblioteca se instalará en /usr/local/lib/i386-linux-gnu (en el caso de 32 bits) o en /usr/local/lib/x86\_64-linux-gnu (en el caso de 64 bits).

---

#### 9.2.3 C.- Compilar e instalar libdigidocpp

Dependencias: libxerces-c-dev, libxml-security-c-dev, xsdcxx

```
$ cd libdigidocpp  
$ cmake .  
$ make  
# make install (como root)
```

---

**Nota:** Los archivos de encabezado de libdigidoc se instalarán en /usr/local/include/digidocpp y la biblioteca se instalará en /usr/local/lib/i386-linux-gnu (en el caso de 32 bits) o en /usr/local/lib/x86\_64-linux-gnu (en el caso de 64 bits).

---

### 9.2.4 D.- Compilación e instalación de libfirmaxml

Para compilar e instalar libfirmaxml puede descargar los fuentes de la biblioteca con el siguiente comando:

```
$ svn export  
https://seguridad.cenditel.gob.ve/firmaxml/svn/libfirmaxml/trunk  
libfirmaxml
```

Dependencias: qmake-qt4, libqt4-dev

**Para compilar e instalar libfirmaxml:**

```
$ cd libfirmaxml  
$ qmake-qt4  
$ make  
# make install (como root)
```

---

**Nota:** Los archivos de encabezado se instalarán en (/usr/include/libfirmaxml) y la biblioteca compartida en (/usr/lib).

---

### 9.2.5 E.- Compilación e instalación del Servicio Web (HTTPS) FirmaDosPartes

Dependencias: qmake-qt4, libqt4-dev

Este servicio permite desde una petición web firmar producir un documento firmado en formato BDOC. Es IMPORTANTE señalar que por defecto el servicio está disponible por el puerto 5000. Utilizando apache puede redireccionar el puerto.

Para desplegar el servicio debe realizar los siguientes pasos:

#### 1° PRIMER PASO

- Descargue el código del servicio FirmaDosPartes:

```
$ svn export  
https://seguridad.cenditel.gob.ve/firmaxml/svn/firmaDosPartes  
firmaDosPartes
```

## 2° SEGUNDO PASO

- Compile el código utilizando los siguientes comandos:

```
$ cd firmaDosPartes/qhttpserver (ir al directorio fuente del servidor web)
$ qmake-qt4 (crea el archivo de compilación Makefile qhttpserver)
$ make (Compilar qhtserver)
$ cd .. (regresar al directorio firmaDosPartes)
$ qmake-qt4 (crea el archivo de compilación Makefile firmaDosPartes)
$ make (Compilar firmaDosPartes)
```

---

**Nota: Si todo está correcta la compilación debe terminar correctamente.**

---

## 3° TERCER PASO

- Corra el servicio (Corre por defecto por el puerto 5000)

```
$ ./firmaDosPartes
```

---

**Nota: En la cónsola se mostrará el registro (log) de las peticiones**

---

- Si quiere correr el servicio en segundo plano puede escribir:

```
$ ./firmaDosPartes &
```

## 4° CUARTO PASO

- Para terminar el servicio puede ejecutar el siguiente comando

```
$ pkill firmaDosPartes
```

---

**Nota: Los archivos firmados por defecto se almacenan en el directorio (/www/var/media)**

---

## 9.3 Uso de la firma electrónica de manera Web

### 9.3.1 A.- Utilización del servicio Web FirmaDosPartes

Puede utilizar el servicio desde cualquier lenguaje de programación, o incluso desde la línea de comandos (**por ejemplo utilizando la herramienta curl**).

A continuación se explicará desde **Javascript**:

### 1° PRIMER PASO

1.- Enviar el contenido del documento para ser pre-firmado (El pre-firmado consiste en generar un hash, y luego completar la firma). Es requisito tener instalado la librería jquery para hacer uso de la función de llamado web.

```
function createData() {
    // Obtiene el certificado de la tarjeta
    var cert = plugin().getCertificate();

    // Obtiene el certificado en formato PEM para ser transmitido al
    // servidor.
    var mypem = cert.certificateAsPEM;

    try { //Manejo de excepciones
    /*
        $.post.... Consulta post de envío de documentos:
        El primer parámetro es la dirección del servicio Web FirmaDosPartes.

        El segundo parámetro (idaction) es el número de la acción, en este
        caso 1 corresponde con crear la prefirma.

        El tercer parámetro (content) es el texto en plano que se quiere
        firmar se puede enviar un archivo especificando.

        la siguiente sintaxis: texto plano texto <SAFETSEPARATOR/>
        ruta_al_archivo, por ejemplo: "esto es un contrato
        <SAFETSEPARATOR/>/var/www/media/contrato.pdf".

        El cuarto parámetro (id) es el id del documento.

        El quinto parámetro (path) es nombre del archivo firmado en el
        servidor (por defecto se almacenan en /var/www/media).

        El sexto parámetro es el certificado en formato PEM
        (se extrajo antes en la función).

        El código de vuelta (hash) es procesado por la función function(data)
    */

    $.post("https://gestion.cenditel.gob.ve/bdoc/create/testdoc",
    {idaction:1, content:document.getElementById('textareadoc').value,
    id:12,
    path:'oeb2iGqwVg96jG0rpTC8.bdoc',
    pem: mypem
    },

    function(data) { document.getElementById('hash').value = data;
    hasfirst = true; } );

    } catch(e) {

    // Captura la excepción si la hubiese
```

```
}
```

```
}
```

**2.-** La página debe contener una función escrita en javascript (getdocumentid) que genere el nombre del documento firmado, para que pueda ser vinculado con el sistema **Safet**. La función tiene como parámetro (message) el mensaje (documento) que relacionará, también hace uso de la función hex2num. Este mensaje se obtiene concatenando todos los campos del formulario o registro. La función que calcula el nombre del documento es la siguiente:

```
function hex2num (s_hex) {
    var myvalue = parseInt(s_hex,16);

    if (myvalue > 128 ) {
        myvalue = 128 - ( myvalue - 128);
    }
    return myvalue;
}

function getdocumentid(message) {

    var hash = CryptoJS.SHA1(message).toString();
    console.log("HASH SHA1:" + hash);
    var tableid = "\\
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    var output = "";

    for (i = 0; i < hash.length/2; i++) {
        var index = 0;
        var mynum = hex2num(hash.substring(i*2,i*2+2));
        if ( (mynum % 62) > 0 ) {
            index = mynum % 62;
        }
        else {
            index = (mynum % 62)*-1;
        }

        output = output + tableid[ index ].toString();
    }

    console.log("HASH SHA1 OUTPUT: | " + output + "| ");
    return output;
}
```

---

**Nota: Se debe incluir la librería para el manejo de HASH (sha1):**

```
<script
src="https://crypto-js.googlecode.com/svn/tags/3.1.2
```

```
/build/rollups/shal.js">
</script>
```

---

### 2º SEGUNDO PASO

- Realizar la firma del hash (Segunda parte de la firma). Luego de ejecutar la función createData y obtener el hash, debe completar la firma. A continuación se muestra un código de ejemplo (en Javascript).

```
function signFinal() {

    if ( !hasfirst) {
        // Si no ha ejecuta la firma, aparece un mensaje de error
        alert("Debe ejecutar el botón \"(1) Verificar Integridad\""
        antes de ejecutar esta acción");
        return;
    }

    // Limpiar algunos objetos del documento
    document.getElementById('errorMessage').innerHTML = '';
    document.getElementById('errorCode').innerHTML = '0';
    document.getElementById('signature').innerHTML = '';
    var hash = document.getElementById('hash').value;

    try {
        // Obtener el identificador del certificado que está en la tarjeta
        var id = plugin().getCertificate().id;

        /*
        Realizar la firma plugin().sign... Los parámetros son:
        - El primer parámetro (id) es el identificado del certificado
        - El segundo parámetro (hash) es el hash (generado en la prefirma) a ser firmado
        - El tercer parámetro es el lenguaje en que se le muestra el cuadro de diálogo
          al usuario, Disponible inglés, ruso y estonio (es posible agregar español)
        */
        // , { language: 'et' }
        var mysignature = plugin().sign(id, hash, { language: 'en' });

        var s = "";

        /*
        Enviar la firma al servidor para cerrar el documento firmado (bdoc)
        - El primer parámetro es la dirección del sitio Web
        - El segundo parámetro (idaction) es el id de la acción a ejecutar,
          en este caso 2 corresponde con finalizar firma.
        - El tercer parámetro corresponde a la firma generada por el usuario (reseña).
        - La función function procesa los datos devueltos (mensaje de error o de
          éxito de la firma).
        */
    }
}
```

```

$.post("https://gestion.cenditel.gob.ve/bdoc/finalize", {
    idaction:2,signature: mysignature
},
function(data) { alert(data); } );

alert('Firma electrónica realizada exitosamente!');
location.reload(true); // Recargar la página
}
catch (e) {
    alert(plugin.errorMessage); //Mensaje de error
}
}
}

```

### 3° TERCER PASO

- Mostrar datos del certificado. El código de abajo (showData) permite obtener y mostrar información sobre los datos de la tarjeta inteligente que está conectado al computador:

```

function showData() {

    document.getElementById('errorMessage').innerHTML = '';
    document.getElementById('errorCode').innerHTML = '0';
    try {
        var cert = plugin().getCertificate();
        for (var i = 0; i < names.length; i++) {
            name = names[i];
            document.getElementById(name).innerHTML = cert[name];
        }
        document.getElementById('PEM').innerHTML = cert.certificateAsPEM;
        var hex = cert.certificateAsHex;
        var s = "";
        if (hex) for (var i = 0; i < hex.length; i += 64)
            s += hex.substr(i, 64) + "\n";
        document.getElementById('hex').innerHTML = s;
    }
    catch (e) {
        document.getElementById('errorMessage').innerHTML =
        plugin().errorMessage;
        document.getElementById('errorCode').innerHTML =
        plugin().errorCode;
    }
}

```



Figura 9.1: Documentación PySafet.pdf

## Documentación del sistema de Plan Operativos Anuales(POA)

---

### 10.1 1.- Introducción

#### 10.1.1 Proyecto Operativos Anuales (POA)

El sistema (**POA**) se desarrollan diferentes proyectos informático. Los proyectos de este año se toman como base de los proyectos del año anterior.

Se sacan los productos entre esos están el (desarrollo de software, desarrollo de software educativo, dispositivo, jornadas de civilización, metodología entre otros) y se empezó a trabajar a cada uno de esos proyectos agregarle una metodología, es decir cuando el personal valla a trabajar tengan una planificación o una metodología ya definida en el sistema. Ese fue el primer tramo que se hizo desde indicadora.

Se buscaron los productos, se reviso, (se buscaron a los directores y cada uno visible) de cada uno de esos proyectos que tenían ese tipo de productos y ellos lograron obtener como una metodología o una planificación, de lo que se quería realizar para dar respuesta a esa necesidad. Porqué se quiso hacer esto, porque cuando llegaban esos proyectos personal no tenían la mas mínima idea de que se estaba haciendo y hacían varias metodologías, decir realizaban flujos muy grandes y pequeños y no agravan los proyectos (**poa**) y los proyectos (**extraPoa**) pasaban por en sima de esa metodología.

**Qué es (POA):** Son proyectos Plan Operativo Anual, la cual son proyectos que desde el ministerio no evalúa y dan presupuesto de esos proyectos.

Tibisay seguimiento y planificación es un sistema WEB configurado para colaborar en los procesos de planificación, control y seguimiento de los proyectos del Plan Operativo Anual (POA) de la Fundación CENDITEL. Este sistema permite modelar procesos de trabajos que correspondería al desarrollos de los proyectos a través de flujos de trabajos.

Incorporación de la información básica de los proyectos POA tales como, nombre, descripción, objetivo general, objetivos específicos, problemas a enfrentar, acciones específicas, se especifica los grupos responsables del desarrollos de los proyectos.

Incorporación de la planificación representadas en flujo de trabajos, donde cada estado del flujo corresponde a actividades o procesos del desarrollo del proyecto, a cada actividad se le asigna, fecha de ejecución, peso o porcentaje de esa actividad en el proyecto, persona responsable en ejecutarla o coordinar la actividad, permite la modificación para cada actividad el porcentaje asignado, persona responsable,

Permite reportar el avance de las actividades realizadas.

permite visualizar:

- Flujo de trabajo junto con un reporte para cada proyecto donde se especifica, por cada actividad, porcentaje asignado, porcentaje ejecutado, persona responsable, agrega color amarillo a las actividades que han sido ejecutadas, también aparece el porcentaje total ejecutado del proyecto.
- Flujo de trabajo junto con un reporte, donde se muestra la comparación entre las fechas planificadas y ejecutadas, mostrando en cada actividad el color rojo si no se ejecuto la actividad en la fecha planificada, color amarillo si esta cercano la fecha planificada, color blanco si esta muy retirado la fecha.
- Presenta varios reportes, entre los que se encuentra un reporte donde se visualiza:
- Las Acciones Específicas agrupadas por proyectos POA, productos, persona responsable (Cara Visible), Director(a), estado en que se encuentra, fecha de entrega del producto, número de productos, tipo de producto, porcentaje de avance mes a mes, porcentaje total.
- Las Acciones Específicas agrupadas por proyectos POA, productos, persona responsable (Cara Visible), Director(a), estado en que se encuentra, número de productos, tipo de producto, porcentaje total y se agrega un gráfico de barra donde se visualiza el porcentaje total ejecutado.
- Reporte por cada proyecto donde se visualiza todos los registro realizado en el sistema en la ejecución del proyecto.

### WebSAFET

Es un sistema WEB que forma parte del proyecto SAFET, fue configurado para llevar le seguimiento y control de los proyectos POA de la fundación CENDITEL, permite modelar procesos de trabajo que corresponde a la metodología de desarrollo que los grupos de trabajos utiliza en el desarrollo de los proyectos. Los flujos de trabajos que representa los procesos de las metodologías son basados en sencillo en XML.

Web SAFET ofrece:

- Gestión de usuarios y roles
- Ver información en tiempo real
- Reporte, informes y diagramas de flujos
- Modificación de diagramas de flujos gráficamente
- Abierto para incorporar herramientas como la firma electrónica y estampado de tiempo. *Figura 1: Navegador Web:*

El sistema comprende de una sección de autenticación (ver figura 2) esta sección tiene la posibilidad de habilitar autenticación usando tarjetas inteligentes. *Figura 2. Sección de autenticación:*

Se han definidos roles de acuerdo a las responsabilidades que tienen los usuarios del sistema, por ejemplo: el cara visible o responsable del proyecto de sensibilización solo puede ver y modificar información referente al proyecto. El sistema cuenta con un módulo de registro donde almacena información de las operaciones realizada por el usuario autenticado.

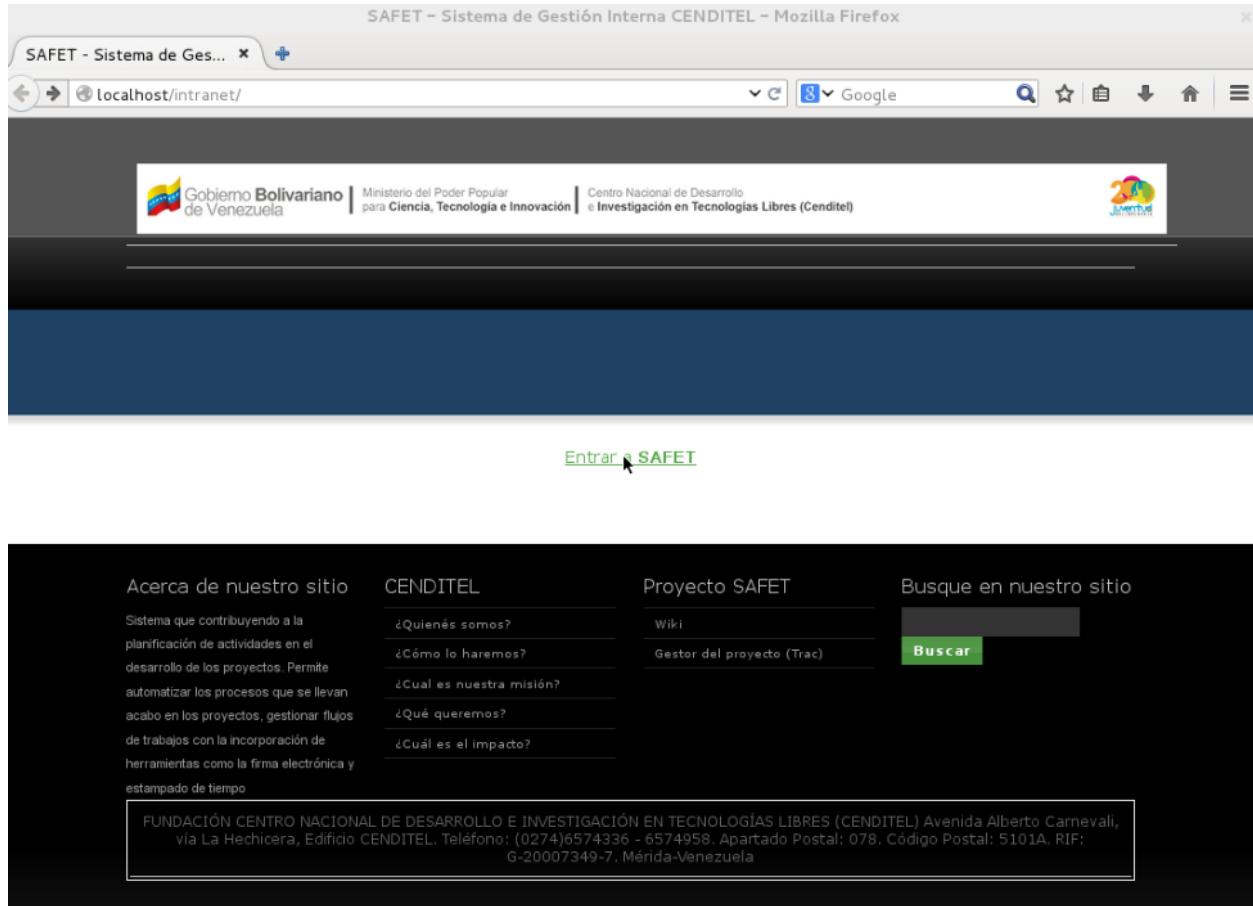


Figura 10.1: Figura 1: Navegador Web

## Documentación de PySafet, Publicación

Nombre de usuario: admin

Contraseña: .....

Recordar mis datos

Iniciar Sesión

Registrarse

Sistema Automatizado para la Firma y Estampillado de Tiempo SAFET,

Figura 10.2: Figura 2. Sección de autenticación

El sistema despliega los procesos de la metodología de desarrollo en flujo de datos (ver figura 3) donde se puede obtener información entre las que tenemos:

- Estatus del proyecto
- Fecha de modificación de los procesos
- Porcentajes ejecutado hasta el momento de la consulta. *Figura 3. Flujo de trabajo:*

Entre los reporte que generar el sistema se cuenta con uno donde se muestra: (ver figura 4 )

- Los Productos por acción específica.
- Porcentaje de avance de los productos mes a mes.
- Porcentaje de avance de la acción específica.
- Estatus del producto.
- Fecha de entrega de los productos
- Tipo de producto
- Descripción de la actividad de cada producto
- Persona del grupo de trabajo asignada como responsable del producto
- Persona asignada por la presidencia como responsable del producto

*Figura 4. Flujo de trabajo:*

## SAFET - Reportes / Gráficos

Reporte de Actividades por Acción Específica (14 actividades )

Acción: Creación de contenidos que propicien la reflexión sobre el quehacer científico-tecnológico y la apropiación social del conocimiento de las organizaciones comunitarias.

id	actividad	status	responsable	presiden	entrega	num	tipo_prod	ene(0.0%)	feb(1.3%)	mar(5.0%)	abr(8.8%)	may(15.0%)	jun(5.0%)	jul(7.3%)	ago(6.3%)	sep(16.0%)	oct(15.0%)
								0	0	0	10	10	20	0	0	45	5
57	Formulación y divulgación de un planteamiento teórico y crítico sobre la noción dominante de la neutralidad de la ciencia.	Construcion_modelo_p	González, Carlos	Roca, Santiago	III trimestre	2	Publicación	0	0	0	10	10	20	0	0	45	5
70	Enfoque educativo para la transferencia de poder a las comunidades	Planificacion_experienci	Contreras, José Joaquín	Roca, Santiago	III trimestre	1	Publicación	0	0	0	25	0	0	24	0	0	31
56	Sensibilización comunitaria para la generación y apropiación social del conocimiento como bien público, con miras a la construcción del Estado Comunal.	Produccion	Roca, Santiago	Montilla, Maricela	III trimestre	1	Publicación	0	0	20	0	20	0	5	0	5	20
66	Sensibilización (Apropiación) comunitaria para la generación y apropiación social del conocimiento como bien público, con miras a la construcción del Estado Comunal.	Correccion_textos	Montilla, Maricela	Roca, Santiago	III trimestre	1	Publicación	0	5	0	0	30	0	0	25	15	5

Figura 10.3: **Figura 3. Flujo de trabajo**

El sistema a permitido:

- Mejorar el control del avance de los proyectos
- Aumentar el número de actividades y productos terminados
- Aplicar metodologías en el desarrollo de los proyecto
- Automatizar las actividades de reporte y rendición de cuantas

### 10.1.2 Gestión (Línea estratégica)

En **CENDITEL** se trabaja con varias líneas estratégicas, entre esas tenemos (**InfoGobierno, Educativo, Mapa Industrial**). Las líneas estratégicas pueden cambiar en cada año en los ministerios e instituciones y que permita cambiarla en el sistema.

En las **Línea estratégica** tenemos tres secciones (**Agregar línea estratégica, Modificar línea estratégica, Borrar línea estratégica**), como se muestra en la siguiente *Figura 5: Líneas Estratégicas*:

#### Agregar línea estratégica

En esta opción tenemos los campos (**Nombre de la línea estratégica\***) y (**Descripción de la línea estratégica\***) como se muestra en la siguiente *Figura 6: Agregar (Líneas Estratégicas)*:

## SAFET - Reporte / Gráfico

71

Generado 24 Nov 2011 09:48:32am

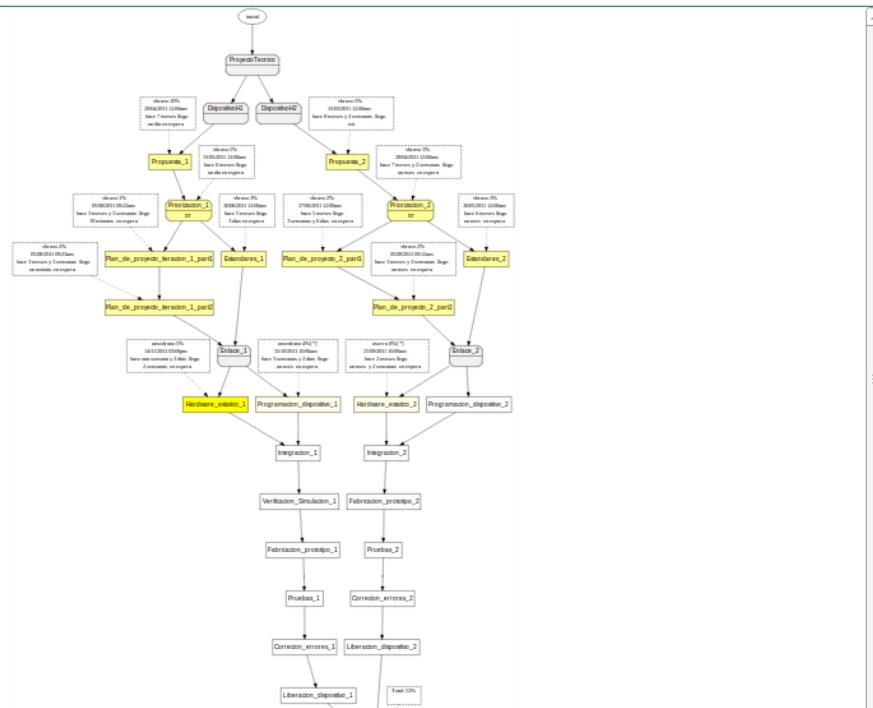


Figura 10.4: Figura 4. Flujo de trabajo

Figura 10.5: Figura 5: Líneas Estratégicas

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Youth Foundation logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session status message. The main content area has a dark blue header with the title "Tibisay - Gestión" and a small icon. Below this, a section titled "Agregar línea estratégica" contains two input fields: one for the "Nombre de la línea estratégica \*" and another for the "Descripción de la línea estratégica \*". At the bottom of this section are two buttons: "Enviar" and "Limpiar".

Figura 10.6: **Figura 6: Agregar (Líneas Estratégicas)**

### Modificar línea estratégica

En esta opción tenemos los campos (**Línea estratégica \***), (**Nombre de la línea estratégica \***), (**Descripción de la línea estratégica \***), (**Estatus de la línea estratégica \***), como se muestra en la siguiente *Figura 7: Modificar (Líneas Estratégicas)*:

**Nota:** En la Línea estratégica tenemos (**InfoGobierno, Educativo, Mapa Industrial y ExtraPOA**). El proyecto **ExtraPOA**

1. **InfoGobierno:** Son todas aquellos proyectos P0A que apunta a esta línea estratégicas.
2. **ExtraPOA:** Son todos aquellos proyectos que no están en (**Plan Operativo Anual**)
  - En el Estatus de la línea estratégica tenemos (**Activo y Inactivo**), ya se sacan reportes por año.

### Borrar línea estratégica

En esta opción tenemos los campos (**Línea estratégica\***), (**Nombre de la línea estratégica\***), (**Descripción de la línea estratégica\***), (**Estatus de la línea estratégica\***) como se muestra en la siguiente *Figura 8: Borrar (Líneas Estratégicas)*:

## Documentación de PySafet, Publicación

Línea estratégica \*

Nombre de la línea estratégica \*

Descripción de la línea estratégica \*

Estatus de la línea estratégica \*

**Enviar** **Limpiar**

InfoGobierno  
Educativo  
Mapa\_Industrial  
ExtraPOA

Figura 10.7: Figura 7: Modificar (Líneas Estratégicas)

Borrar línea estratégica

Línea estrategica \*

**Enviar** **Limpiar**

InfoGobierno  
Educativo  
Mapa\_Industrial  
ExtraPOA

Figura 10.8: Figura 8: Borrar (Líneas Estratégicas)

### 10.1.3 Gestión (Tipo de Productos)

En los tipo de productos que se desarrollan en **CENDITEL** son:

- Desarrollo de software
- Desarrollo de software educativo
- Mapas productivo
- Jornadas de civilización
- Publicación

En los **tipos de productos** tenemos cuatro secciones (**Agregar tipo de producto, Modificar tipo de producto, Borrar tipo de producto, Cargar flujo de trabajo**), como se muestra en la siguiente *Figura 9: Tipo de Productos*:

The screenshot shows a web-based administrative interface for 'Tibisay - Gestión'. At the top, there's a header bar with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the CENDITEL logo, and a user session indicator ('Usuario de Administrador [admin/Administrador]'). Below the header is a dark navigation bar with links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. The main content area has a blue header titled 'Tibisay - Gestión' with a small globe icon. Below this, there's a section titled 'Líneas Estratégicas' with a bar chart icon, and another titled 'Tipo de Productos' with a bar chart icon. Both sections contain green links for 'Agregar', 'Modificar', 'Borrar', and 'Cargar' actions related to their respective categories.

Figura 10.9: Figura 9: Tipo de Productos

#### Agregar Tipo de Producto

En esta opción tenemos los campos (**Tipo producto\***), (**Tipo producto\***), (**Nombre del indicador requerimiento\***), (**Nombre del indicador producto\***), (**Cargar archivo flujo\***) como se muestra en la siguiente *Figura 10: Agregar (Tipo de Producto)*:

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header with the Venezuelan Government logo, the text 'Gobierno Bolivariano de Venezuela', 'Ministerio del Poder Popular para Ciencia, Tecnología e Innovación', 'Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)', and the 'Juventud' logo. Below the header, a navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. A user session message 'Usuario de Administrador [admin/Administrador]' is also present. The main content area has a dark blue header with the text 'Tibisay - Gestión' and a small globe icon. Below this, the page title 'Agregar tipo de producto' is displayed. The form consists of several input fields: 'Tipo producto \*' with a text input field; 'Descripción \*' with a large text area; 'Nombre del indicador requerimiento \*' with a text input field; 'Nombre del indicador producto \*' with a text input field; and 'Cargar archivo flujo \*' with a file upload input field. At the bottom left of the form are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.10: Figura 10: Agregar (Tipo de Producto)

## Modificar Tipo de Producto

En esta opción tenemos los campos (**Id\***), (**Tipo producto\***), (**Descripción\***), (**Nombre del indicador requerimiento\***), (**Nombre del indicador producto\***), (**Cargar archivo flujo\***), (**Estatus tipo producto\***) como se muestra en la siguiente *Figura 11: Modificar (Tipo de Producto)*:

The screenshot shows a web-based application interface. At the top, there is a header with the logo of the Government of Venezuela, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cendite). Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Usuario de Administrador [admin/Administrador]". The main title "Tibisay - Gestión" is displayed above the form. The form itself is titled "Modificar tipo de producto". It contains several input fields: "Id \*", "Tipo producto \*", "Descripción \*", "Nombre del indicador requerimiento \*", "Nombre del indicador producto \*", "Cargar archivo flujo \*", and "Estatus tipo producto \*". At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Figura 10.11: **Figura 11: Modificar (Tipo de Producto)**

## Borrar Tipo de Producto

En esta opción tenemos los campos (**Id\***) como se muestra en la siguiente *Figura 12: Borrar (Tipo de Producto)*:

The screenshot shows the Tibisay - Gestión application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session message indicates "Usuario de Administrador [admin/Administrador]". The main content area has a blue header with a small globe icon and the text "Tibisay - Gestión". Below this, a sub-header says "Borrar tipo de producto". A dropdown menu is open over a text input field labeled "Id \*". The dropdown contains the following options: (14) Software\_libre, (15) Software\_educativo, (16) Dispositivos, and (17) Metodologías. A button labeled "Enviar" is located at the bottom left of the dropdown.

Figura 10.12: **Figura 12: Borrar (Tipo de Producto)**

### Cargar flujo de trabajo

En esta opción tenemos los campos (**Nombre\***) como se muestra en la siguiente *Figura 13: Cargar flujo de trabajo*:

---

**Nota:** El flujo donde aparece la metodología o actividades que se deben realizar para dar cumplimiento a ese producto como tal

---

#### 10.1.4 Gestión (Indicadores de procesos)

En los **indicadores de procesos** tenemos tres secciones (**Inicializar indicador de procesos**, **Modificar indicador de procesos**, **Borrar indicador de procesos**), como se muestra en la siguiente *Figura 14: Indicadores de procesos*:

#### Inicializar indicador de procesos

En esta opción tenemos los campos (**Flujo\***), (**Tarea\***) y (**Indicador**) como se muestra en la siguiente *Figura 15: Inicializar (Indicador de procesos)*:

The screenshot shows the Tibisay - Gestión web application. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Youth Ministry logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session message indicates the user is an administrator [admin/Administrador]. The main content area has a blue header with the text "Tibisay - Gestión". Below this, a section titled "Cargar flujo de trabajo" contains a file input field with the placeholder "sin seleccionar". A "Examinar..." button is next to it, and a message states "No se ha seleccionado ningún archivo.". At the bottom of this section are two buttons: "Enviar" and "Limpiar".

Figura 10.13: **Figura 13: Cargar flujo de trabajo**

### Modificar indicador de procesos

En esta opción tenemos los campos **(Id\*)**, **(Responsable)**, **(Porcentaje tarea)** y **(Descripción tarea)** como se muestra en la siguiente *Figura 16: Modificar (Indicador de procesos)*:

### Borrar indicador de procesos

En esta opción tenemos los campos **(Número de actividad\*)** como se muestra en la siguiente *Figura 17: Borrar (Indicador de procesos)*:

### 10.1.5 Gestión (POA)

En POA tenemos tres secciones **(Agregar proyecto POA, Modificar proyecto POA y Borrar proyecto POA)**, como se muestra en la siguiente *Figura 18: Indicadores de procesos*:

### Agregar proyecto POA

En esta opción tenemos los campos **(Nombre\*)**, **(Código proyecto POA\*)**, **(Línea estratégica\*)**, **(Descripción\*)**, **(Objetivo General \*)**, **(Objetivo Específico\*)** y **(Problema a enfrentar\*)**, como se muestra en la siguiente *Figura 19: Agregar (proyecto POA)*:



## Líneas Estratégicas

Agregar línea estratégica  
Modificar línea estratégica  
Borrar línea estratégica

## Tipo de Productos

Agregar tipo de producto  
Modificar tipo de producto  
Borrar tipo de producto  
Cargar flujo de trabajo

## Indicadores de procesos

Inicializar indicador de procesos  
Modificar indicador de procesos  
Borrar indicador de procesos

Figura 10.14: Figura 14: Indicadores de procesos

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Jovenes en la Red logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user status message: "Usuario de Administrador [admin/Administrador]". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small icon of a tree. Below this, a section titled "Inicializar indicador de procesos" contains three input fields: "Flujo \*", "Tarea \*", and "Indicador", each with a dropdown menu. At the bottom of this section are two buttons: "Enviar" and "Limpiar".

Figura 10.15: **Figura 15: Inicializar (Indicador de procesos)**

### Modificar proyecto POA

En esta opción tenemos los campos (**Nombre proyecto\***), (**Nombre\***), (**Código proyecto POA\***), (**Línea estratégica\***), (**Descripción\***), (**Objetivo General\***), (**Objetivo Específico\***) y (**Problema a enfretar\***) como se muestra en la siguiente *Figura 20: Modificar (proyecto POA)*:

### Borrar proyecto POA

En esta opción tenemos los campos (**Nombre proyecto\***) como se muestra en la siguiente *Figura 21: Borrar (proyecto POA)*:

### 10.1.6 Gestión (Acciones Específicas)

En las **Acciones Específicas** tenemos tres secciones (**Agregar acción específica**, **Modificar acción específica** y **Borrar acción específica**), como se muestra en la siguiente *Figura 22: Acciones Específicas*:

## Documentación de PySafet, Publicación

The screenshot shows a web-based application with a dark blue header bar. On the left is the logo of the Government of Venezuela. In the center, it says "Gobierno Bolivariano de Venezuela | Ministerio del Poder Popular para Ciencia, Tecnología e Innovación | Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)". On the right is the "Juventud" logo. Below the header, there's a navigation menu with links: INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the menu, it says "Usuario de Administrador [admin/Administrador]". The main content area has a title "Tibisay - Gestión" with a small icon. Below it, the page title is "Modificar indicador de procesos". The form contains several input fields: "Id \*" with a dropdown arrow, "Responsable" with a dropdown arrow, "Porcentaje tarea" with a dropdown arrow, and a large "Descripción tarea" text area with a "..." button. At the bottom are two buttons: "Enviar" and "Limpiar".

Figura 10.16: Figura 16: Modificar (Indicador de procesos)

This screenshot shows the same application interface as Figure 10.16. The title "Tibisay - Gestión" is at the top. The page title is "Borrar indicador de procesos". The form includes a "Número de actividad \*" field with a dropdown arrow. A dropdown menu is open, listing numbered options from 7 to 13, each corresponding to a process step: (7) Planificación, (8) Conceptualización, (9) Planificación, (10) Elaboración\_de\_manuales, (11) Definir\_la\_arquitectura\_de\_la\_aplicación, (12) Validación, and (13) Publicación. At the bottom of the form are "Enviar" and "Limpiar" buttons.

Figura 10.17: Figura 17: Borrar (Indicador de procesos)

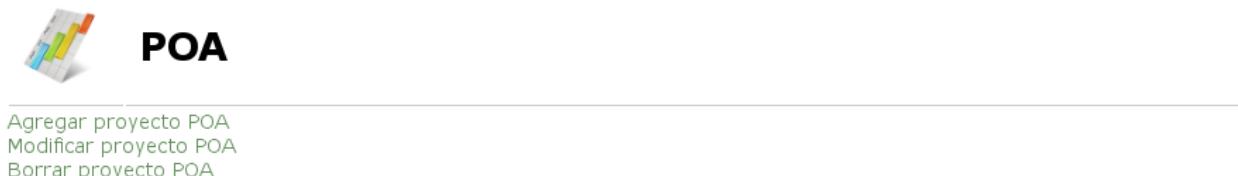


Figura 10.18: Figura 18: Indicadores de procesos

The screenshot shows a form for adding a POA project. The fields are:

- Nombre \*: Nombre del Proyecto POA
- Código proyecto POA \*: Código del proyecto POA a agregar
- Línea estratégica \*: A dropdown menu.
- Descripción \*: A large text area with a '...' button.
- Objetivo General \*: A large text area with a '...' button.
- Objetivo Específico \*: A large text area with a '...' button.
- Problema a enfrentar \*: A large text area with a '...' button.

At the bottom are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.19: Figura 19: Agregar (proyecto POA)

## Documentación de PySafet, Publicación

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user status message: "Usuario de Administrador [admin/Administrador]". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small globe icon. Below this, the page title is "Modificar proyecto POA". The form consists of several input fields: "Nombre proyecto \*", "Nombre \*", "Nombre del Proyecto POA", "Código proyecto POA \*", "Nombre de la acción específica", "Línea estratégica \*", "Descripción \*", "Objetivo General \*", "Objetivo Específico \*", and "Problema a enfrentar \*". Each field is accompanied by a text input box. At the bottom of the form are two buttons: "Enviar" and "Borrar".

Figura 10.20: Figura 20: Modificar (proyecto POA)

The screenshot shows a web interface for 'Tibisay - Gestión'. At the top, there's a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. A user session indicator shows 'Usuario de Administrador [admin/Administrador]'. The main content area has a blue header with a circular icon and the text 'Tibisay - Gestión'. Below this, a section titled 'Borrar proyecto POA' contains a form. The form includes a text input field labeled 'Nombre proyecto \*', two buttons ('Enviar' and 'Limpiar'), and a dropdown menu with the following options: '( 20 ) Proyecto POA 1', '( 21 ) Proyecto POA 2', '( 22 ) Proyecto POA 3', and '( 23 ) Proyecto Extra POA'. The dropdown menu is open, and the option '( 20 ) Proyecto POA 1' is highlighted.

Figura 10.21: Figura 21: Borrar (proyecto POA)

## Acciones Específicas

Agregar acción específica  
Modificar acción específica  
Borrar acción específica

Figura 10.22: Figura 22: Acciones Específicas

### Agregar acción específica

En esta opción tenemos los campos (**Proyecto POA\***), (**Nombre\***) y (**Código\***) como se muestra en la siguiente *Figura 23: Agregar (acción específica)*:

The screenshot shows the Tibisay - Gestión application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session indicator. The main content area has a blue header titled 'Tibisay - Gestión'. Below the header, the page title 'Agregar acción específica' is displayed. The form consists of three input fields: 'Proyecto POA \*' with a dropdown menu, 'Nombre \*' with a text input field, and 'Código \*' with a text input field labeled 'Código de la actividad a agregar'. At the bottom of the form are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.23: **Figura 23: Agregar (acción específica)**

### Modificar acción específica

En esta opción tenemos los campos (**Nombre acción específica\***), (**Nombre\***), (**Código\***) y (**Proyecto POA\***) como se muestra en la siguiente *Figura 24: Modificar (acción específica)*:

### Borrar acción específica

En esta opción tenemos los campos (**Nombre acción específica\***) como se muestra en la siguiente *Figura 25: Borrar (acción específica)*:

The screenshot shows the Tibisay - Gestión web application interface. At the top, there are navigation links: INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of these is the text "Usuario de Administrador [admin/Administrador]". The header also features the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the CENDITEL logo. Below the header, the page title "Tibisay - Gestión" is displayed next to a globe icon. The main content area is titled "Modificar acción específica". It contains several input fields: "Nombre acción específica \*", "Nombre \*", "Código \*", and "Proyecto POA \*". Each field has a corresponding dropdown menu to its right. At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Figura 10.24: Figura 24: Modificar (acción específica)

The screenshot shows the Tibisay - Gestión web application interface. The layout is identical to Figure 10.24, with the same navigation links, header, and title. The main content area is titled "Borrar acción específica". It contains an input field "Nombre acción específica \*" with a dropdown menu open, showing a list of specific actions: "(85) Acción específica N 12, (84) Acción específica N 11, (83) Acción específica N 10, (82) Acción específica N 9, (81) Acción específica N 8, (80) Acción específica N 7, (79) Acción específica N 6, (78) Acción específica N 5, (77) Acción específica N 4, (76) Acción específica N 3, (75) Acción específica N 2, (74) Acción específica N 1.". Below the input field are "Enviar" and "Limpiar" buttons. At the bottom of the page, there are links for "Acerca de nuestro sitio", "CENDITEL", and "Busque en nuestro sitio".

Figura 10.25: Figura 25: Borrar (acción específica)

### 10.1.7 Gestión (Trabajadores)

En los **Trabajadores** tenemos tres secciones (**Agregar trabajador**, **Modificar trabajador** y **Borrar trabajador**), como se muestra en la siguiente *Figura 26: Indicadores de procesos*:



Figura 10.26: **Figura 26: Indicadores de procesos**

#### Agregar trabajador

En esta opción tenemos los campos (**Cédula\***), (**Nombres\***), (**Apellidos\***) y (**Cargo asignado\***) como se muestra en la siguiente *Figura 27: Agregar trabajador*:

The screenshot shows a top navigation bar with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, and the Cenditel logo. It also shows the user 'Usuario de Administrador [admin/Administrador]'. Below this is a dark header with the text 'Tibisay - Gestión' and a small globe icon. The main form area has a light blue background. It contains four input fields: 'Cédula \*' (with a placeholder box), 'Nombres \*' (with a placeholder box), 'Apellidos \*' (with a placeholder box), and 'Cargo asignado \*' (with a dropdown menu). At the bottom are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.27: **Figura 27: Agregar trabajador**

## Modificar trabajador

En esta opción tenemos los campos (**Nombre acción específica\***), (**Nombre\***), (**Código\***) y (**Proyecto POA\***) como se muestra en la siguiente *Figura 28: Modificar trabajador*:

The screenshot shows a web interface for modifying a specific action. At the top, there's a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditec logo. The main title is "Tibisay - Gestión". Below it, the page title is "Modificar acción específica". The form consists of four input fields with dropdown menus: "Nombre acción específica \*", "Nombre \*", "Código \*", and "Proyecto POA \*". At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Figura 10.28: **Figura 28: Modificar trabajador**

## Borrar trabajador

En esta opción tenemos los campos (**Cédula\***) como se muestra en la siguiente *Figura 29: Borrar trabajador*:

### 10.1.8 Gestión (Equipos)

En los **Equipos** tenemos tres secciones (**Agregar equipo**, **Modificar equipo**, **Borrar equipo**, **Asignar personal a equipo** y **Borrar asignación de trabajador a equipo**), como se muestra en la siguiente *Figura 30: Equipos*:

The screenshot shows a web application interface. At the top, there are logos for the Government of Venezuela, the Ministry of Popular Power for Science, Technology, and Innovation, and the National Center for Development and Research in Free Technologies (Cenditel). The menu bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session message indicates "Usuario de Administrador [admin/Administrador]". Below the menu, the title "Tibisay - Gestión" is displayed next to a logo. A sub-menu titled "Borrar trabajador" is open, showing a dropdown list of names and IDs. The list includes:

- V10715251-Aura, Yzarra S.
- V11468638-Carlos R., González M.
- V11914925-Alexander A., Olivares V.
- V11952453-Rodolfo L., Sumoza M.
- V11952892-Elisabeth V., Benítez R.
- V11954451-Juan E., Vizcarrondo R.
- V12048084-Roldán D., Vargas G.
- V12390174-Agustín, Marcos A.
- V12486780-Yuleici, Verdi M.
- V12643114-Solázver, Solé Alvarez
- V12777920-Gusmery C., Paredes
- V12778889-Antoni G., Araujo B.
- V12780545-José Javier, Pérez Angulo
- V12797664-Victor R., Bravo B.
- V12997071-José V., Castillo R.
- V13499949-José L., Moreno A.
- V13500248-Tanger A., Rivas C.
- V13648820-Maryorie, Varela
- V13966598-Endira J., Mora R.

On the left side, there is a sidebar with sections for "Acerca de nuestro sistema" and "Proyecto SAFET". On the right side, there is a search bar labeled "Busque en nuestro sitio" with a "Buscar" button. The main content area has a dark blue header with the title "Tibisay - Gestión".

Figura 10.29: Figura 29: Borrar trabajador



## Equipos

Agregar equipo  
Modificar equipo  
Borrar equipo  
Asignar personal a equipo  
Borrar asignación de trabajador a equipo

Figura 10.30: Figura 30: Equipos

## Agregar equipo

En esta opción tenemos los campos (**Nombre\***) y (**Descripción\***) como se muestra en la siguiente *Figura 31: Agregar equipo*:

Figura 10.31: **Figura 31: Agregar equipo**

## Modificar equipo

En esta opción tenemos los campos (**Id\***), (**Nombre**), (**Descripción**), (**Fecha creación**) y (**Fecha desincorporación**) como se muestra en la siguiente *Figura 32: Modificar equipo*:

## Borrar equipo

En esta opción tenemos los campos (**Id\***) como se muestra en la siguiente *Figura 33: Borrar equipo*:

## Asignar personal a equipo

En esta opción tenemos los campos (**Equipo responsable\***) y (**Personal a agregar\***) como se muestra en la siguiente *Figura 34: Asignar personal a equipo*:

The screenshot shows a web application interface for managing teams. At the top, there is a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Juventud 2040 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. The user is identified as 'Usuario de Administrador [admin/Administrador]'. The main content area has a dark blue header with the text 'Tibisay - Gestión' and a globe icon. Below this, the page title is 'Modificar equipo'. The form consists of several input fields: 'Id \*' with a dropdown menu; 'Nombre' with a text input field; 'Descripción' with a large text area; 'Fecha creación' and 'Fecha desincorporación' with date input fields; and two buttons at the bottom labeled 'Enviar' and 'Limpiar'.

Figura 10.32: **Figura 32: Modificar equipo**

The screenshot shows a web interface for 'Tibisay - Gestión'. At the top, there's a header with the Venezuelan Government logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Cenditel logo. Below the header, a navigation bar includes links for 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. A user session indicator shows 'Usuario de Administrador [admin/Administrador]'. The main content area has a dark blue header with a circular icon and the text 'Tibisay - Gestión'. Below this, a section titled 'Borrar equipo' contains a form field labeled 'Id \*' with a dropdown menu open. The dropdown lists various items, many of which start with '(6...)' followed by a short description. To the right of the form, there's a sidebar with 'Acerca de nuestro sitio' and a list of projects. At the bottom right, there's a search bar with 'Proyecto SAFET' and 'Busque en nuestro sitio'.

Id *	Descripción
(71)	Gestión_interna
(70)	Conocimiento_libre
(69)	Indicadores
(68)	Plataforma_comunicacional_pagina_web
(67)	Privacidad_seguridad_socialista
(66)	Autana
(65)	Infraestructura
(64)	Calidad
(63)	Tibisay_Seguridad
(62)	Contraloría_Social
(61)	Planificación_Estratégica
(60)	Mapa_Agrícola
(59)	HAPA
(58)	TDA_Hardware
(57)	SIG
(56)	ECOALBA
(55)	Simulación
(54)	TDA_Educativo
(53)	Publicaciones

Figura 10.33: Figura 33: Borrar equipo

## Documentación de PySafet, Publicación

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Youth logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session status message. The main content area has a dark blue header with the text "Tibisay - Gestión". Below this, a section titled "Asignar personal a equipo" contains two dropdown menus labeled "Equipo responsable \*" and "Personal a agregar \*". At the bottom of this section are two buttons: "Enviar" and "Limpiar".

Figura 10.34: **Figura 34: Asignar personal a equipo**

### Borrar asignación de trabajador a equipo

En esta opción tenemos los campos (**Clave\***) como se muestra en la siguiente *Figura 35: Borrar asignación*:

#### 10.1.9 Gestión (Productos)

En los **Productos** tenemos tres secciones (**Iniciar producto**, **Modificar producto** y **Borrar producto**), como se muestra en la siguiente *Figura 36: Productos*:

#### Iniciar producto

En esta opción tenemos los campos (**Acción específica que pertenece**), (**Tipo producto\***), (**Nombre del producto\***), (**Id equipo responsable\***), (**Cara visible\***), (**Director\***), (**Origen\***), (**Cargar documento**) como se muestra en la siguiente *Figura 37: Iniciar producto*:

The screenshot shows a web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Juventud logo. Below the header, there are navigation links: INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right, it says "Usuario de Administrador [admin/Administrador]". The main title is "Tibisay - Gestión". Below the title, a sub-section title is "Borrar asignación de trabajador a equipo". A dropdown menu is open, showing a list of names and their associations, such as "Alberto Medrano V.->Medios\_Comunitarios\_Socialista" and "Yuleici Verdi M.->Medios\_Comunitarios\_Socialista". The background of the page includes sections for "Acerca de nuestro proyecto SAFET" and "Busque en nuestro sitio".

Figura 10.35: Figura 35: Borrar asignación

## Productos

[Inicializar producto](#)  
[Modificar producto](#)  
[Borrar producto](#)

Figura 10.36: Figura 36: Productos

The screenshot shows a web-based application interface for product initialization. At the top, there is a header bar with the Venezuelan Government logo, the Ministry of Science, Technology, and Innovation, the Cenditel logo, and a user session indicator. Below the header, a navigation menu includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. The main content area is titled "Tibisay - Gestión" and features a sub-section titled "Inicializar producto". The form contains several input fields: "Acción específica que pertenece" (dropdown), "Tipo producto \*" (dropdown), "Nombre del producto \*" (text input with placeholder "Nombre del producto a agregar"), "Id equipo responsable \*" (dropdown), "Cara visible \*" (dropdown), "Director \*" (dropdown), "Origen \*" (dropdown), and a file upload section for "Cargar documento" with a status message "sin seleccionar" and a "Examinar..." button. At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Inicializar producto

Acción específica que pertenece

Tipo producto \*

Nombre del producto \*

Nombre del producto a agregar

Id equipo responsable \*

Cara visible \*

Director \*

Origen \*

Cargar documento

sin seleccionar

Examinar... No se ha seleccionado ningún archivo.

Enviar Limpiar

Figura 10.37: Figura 37: Inicializar producto

## Modificar producto

En esta opción tenemos los campos (**Nombre producto\***), (**Tipo producto**), (**Producto\***), (**Nombre de la acción específica a la que pertenece**), (**Entrega en el trimestre**), (**Entrega en el trimestre**), (**Equipo responsable**), (**Cara visible**), (**Director**), (**Estado actual**) y (**Porcentaje actual**) como se muestra en la siguiente *Figura 38: Modificar producto:*

The screenshot shows a web application interface for modifying a product. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Juventud 2000 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN, along with a user session indicator. The main title 'Tibisay - Gestión' is displayed above the form. The form itself has ten input fields, each with a label and a dropdown menu. The labels are: 'Nombre producto \*', 'Tipo producto', 'Producto \*' (with a text input field for 'Nombre del producto'), 'Nombre de la acción específica a la que pertenece', 'Entrega en el trimestre', 'Fecha entrega', 'Equipo responsable', 'Cara visible', 'Director', 'Estado actual', and 'Porcentaje actual'. At the bottom of the form are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.38: **Figura 38: Modificar producto**

### Borrar producto

En esta opción tenemos los campos (**Nombre producto\***) como se muestra en la siguiente *Figura 39: Borrar producto*:

The screenshot shows a web-based application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the National Center for Development and Research in Free Technologies (Cenditel). Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session indicator shows "Usuario de Administrador [admin/Administrador]". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small globe icon. Below this, a white form section is titled "Borrar producto". It contains a text input field labeled "Nombre producto \*", two buttons labeled "Enviar" and "Limpiar", and a dropdown menu open over the input field. The dropdown menu lists the following options:

- (140 )Producto N 1
- (141 )Producto N 2
- (142 )Producto N 3
- (143 )Producto N 4
- (144 )Producto N 5
- (145 )Producto N 6
- (146 )Producto N 7
- (147 )Producto N 8
- (148 )Producto N 9

Figura 10.39: **Figura 39: Borrar producto**

### 10.1.10 Gestión (Tareas)

En los **Tareas** tenemos tres secciones (**Inicializar tarea**, **Modificar tarea** y **Borrar tarea**), como se muestra en la siguiente *Figura 40: Tareas*:

The screenshot shows the "Tareas" section of the Tibisay application. It features a yellow square icon with a white clock face. To its right, the word "Tareas" is displayed in large, bold, black font. Below this, there are three green buttons with white text: "Inicializar tarea", "Modificar tarea", and "Borrar tarea".

Figura 10.40: **Figura 40: Tareas**

### Inicializar tarea

En esta opción tenemos los campos (**Flujo\***), (**Producto\***), (**Tarea\***), (**Responsable**), (**Porcentaje tarea**) y (**Descripción tarea**) como se muestra en la siguiente *Figura 41: Inicializar tarea*:

The screenshot shows the Tibisay - Gestión web application. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a 20 years of the revolution logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. The user is identified as "Usuario de Administrador [admin/Administrador]". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small circular icon. Below this, the page title "Inicializar tarea" is displayed. The form consists of several input fields: "Flujo \*", "Producto \*", "Tarea \*", "Responsable", "Porcentaje tarea", and "Descripción tarea". At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Inicializar tarea

Flujo \*

Producto \*

Tarea \*

Responsable

Porcentaje tarea

Descripción tarea

Enviar Limpiar

Figura 10.41: Figura 41: Inicializar tarea

### Modificar tarea

En esta opción tenemos los campos (**Id\***), (**Responsable**), (**Porcentaje tarea**) y (**Descripcion tarea**), como se muestra en la siguiente *Figura 42: Modificar tarea*:

The screenshot shows the Tibisay - Gestión application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Youth Ministry logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. On the right side of the header, it says "Usuario de Administrador [admin/Administrador]". The main title "Tibisay - Gestión" is displayed above the form. The form itself is titled "Modificar tarea". It contains four input fields: "Id \*", "Responsable", "Porcentaje tarea", and "Descripcion tarea". Below the "Descripcion tarea" field is a large text area with a resize handle. At the bottom of the form are two buttons: "Enviar" and "Limpiar".

Figura 10.42: **Figura 42: Modificar tarea**

### Borrar tarea

En esta opción tenemos los campos (**Número de actividad\***), como se muestra en la siguiente *Figura 43: Borrar tarea*:

The screenshot shows the Tibisay - Gestión web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the National Center for Development and Research in Free Technologies (Cenditel), and the Juventud 20 logo. Below the header, there is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Usuario de Administrador [admin/Administrador]". The main content area has a dark blue header with the text "Tibisay - Gestión" and a small icon. Below this, there is a section titled "Borrar tarea" with a form. The form includes a text input field labeled "Número de actividad \*", two buttons "Enviar" and "Limpiar", and a dropdown menu containing a list of 13 items related to project activities. The dropdown menu is open, showing the following options:

- (7 )Planificación
- (8 )Conceptualización
- (9 )Planificación
- (10 )Elaboración\_de\_manuales
- (11 )Definir\_la\_arquitectura\_de\_la\_aplicación
- (12 )Validación
- (13 )Publicación

Figura 10.43: Figura 43: Borrar tarea

#### 10.1.11 Gestión (Avance de actividades)

En los **Tareas** tenemos tres secciones (**Siguiente estado proyectos**), como se muestra en la siguiente *Figura 44: Avance de actividades*:

The screenshot shows the "Avance de actividades" section of the Tibisay - Gestión application. It features a title "Avance de actividades" with a chart icon. Below the title, there are three navigation items: "Actividades de los proyectos" (with a factory icon), "Siguiente estado proyectos" (with a green checkmark icon), and another "Siguiente estado proyectos" item. The second "Siguiente estado proyectos" item is highlighted in green.

Figura 10.44: Figura 44: Avance de actividades

#### Siguiente estado proyectos

En esta opción tenemos los campos (**Numero de actividad\***), (**Siguiente Estado\***), (**Porcentaje\***) y (**Enlace para el producto**) como se muestra en la siguiente *Figura 45: Siguiente estado*:

The screenshot shows a web application interface for 'Tibisay - Gestión'. At the top, there is a header bar with the Venezuelan Government logo, the text 'Gobierno Bolivariano de Venezuela', 'Ministerio del Poder Popular para Ciencia, Tecnología e Innovación', 'Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel)', and the 'Juventud' logo. Below the header, a navigation menu includes 'INICIO', 'GESTIÓN', 'CONSULTAS', 'CRÉDITOS', and 'CERRAR SESIÓN'. To the right of the menu, it says 'Usuario de Administrador [admin/Administrador]'. The main content area has a dark blue header with the text 'Tibisay - Gestión' and a circular icon. Below this, there is a form titled 'Siguiente estado proyectos'. The form contains four input fields: 'Número de actividad \*' with a dropdown menu, 'Siguiente Estado \*' with a dropdown menu, 'Porcentaje \*' with a text input field, and 'Enlace para el producto' with a text input field. At the bottom of the form are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.45: **Figura 45: Siguiente estado**

### 10.1.12 Gestión (Administración)

En los **Tareas** tenemos tres secciones (**Modificar registro de actividad**), como se muestra en la siguiente *Figura 46: Administración*:



Figura 10.46: **Figura 46: Administración**

#### Modificar registro de actividad

En esta opción tenemos los campos (**Numero registro\***), (**Propietario**), (**Porcentaje**) y (**Fecha de la acción**) como se muestra en la siguiente *Figura 47: Modificar registro*:

The screenshot shows a form titled 'Tibisay - Gestión' under the heading 'Modificar registro de actividad'. The form contains four input fields: 'Número registro \*' (with a dropdown arrow), 'Propietario', 'Porcentaje', and 'Fecha de la acción'. At the bottom are two buttons: 'Enviar' and 'Limpiar'.

Figura 10.47: **Figura 47: Modificar registro**

### 10.1.13 Consulta (Reportes)

En los **Reportes** tenemos cinco secciones (**Listar datos**, **Listar datos consolidado**, **Listar datos general**, **Listar datos con autofiltro** y **Listar datos con filtrorecursivo**), como se muestra en la siguiente *Figura 48: Reportes*:

The screenshot shows the Tibisay - Consultas application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Youth logo. Below the header is a navigation bar with links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar is the text "Usuario de Administrador [admin/Administrador]". The main content area has a green header with the title "Tibisay - Consultas" and a sub-header "Reportes". Under "Reportes", there is a list of five options: "Listar datos", "Listar datos consolidado", "Listar datos general", "Listar datos con autofiltro", and "Listar datos con filtrorecursivo".

Figura 10.48: **Figura 48: Reportes**

#### Listar datos

En esta opción tenemos los campos (**Cargar archivo flujo\***) y (**Variable\***) como se muestra en la siguiente *Figura 49: Listar datos*:

#### Listar datos consolidado

En esta opción tenemos los campos (**Cargar archivo flujo\***) y (**Variable\***) como se muestra en la siguiente *Figura 50: Listar datos consolidado*:

#### Listar datos general

En esta opción tenemos los campos (**Cargar archivo flujo\***) y (**Variable\***) como se muestra en la siguiente *Figura 51: Listar datos general*:

The screenshot shows the Tibisay - Consultas application. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and the Juventud 200 logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session message indicates "Usuario de Administrador [admin/Administrador]". The main content area has a green header titled "Tibisay - Consultas" with a small icon of a person holding a torch. Below this, the text "Listar datos" is displayed. The form contains fields for "Cargar archivo flujo \*" (with a dropdown menu), "Variable \*" (with a dropdown menu), and two buttons: "Enviar" and "Limpiar". A "Parámetros" button is also present.

Figura 10.49: Figura 49: Listar datos

The screenshot shows the Tibisay - Consultas application. The layout is identical to Figure 49, featuring the same header, navigation bar, and user session message. The main content area has a green header titled "Tibisay - Consultas" with a small icon of a person holding a torch. Below this, the text "Listar datos consolidado" is displayed. The form contains fields for "Cargar archivo flujo \*" (with a dropdown menu), "Variable \*" (with a dropdown menu), and two buttons: "Enviar" and "Limpiar". A "Parámetros" button is also present.

Figura 10.50: Figura 50: Listar datos consolidado

## Documentación de PySafet, Publicación

The screenshot shows the Tibisay - Consultas application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo, and a Juventud logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session message indicates the user is 'Usuario de Administrador [admin/Administrador]'. The main content area has a green header titled 'Tibisay - Consultas'. Below this, a section titled 'Listar datos general' contains two dropdown menus: 'Cargar archivo flujo \*' and 'Variable \*'. At the bottom of this section are 'Enviar' and 'Limpiar' buttons, and a 'Parámetros' link.

Figura 10.51: **Figura 51: Listar datos general**

### Listar datos con autofiltro

En esta opción tenemos los campos (**Cargar archivo flujo\***), (**Autofiltro**) y (**Variable\***) como se muestra en la siguiente *Figura 52: Listar datos con autofiltro*:

### Listar datos con filtrorecursivo

En esta opción tenemos los campos (**Cargar archivo flujo\***), (**Filtro recursivo\***) y (**Variable\***) como se muestra en la siguiente *Figura 53: Listar datos con filtrorecursivo*:

#### 10.1.14 Consulta (Gráficos de proyectos)

En los **Gráficos de proyectos** tenemos dos secciones (**Generar gráfico coloreado** y **Generar gráfico para clave Proyecto**), como se muestra en la siguiente *Figura 54: Gráficos de proyectos*:

The screenshot shows a web application interface for 'Tibisay - Consultas'. At the top, there are navigation links: INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of these is a user session message: 'Usuario de Administrador [admin/Administrador]'. The main title 'Tibisay - Consultas' is displayed above a search form. The search form includes fields for 'Cargar archivo flujo \*' (with a dropdown arrow), 'Autofiltro' (with a dropdown arrow), 'Variable \*' (with a dropdown arrow), and two buttons: 'Enviar' and 'Limpiar'. Below the search form is a blue rectangular button labeled 'Parámetros'.

Figura 10.52: Figura 52: Listar datos con autofiltro

### Generar gráfico coloreado

En esta opción tenemos los campos (**Cargar archivo flujo\***) como se muestra en la siguiente *Figura 55: Gráfico coloreado*:

### Generar gráfico para clave Proyecto

En esta opción tenemos los campos (**Id\***) y (**Cargar archivo flujo\***) como se muestra en la siguiente *Figura 56: Gráfico por clave*:

#### 10.1.15 Consulta (Gestión de gráficos)

En los **Gestión de gráficos** tenemos nueve secciones (**Restaurar gráfico de flujo**, **Borrar gráfico de flujo**, **Agregar nodo a gráfico de flujo**, **Borrar nodo de gráfico de flujo**, **Cambiar conexión de gráfico de flujo**, **Cambiar rol de tarea**, **Cambiar nota de nodo**, **Cambiar información del nodo** y **Indicadores de procesos**), como se muestra en la siguiente *Figura 57: Gestión de gráficos*:

The screenshot shows the Tibisay - Consultas interface. At the top, there are logos for the Government of Venezuela, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel). The menu bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. A user session indicator shows "Usuario de Administrador [admin/Administrador]". Below the menu, a green header bar displays the title "Tibisay - Consultas" next to a small icon. The main content area is titled "Listar datos con filtrorecursivo". It contains three input fields: "Cargar archivo flujo \*", "Filtro recursivo \*", and "Variable \*". Below these fields are two buttons: "Enviar" and "Limpiar". A "Parámetros" button is located at the bottom left of the form.

Figura 10.53: **Figura 53: Listar datos con filtrorecursivo**

## Gráficos de proyectos

Generar gráfico coloreado  
Generar gráfico para clave Proyecto

Figura 10.54: **Figura 54: Gráficos de proyectos**

The screenshot shows a web application interface. At the top, there is a header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. Below the header, a navigation bar includes links for INICIO, GESTIÓN, CONSULTAS, CRÉDITOS, and CERRAR SESIÓN. To the right of the navigation bar, it says "Usuario de Administrador [admin/Administrador]". The main title "Tibisay - Consultas" is displayed in a green header bar. Below the title, there is a section titled "Generar gráfico coloreado". A form field labeled "Cargar archivo flujo \*" has a dropdown menu open, listing various options: Reporte de indicadores, Reporte de las acciones Específicas, Software\_libre, Software\_educativo, Dispositivos, Metodologías, Publicación, and Publicación\_arbitrada. There are also "Enviar" and "Limpiar" buttons. A "Parámetros" button is located at the bottom left of the form.

Figura 10.55: **Figura 55: Gráfico coloreado**

The screenshot shows a similar web application interface to Figura 55. It features the same header with the Government of Venezuela logo, the Ministry of Popular Power for Science, Technology, and Innovation, and the Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (Cenditel) logo. The navigation bar and user information are identical. The main title "Tibisay - Consultas" is in a green header bar. Below the title, there is a section titled "Generar gráfico para clave Proyecto". A form field labeled "Id \*" contains a dropdown menu. Below it is another form field labeled "Cargar archivo flujo \*". At the bottom of the form are "Enviar" and "Limpiar" buttons, and a "Parámetros" button.

Figura 10.56: **Figura 56: Gráfico por clave**

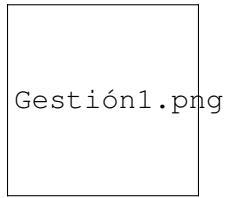


Figura 10.57: **Figura 57: Gestión de gráficos**

### Restaurar gráfico de flujo

En esta opción tenemos los campos (**Nombre grafo\***) como se muestra en la siguiente *Figura 58: Restaurar (gráfico de flujo)*:

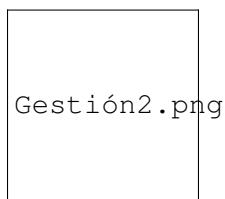


Figura 10.58: **Figura 58: Restaurar (gráfico de flujo)**

### Borrar gráfico de flujo

En esta opción tenemos los campos (**Nombre grafo\***) como se muestra en la siguiente *Figura 59: Borrar (gráfico de flujo)*:

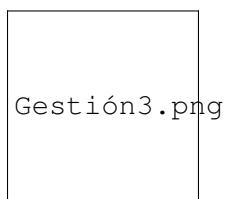


Figura 10.59: **Figura 59: Borrar (gráfico de flujo)**

### Agregar nodo a gráfico de flujo

En esta opción tenemos los campos (**Nombre grafo\*),(Nodo anterior\*),(Nombre nuevo nodo\*),(Es paralelo al nodo anterior),(Campo options),(Campo query),(Campo title)** y (**Campo documentsource**) como se muestra en la siguiente *Figura 60: Agregar nodo al (gráfico de flujo)*:

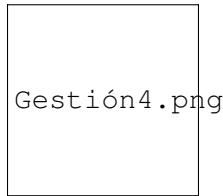


Figura 10.60: **Figura 60: Agregar nodo al (gráfico de flujo)**

### Borrar nodo de gráfico de flujo

En esta opción tenemos los campos (**Nombre grafo\***) y (**Nodo a borrar\***) como se muestra en la siguiente *Figura 61: Borrar nodo de (gráfico de flujo)*:

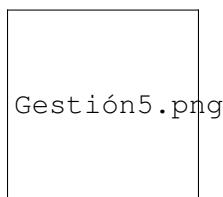


Figura 10.61: **Figura 61: Borrar nodo de (gráfico de flujo)**

### Cambiar conexión de gráfico de flujo

En esta opción tenemos los campos (**Nombre grafo\***), (**Nodo origen\***), (**Nodo destino actual\***) y (**Nuevo nodo destino\***) como se muestra en la siguiente *Figura 62: Cambiar conexión de (gráfico de flujo)*:

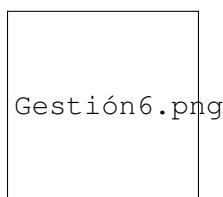


Figura 10.62: **Figura 62: Cambiar conexión de (gráfico de flujo)**

### Cambiar rol de tarea

En esta opción tenemos los campos (**Nombre grafo\***), (**Tarea a cambiar\***) y (**Nombre del rol\***) como se muestra en la siguiente *Figura 63: Cambiar rol de tarea del (gráfico de flujo)*:

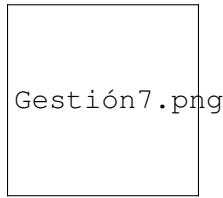


Figura 10.63: **Figura 63: Cambiar rol de tarea del (gráfico de flujo)**

### Cambiar nota de nodo

En esta opción tenemos los campos (**Nombre grafo\***), (**Tarea a cambiar\***) y (**Nota**) como se muestra en la siguiente *Figura 64: Cambiar nota de nodo del (gráfico de flujo)*:

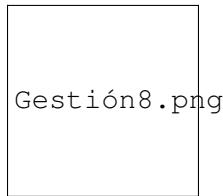


Figura 10.64: **Figura 64: Cambiar nota de nodo del (gráfico de flujo)**

### Cambiar información del nodo

En esta opción tenemos los campos (**Nombre grafo\***), (**Tarea a cambiar\***) y (**Informacion del nodo\***) como se muestra en la siguiente *Figura 65: Cambiar información del nodo del (gráfico de flujo)*:

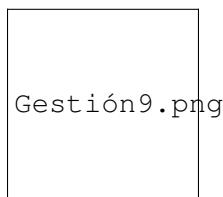


Figura 10.65: **Figura 65: Cambiar información del nodo del (gráfico de flujo)**

### Indicadores de procesos

En esta opción tenemos los campos (**Flujo\***), (**Tarea a cambiar\***) y (**Informacion del nodo\***) como se muestra en la siguiente *Figura 66: Indicadores de procesos del (gráfico de flujo)*:

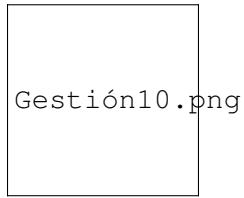


Figura 10.66: **Figura 66: Indicadores de procesos del (gráfico de flujo)**

### 10.1.16 Consulta (Planificación)

En la **Planificación** tenemos seis secciones (**Agregar planificación**, **Cambiar fecha de planificación**, **Copiar fechas de planificación**, **Borrar planificación**, **Copiar planificación** y **Comparar gráfico de flujo**), como se muestra en la siguiente *Figura 67: Planificación*:

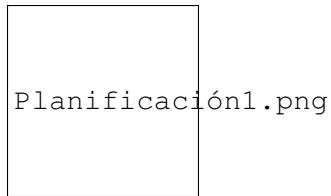


Figura 10.67: **Figura 67: Planificación**

### Agregar planificación

En esta opción tenemos los campos (**Cargar archivo flujo\***), (**Id\***) y (**Nombre grafo\***) como se muestra en la siguiente *Figura 68: Agregar (planificación)*:

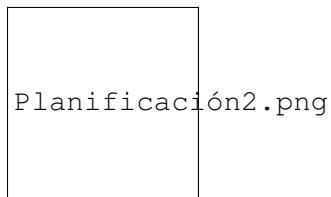


Figura 10.68: **Figura 68: Agregar (planificación)**

### Cambiar fecha de planificación

En esta opción tenemos los campos (**Nombre grafo\***), (**Tarea\***) y (**Fecha planificada\***) como se muestra en la siguiente *Figura 69: Cambiar fecha de (planificación)*:

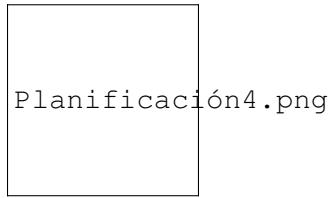


Figura 10.69: **Figura 69: Cambiar fecha de (planificación)**

### Copiar fechas de planificación

En esta opción tenemos los campos (**Nombre grafo origen\***) y (**Nombre grafo destino\***) como se muestra en la siguiente *Figura 70: Copiar fechas de (planificación)*:

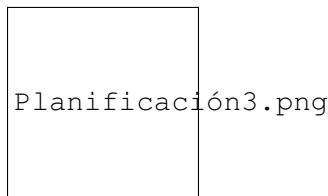


Figura 10.70: **Figura 70: Copiar fechas de (planificación)**

### Borrar planificación

En esta opción tenemos los campos (**Nombre grafo\***) como se muestra en la siguiente *Figura 71: Borrar (planificación)*:

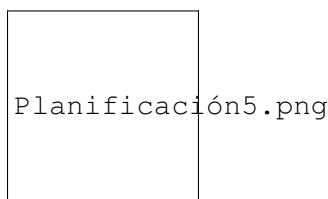


Figura 10.71: **Figura 71: Borrar (planificación)**

### Copiar planificación

En esta opción tenemos los campos (**Nombre planificacion\***) y (**Nombre grafo\***) como se muestra en la siguiente *Figura 72: Copiar (planificación)*:

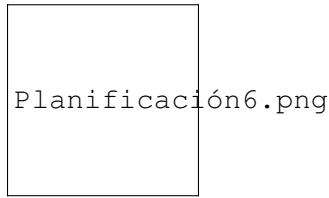


Figura 10.72: **Figura 72: Copiar (planificación)**

### Comparar gráfico de flujo

En esta opción tenemos los campos (**Nombre grafo\***) y (**Planificado\***) como se muestra en la siguiente *Figura 73: Comparar gráfico de flujo*:

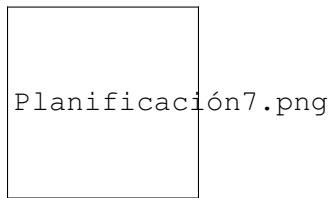


Figura 10.73: **Figura 73: Comparar gráfico de flujo**

### 10.1.17 Consulta (Edición de flujos)

En los **Edición de flujos** tenemos tres secciones (**Agregar grafo**, **Editar grafo** y **Borrar grafo**), como se muestra en la siguiente *Figura 74: Edición de flujos*:

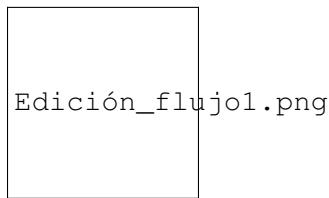


Figura 10.74: **Figura 74: Edición de flujos**

#### Agregar grafo

En esta opción tenemos los campos (**Nombre grafo\***) como se muestra en la siguiente *Figura 75: Agregar (grafo)*:

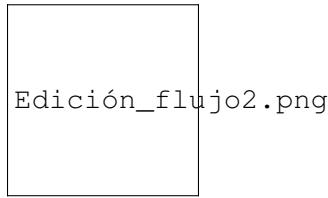


Figura 10.75: **Figura 75: Agregar (grafo)**

### Editar grafo

En esta opción tenemos los campos (**Cargar archivo flujo\***) como se muestra en la siguiente *Figura 76: Editar (grafo)*:

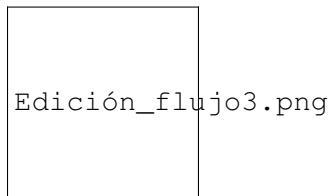


Figura 10.76: **Figura 76: Editar (grafo)**

### Borrar grafo

En esta opción tenemos los campos (**Nombre flujo\***) como se muestra en la siguiente *Figura 77: Borrar (grafo)*:

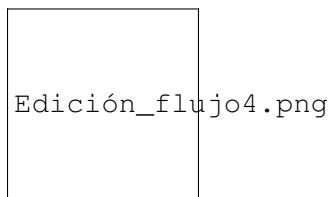


Figura 10.77: **Figura 77: Borrar (grafo)**

## 10.2 2.- Instalación

### 10.2.1 Herramienta Editflow para modela flujos de trabajo

El **Editorflow** fue hecho con **Wireit** la podemos encontrar en la dirección <http://neyric.github.io/wireit/docs/>

## A.- Descargamos el Editflow

### 1° PRIMER PASO

- Descargamos el archivo (**Editflow.tar.gz**), dando un clik al link que se muestra en la siguiente imagen:



Figura 10.78: Editflow.tar.gz

### 2° SEGUNDO PASO

- Descomprimimos el archivo **tar.gz** en nuestro <HOME>.

## B.- Visualización de Editflow en el servidor web

### 1° PRIMER PASO

- Abrimos nuestro navegador **Web** y buscamos en el menú la opción (**Abrir archivo**), como se muestra en la siguiente *Figura 78: Navegador Web*:

### 2° SEGUNDO PASO

- Seleccionamos el (**index.html**) de nuestro carpeta (**editFlow/trunk/**), como se muestra en la siguiente *Figura 79: Seleccionar archivo principal*:

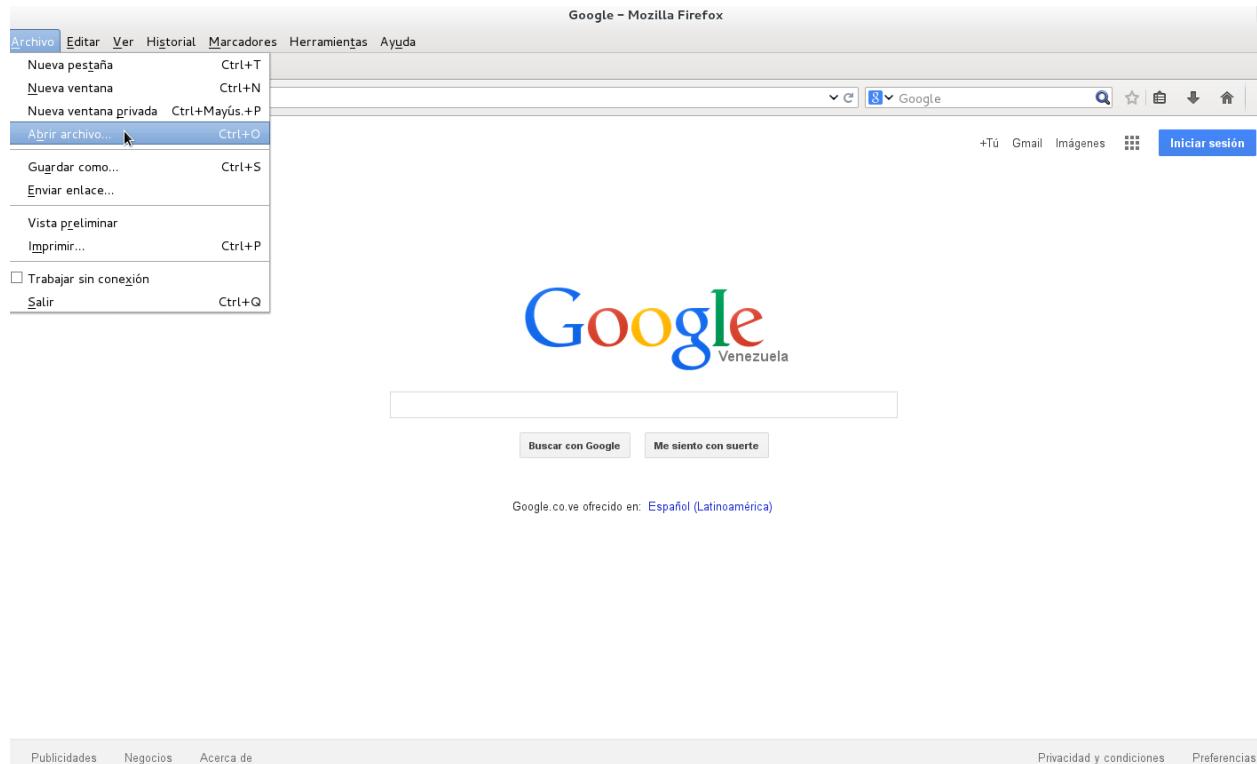


Figura 10.79: Figura 78: Navegador Web

### 3° TERCER PASO

- Se nos mostrara la pagina principal del (**EditFlow**) que contiene una **área** de trabajo, un **menú** de opciones, una **barra** de herramienta y propiedades. Observen la siguiente *Figura 80: Editflow*:

#### 10.2.2 PySafet y Web-Safet

##### A.- Instalacion de PySafet

Para realizar la instalación nos vamos al siguiente enlace: Instalación de PySafet.

##### B.- Instalación del servirdor Apache

Para realizar la instalación nos vamos al siguiente enlace: *A.- Instalación de servidor apache y su modulo python* .

##### C.- Instalación del Framework Web-Safet

Para realizar la instalación nos vamos al siguiente enlace: *B.- Instalación de WebSafet* .

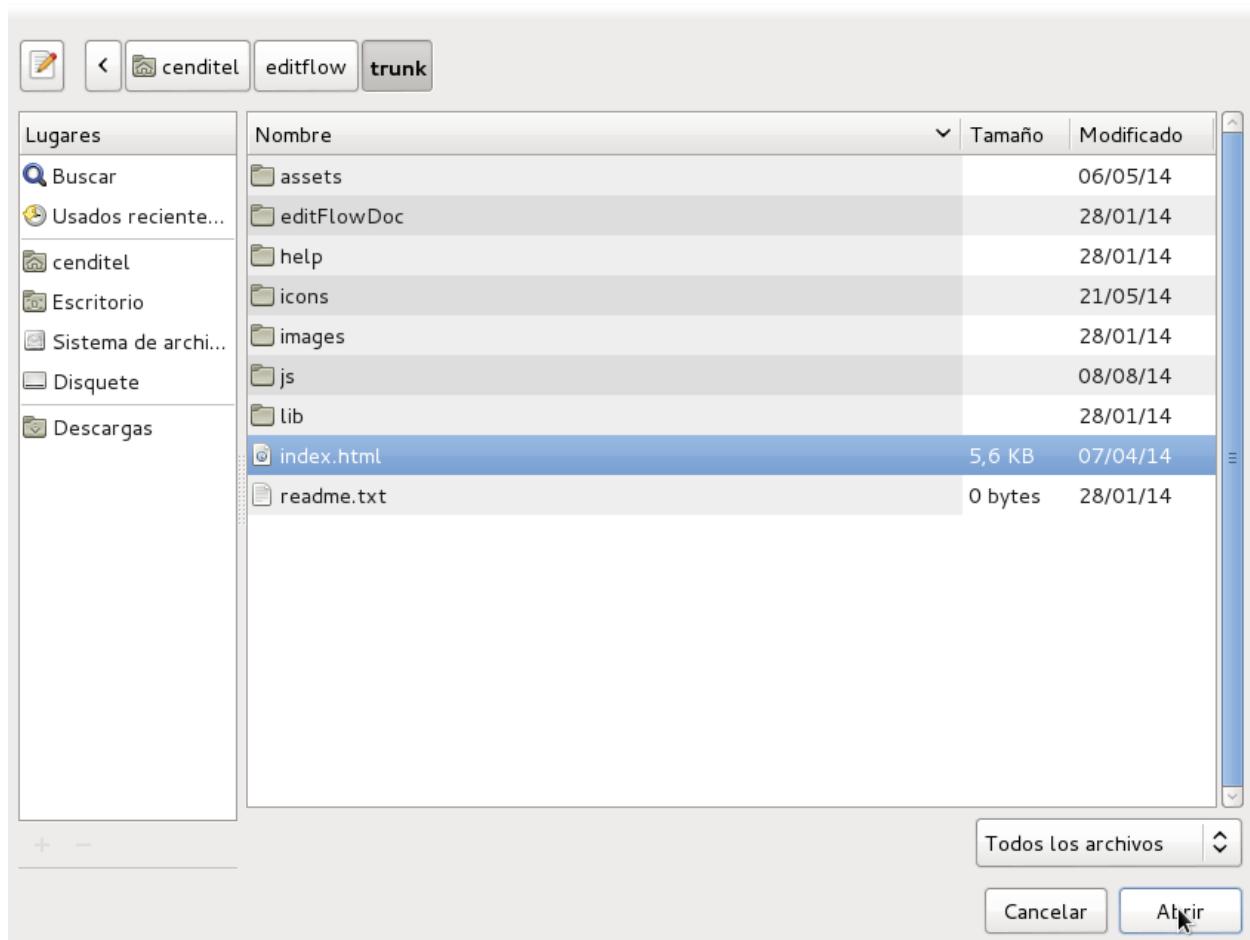


Figura 10.80: Figura 79: Seleccionar archivo principal

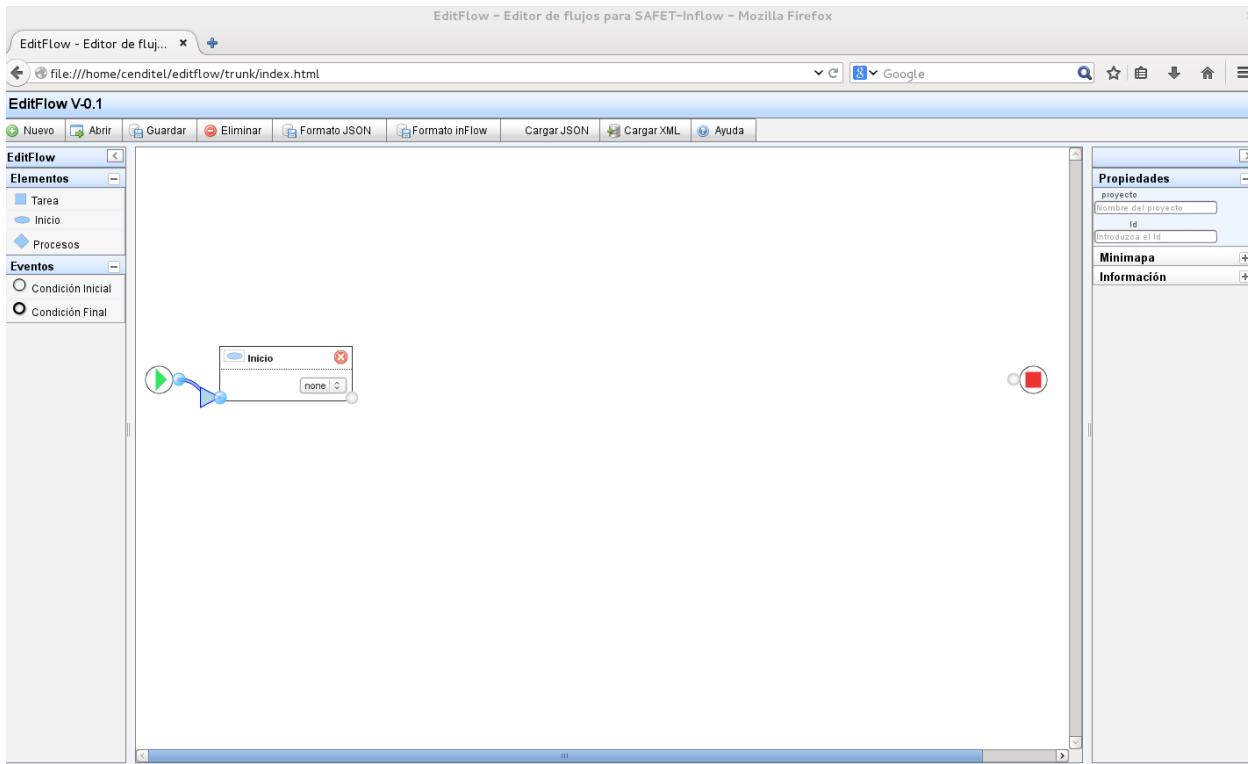


Figura 10.81: **Figura 80: Editflow**

### 10.3 3.- Configuración

#### 10.3.1 Servidor Apache

Para realizar la configuración del servidor Apache nos vamos al siguiente enlace: [C.- Configuración de servidor apache2](#).

#### 10.3.2 Web-Safet

Para realizar la configuración del framework Web-Safet nos vamos al siguiente enlace: [D.- Configuración de WebSafet en los directorio \(intranet y media\)](#).

#### 10.3.3 Cargar el proyecto gestión en el directorio .safet/

##### 1° PRIMER PASO

- Descargamos el archivo (**Proyecto\_de\_Gestion.tar**), dando un clik al link que se muestra en la siguiente imagen:



Figura 10.82: Proyecto de Gestión.tar

## 2° SEGUNDO PASO

Para cargar el proyecto gestón se debe realizar como en el siguiente enlace: Cargar el proyecto gestión mediante este ejemplo .

## 10.4 4.- Secciones del Editorflow

### 10.4.1 Interfaz Gráfica EditFlow

#### A.- Barra de Menú del EditFlow



Figura 10.83: Figura 81: Menú

**1.- Botón (Nuevo):** permite limpiar el área de diagrama para crear un nuevo flujo, los únicos elementos que no son removidos son las condiciones inicial y final.

**2.- Botón (Abrir):** permite mostrar el listado de flujos que han sido guardados anteriormente en el computador para poder seleccionar uno y cargarlo en la interfaz.

**3.- Botón (Guardar):** permite almacenar el flujo que se encuentre en el área de diagrama en el computador y posteriormente poder cargarlo. Para almacenar un flujo los campos de ciertas propiedades deben estar llenos (Id, Key y KeySource). Si alguno de ellos no lo está la aplicación nos alertará con un mensaje.

**4.- Botón (Eliminar):** esta opción permite eliminar el flujo que se encuentre en ese momento cargado en la interfaz. Después de esta acción no se podrá recuperar la información de este flujo.

**5.- Botón (Formato JSON):** este botón permite convertir el flujo que tengamos dibujado en el área de diagrama en un archivo en formato JSON (este formato permite conservar la ubicación de los elementos en el área de diagrama).

**6.- Botón (Formato InFlow):** este botón nos da la opción de convertir el flujo que tengamos dibujado en el área de diagrama en un archivo en formato XML que puede ser utilizado como entrada de la aplicación inFlow para su análisis.

**7.- Botón (Cargar JSON):** con esta opción podremos cargar flujos que tengamos en formato JSON, que previamente hayan sido generados con editFlow.

**8.- Botón (Cargar XML):** permite cargar al igual que en la opción anterior flujos en XML en formato inFlow, con la diferencia de que este formato no conserva la ubicación de los elementos del flujo en el área de diagrama.

**9.- Botón (Ayuda):** permite ingresar a este documento de ayuda. **Observen la Figura 81: Menú**

## B.- Barra de herramientas del Editflow

### Elementos

1. **Tarea:** Este elemento permite describir el contenido de una tarea.



**Figura 82: Tarea**

2. **Inicio:** Este elemento permite describir el inicio del flujo de trabajo.



Figura 83: Inicio

## 3. Procesos:



Figura 84: Procesos

## Eventos

1. **Evento Inicial:** el evento inicial representa el punto de inicio de los flujos de trabajo, es cargado por defecto en el área de diagrama y no es posible eliminarlo ya que todo flujo debe tener uno.



Figura 85: Evento inicial.

2. **Evento Final:** el evento final representa el punto final de los flujos de trabajo, es cargado por defecto en el área de diagrama y no es posible eliminarlo ya que todo flujo debe tener uno.



Figura 86: Evento Final.

3. **Enlaces:** los enlaces son una especie de cables que permiten conectar a través de los terminales los distintos elementos que se encuentren en el área de diagrama. Observen la *Figura 87: Enlace*

4. **Conexión:** Este elemento permite interconectar tareas entre ellas y con las condiciones inicial y final.

5. **Variable:** Este elemento permite asociar información de una variable a una tarea, cada tarea debe tener asociado como mínimo una variable. El terminal de conexión de la variable se debe conectar

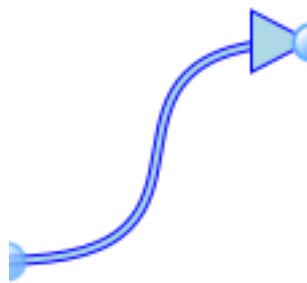


Figura 10.84: Figura 87: Enlace

con el terminal que se encuentra en la parte superior de la tarea.

6. **Autofiltro** Este elemento permite asociar información de los autofiltros de una tarea. El terminal superior de conexión del autofiltro se debe conectar con el terminal que se encuentra en la parte superior de la tarea, y el terminal inferior del autofiltro se debe conectar de la misma forma con la tarea “Source” del autofiltro.

### C.- Barra de herramientas de Propiedades

**1.- Propiedades:** Las propiedades de un flujo se encuentran en el panel derecho de la interfaz. Contiene los siguientes campos:

- **Proyecto:** en este campo se debe introducir el **nombre** del flujo para a guardarlo. flujo al guardar.
- **Id:** en este campo se debe introducir el identificador del flujo. **Observen la siguiente Figura 105: Proyecto**

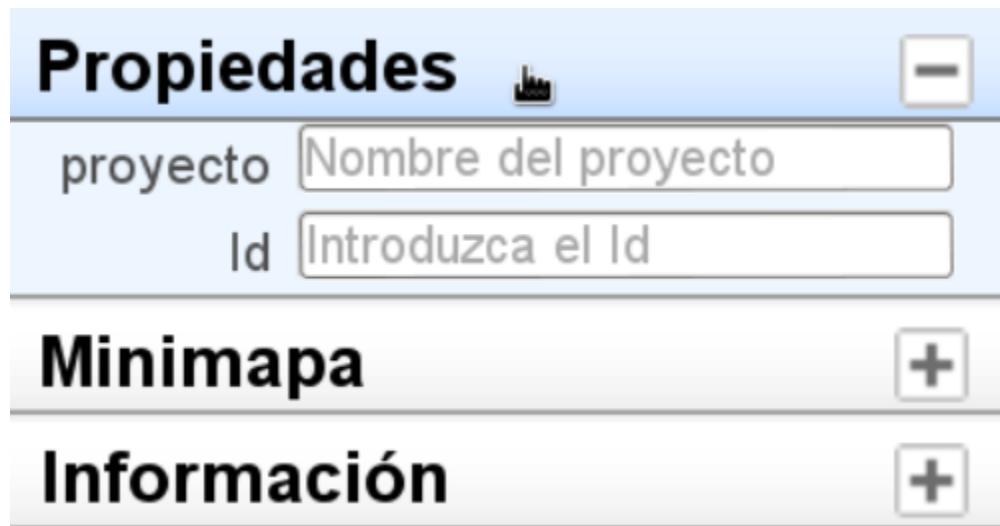


Figura 10.85: Figura 88: Proyecto

**2.- Minimapá:** Aquí se nos muestra un minimapa del área de trabajo. **Observen la siguiente Figura 89: Mapa**



Figura 10.86: **Figura 89: Mapa**

**3.- Información:** Aquí una pequeñas información de los que hace el **EditFlow**. **Observen la siguiente Figura 90: Información**

#### Ejemplo de un Diagrama hecho en el EditFlow

Esta área esta compuesta por una especie de lienzo de color blanco en donde pueden ser colocados los distintos elementos que componen un flujo y conectarse entre ellos hasta armar la configuración requerida.

**Observen el diagrama de un inventario Figura 91: Diagrama**

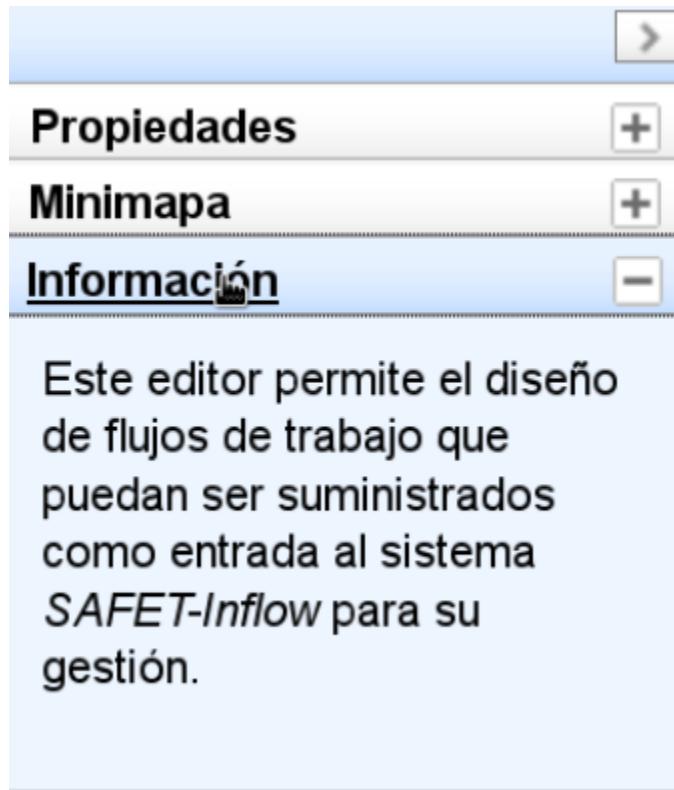


Figura 10.87: Figura 90: Información

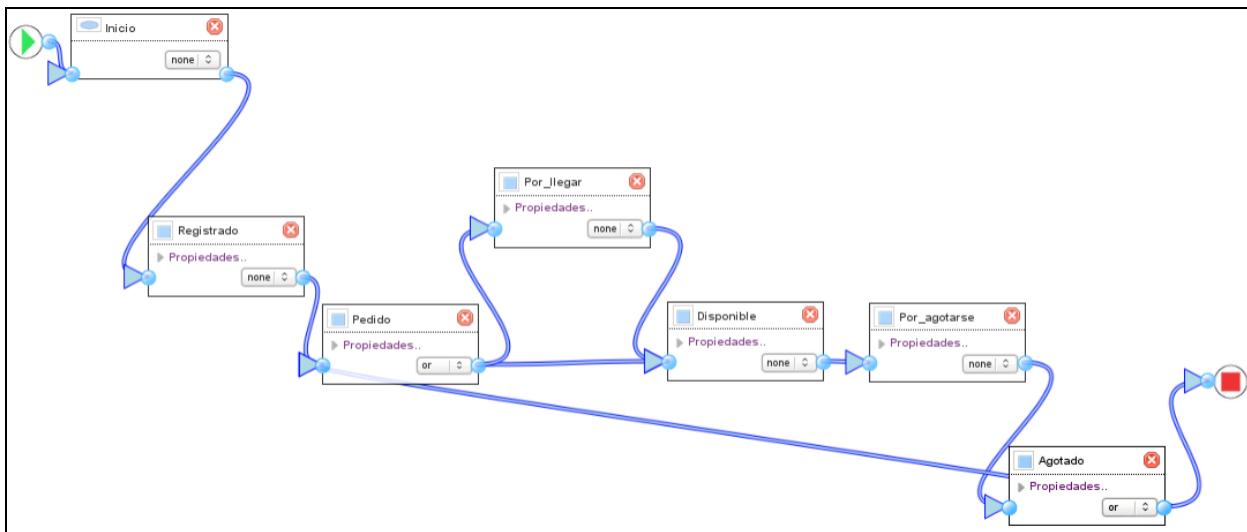


Figura 10.88: Figura 91: Diagrama

## 10.5 5.- Ejemplo paso a paso de la herramienta EditFlow

### 10.5.1 Creación de un modelos de flujo de trabajo

#### A.- Abrimos la herramienta EditFlow

##### 1° PRIMER PASO

- Abrimos nuestro navegador **Web** y buscamos en el menú la opción (**Abrir archivo**), como se muestra en la siguiente *Figura 92: Navegador Web*:

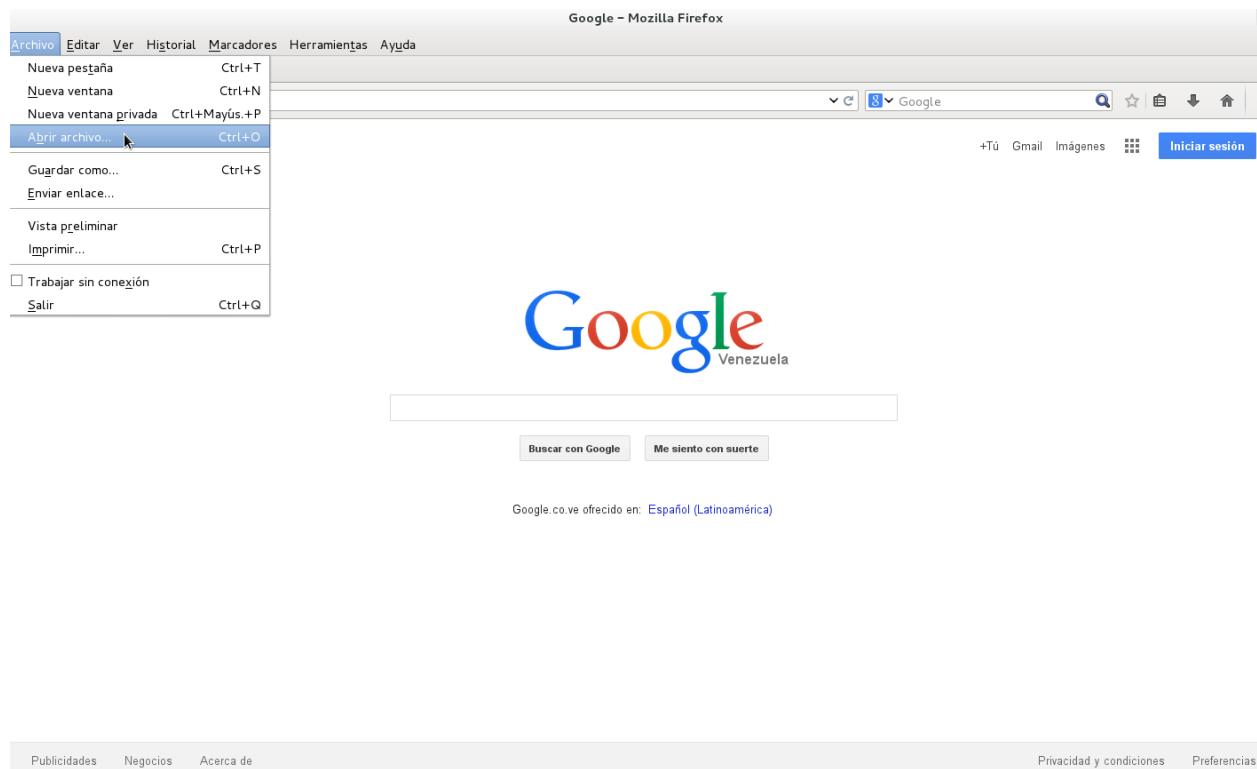


Figura 10.89: Figura 92: Navegador Web

##### 2° SEGUNDO PASO

- Seleccionamos el (**index.html**) de nuestro carpeta (**editFlow/trunk/**), como se muestra en la siguiente *Figura 93: Seleccionar archivo principal*:

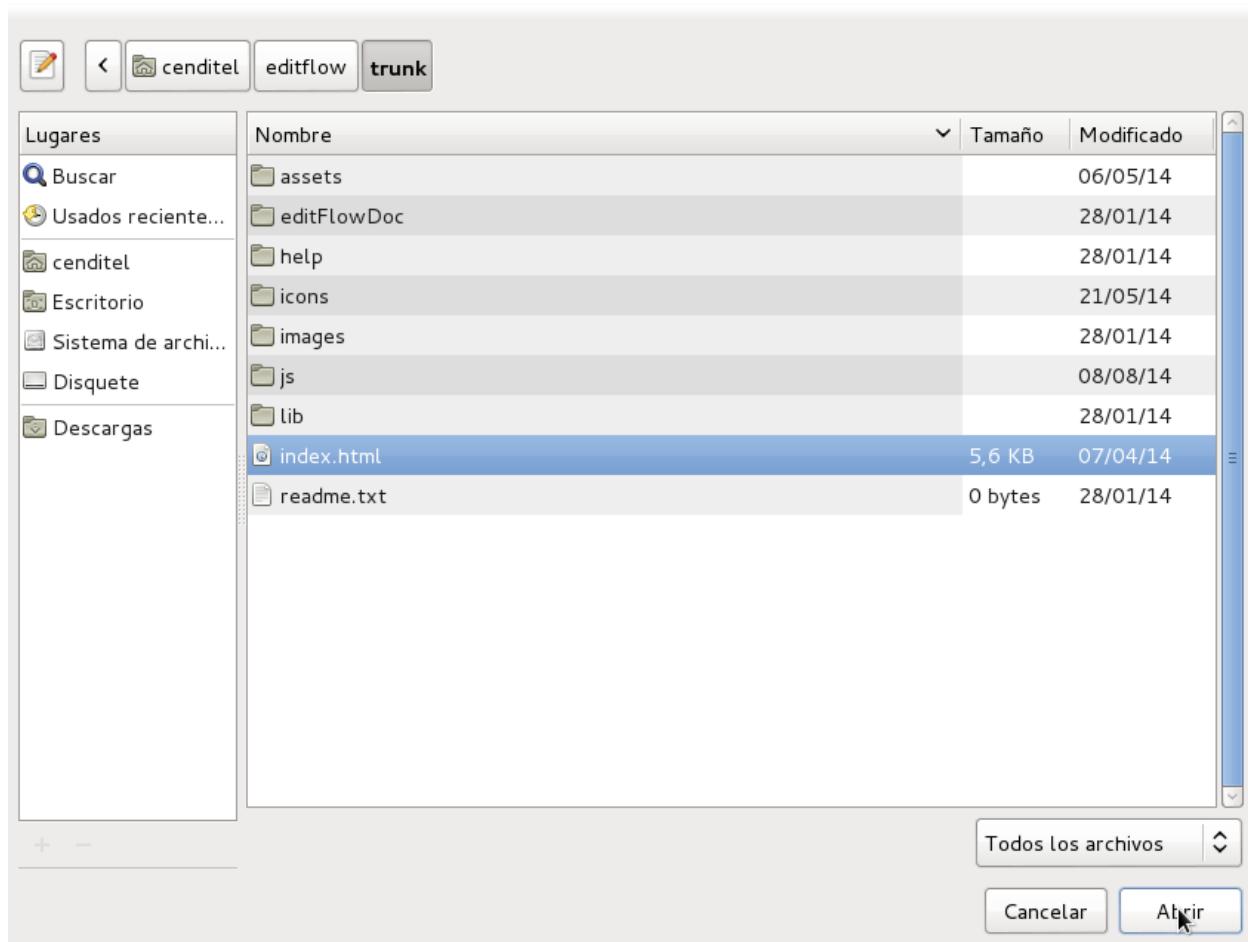


Figura 10.90: Figura 93: Seleccionar archivo principal

### 3° TERCER PASO

- Se nos mostrara la pagina principal del (**EditFlow**) que contiene una **área** de trabajo, un **menú** de opciones, una **barra** de herramientas y propiedades. Observen la siguiente *Figura 94: Editflow*:

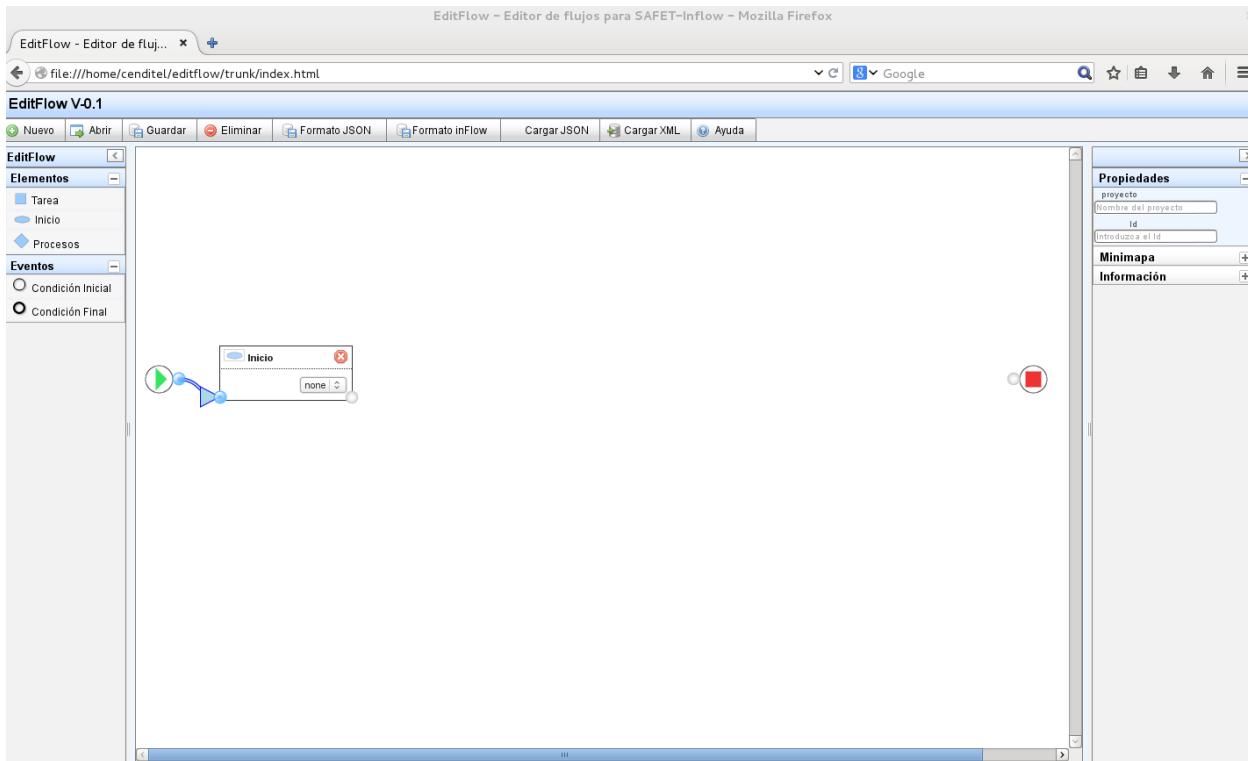


Figura 10.91: **Figura 94: Editflow**

### B.- Creación de un modelo de publicación

En este ejemplo vamos a modelar el gráfico de Publicación. Observen la siguiente *Figura 95: Publicación*:

A continuación seguimos los siguientes paso:

### 1° PRIMER PASO

- Seleccionamos la tarea, como se muestre la siguiente *Figura 96: Tarea*:

### 2° SEGUNDO PASO

- Arrastramos la tarea seleccionada al block de trabajo, como se muestre la siguiente *Figura 97: Block de trabajo*:

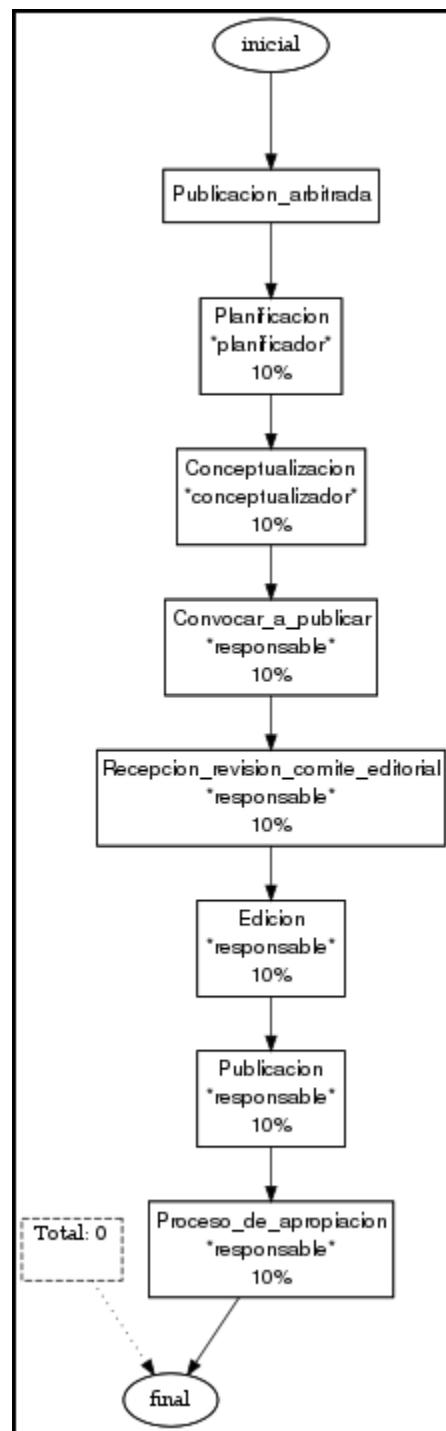


Figura 10.92: Figura 95: Publicación

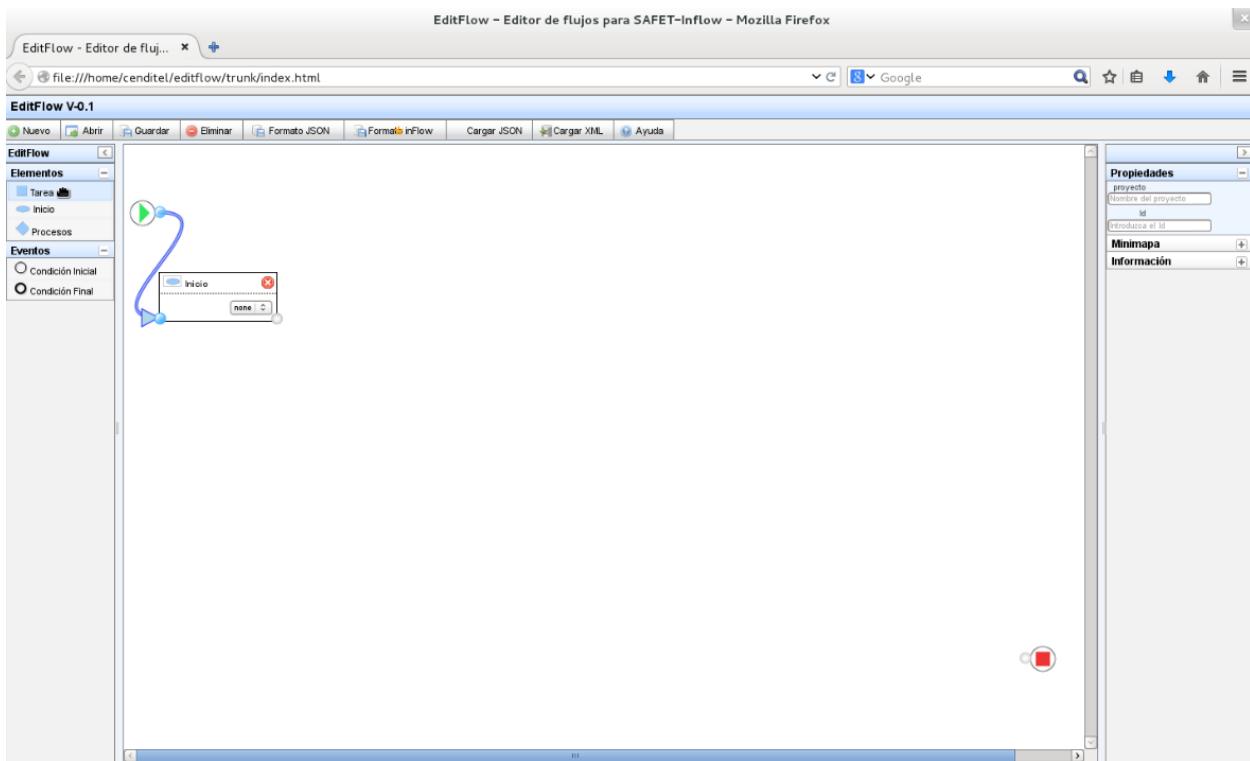


Figura 10.93: Figura 96: Tarea

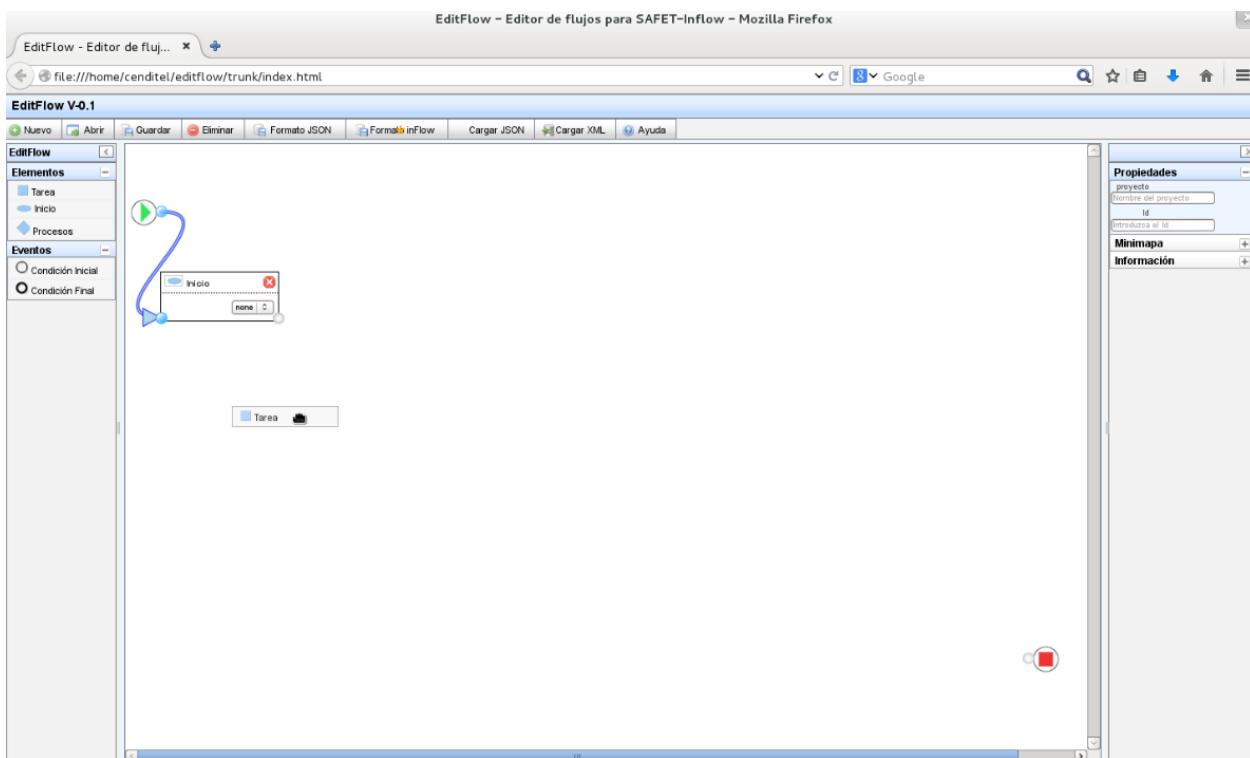


Figura 10.94: Figura 97: Block de trabajo

### 3° TERCER PASO

- Damos clik a las propiedades de la tarea, como se muestre la siguiente *Figura 98: Propiedades de la tarea*:

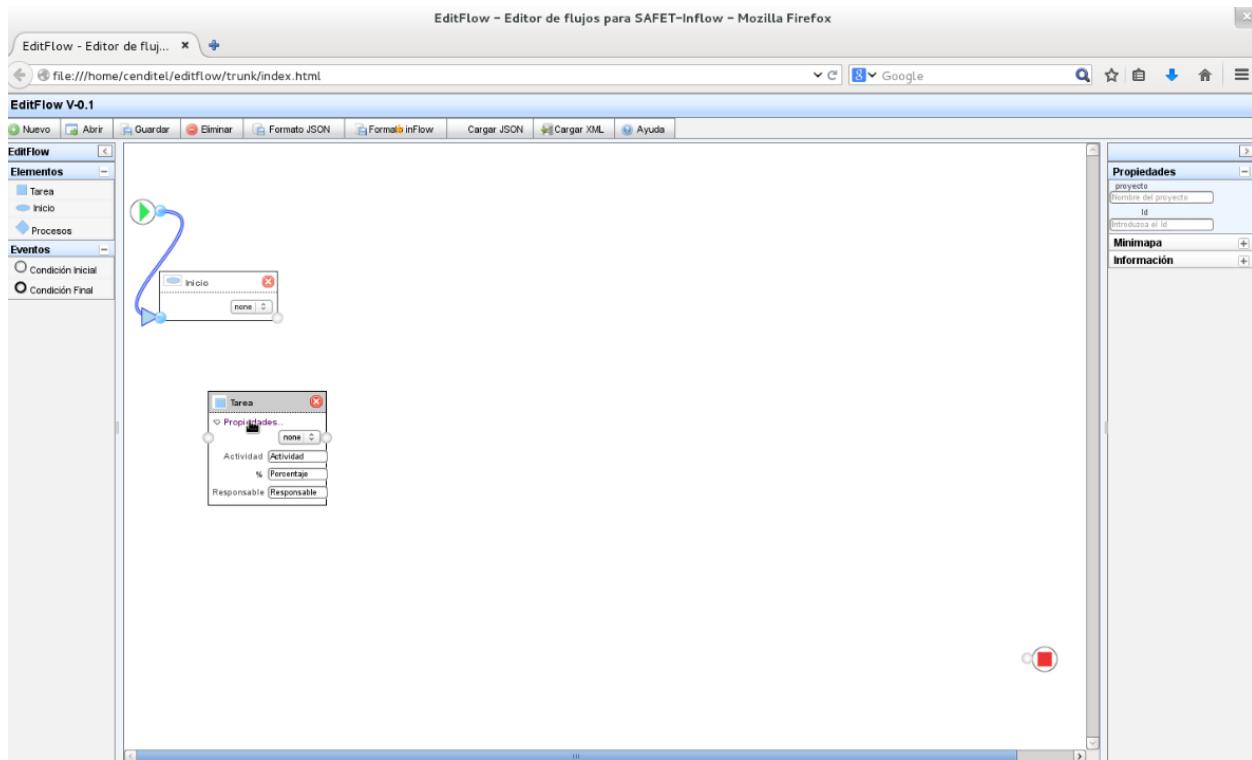


Figura 10.95: *Figura 98: Propiedades de la tarea*

### 4° CUARTO PASO

- Llenamos los campos de la tarea por ejemplo (**Publicacion\_arbitrada,10,administrador**), como se muestre la siguiente *Figura 99: Campos de la tarea*:

### 5° CUARTO PASO

- Cerramos las propiedades de la tarea, como se muestre la siguiente *Figura 100: Opción propiedades*:

### 6° SEXTO PASO

- Seleccionamos en la condición (**inicial o tarea**) el operador a puntar, por ejemplo el operador (**none**) solo para una tarea y el operador (**or**) para dos tareas o procesos, como se muestre la siguiente *Figura 101: Selección de operador*:

## Documentación de PySafet, Publicación

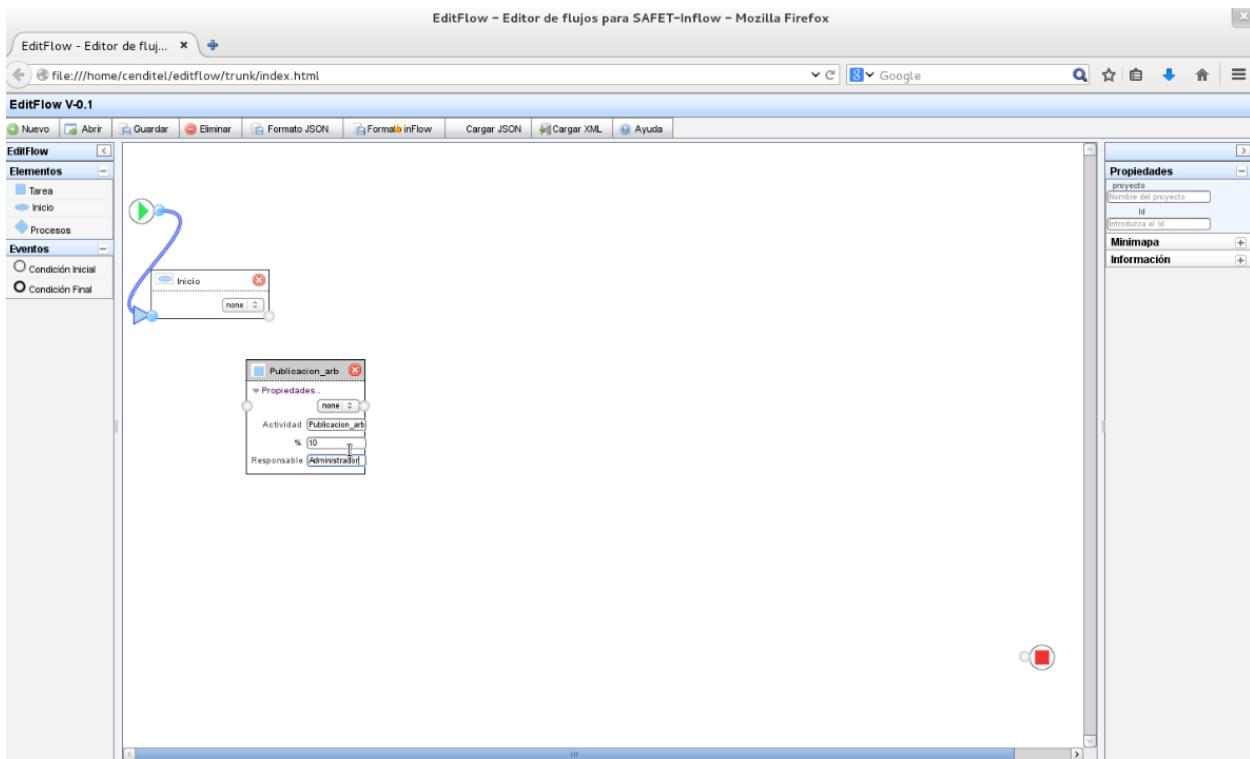


Figura 10.96: Figura 99: Campos de la tarea

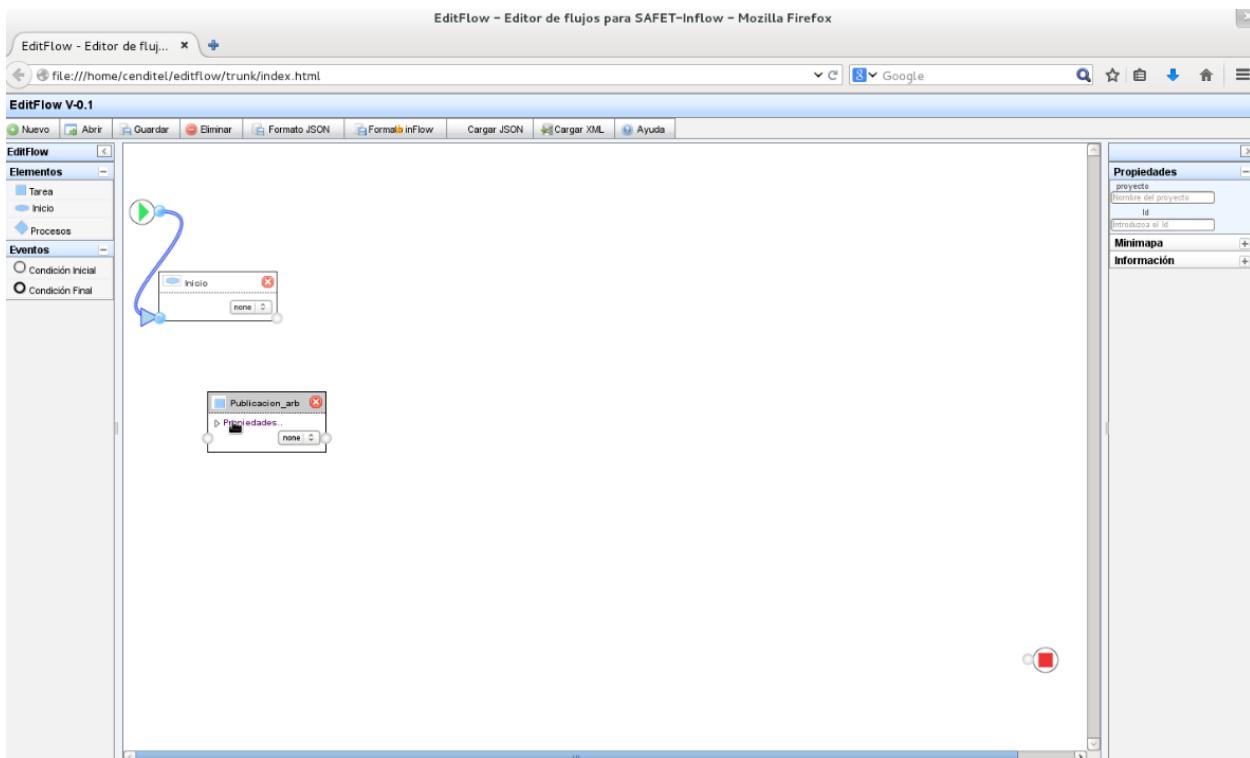


Figura 10.97: Figura 100: Opción propiedades

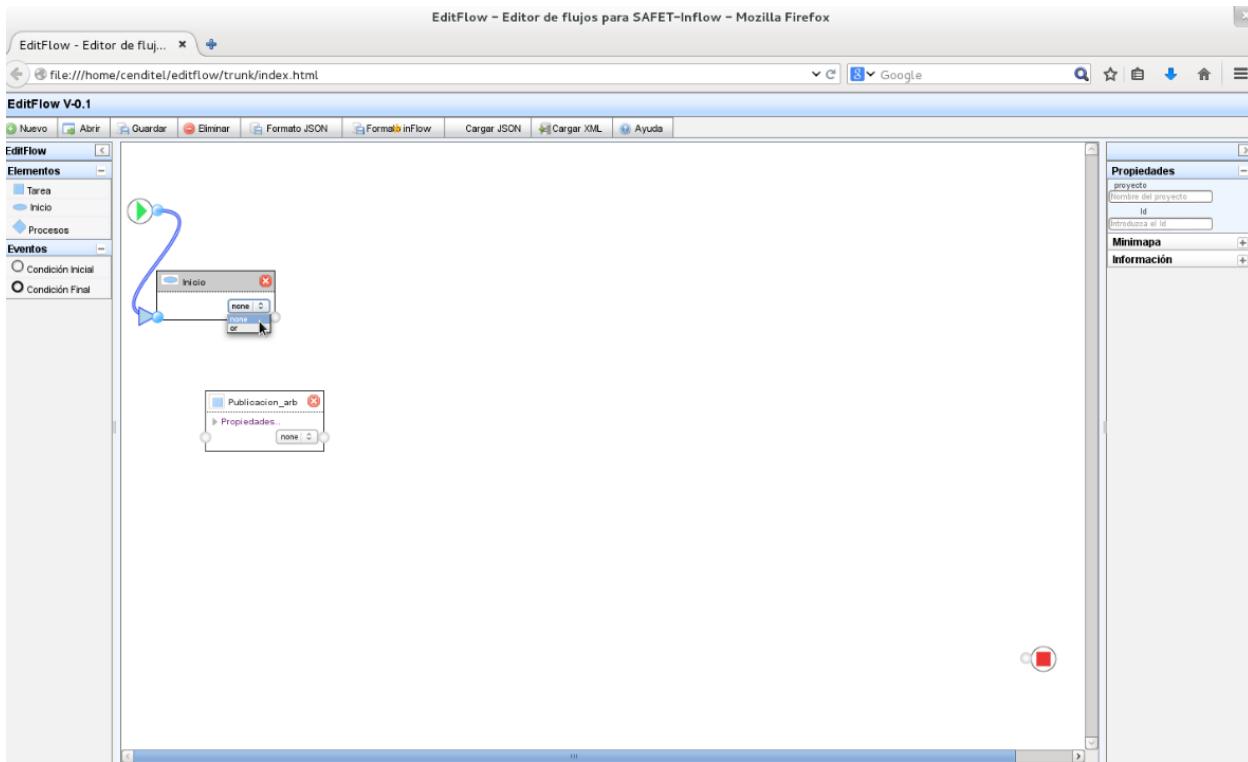


Figura 10.98: **Figura 101: Selección de operador**

## 7° SEPTIMO PASO

- De la condición inicial o tarea llevamos el enlace a la tarea (**Publicacion\_arbitrada**), como se muestre la siguiente *Figura 102: Enlace*:

---

**Nota:** Para realizar las demás tareas se debe hacer los mismos pasos del (1 al 7)

---

## 8° OCTAVO PASO

- Al ya realizado las demás tareas se enlaza con la condición final, como se muestre la siguiente *Figura 103: Enlace condición final*:

---

**Nota:** Así quedara el flujo de trabajos con los enlaces y las cajitas *Figura 104: Diagrama de planificación*:

---

## 9° NOVENO PASO

- Damos nombre al flujo de trabajo por ejemplo (**Planificación**) y su **id**, como se muestre la siguiente *Figura 106: Nombre del flujo*:

## Documentación de PySafet, Publicación

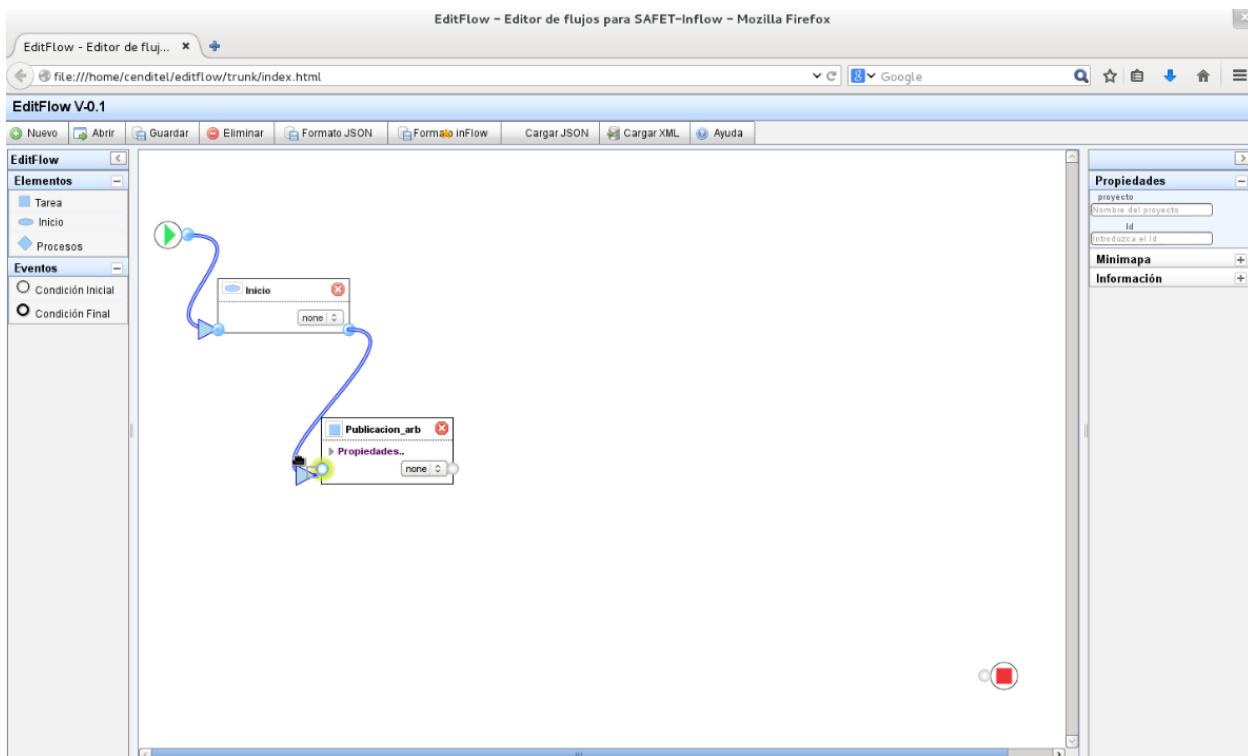


Figura 10.99: Figura 102: Enlace

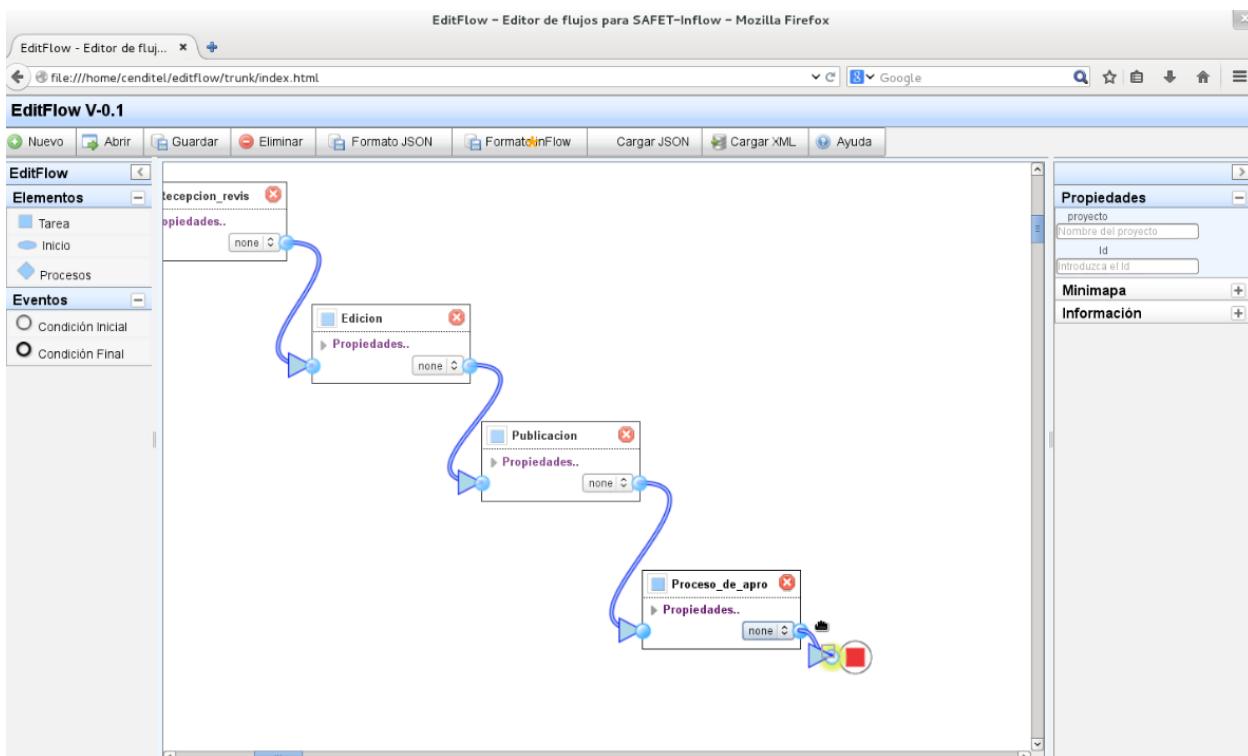


Figura 10.100: Figura 103: Enlace condición final

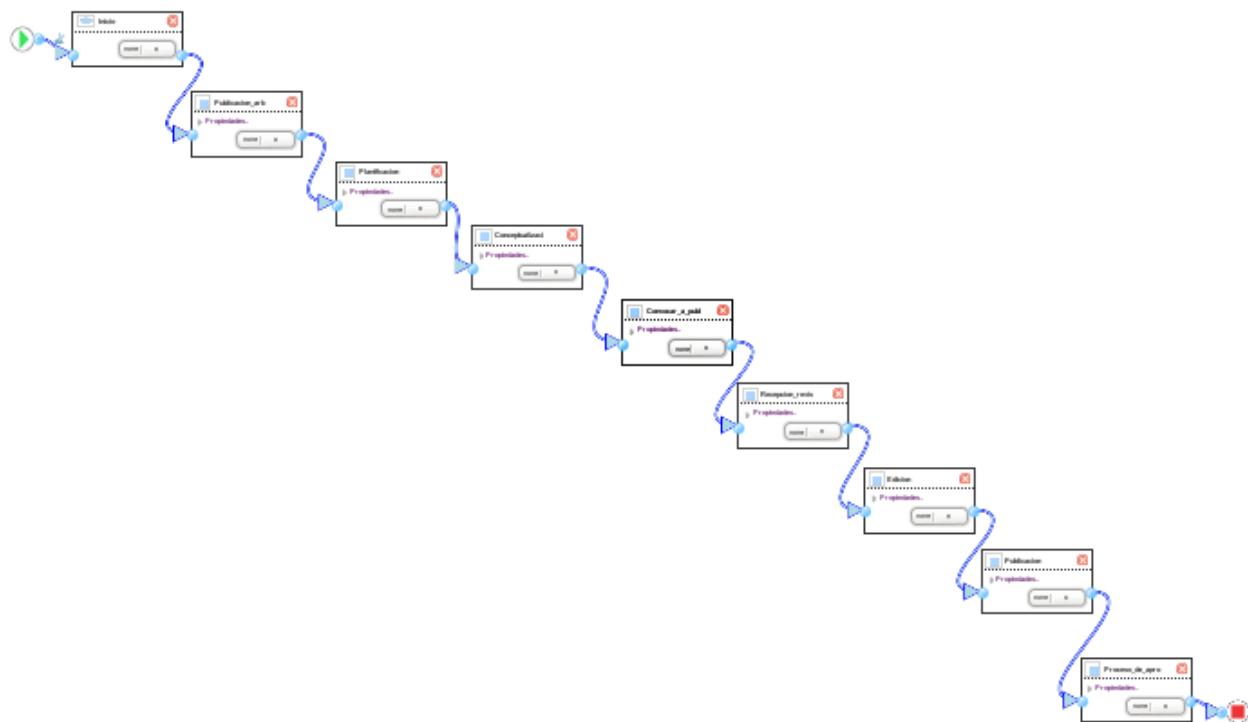


Figura 10.101: Figura 104: Diagrama de planificación

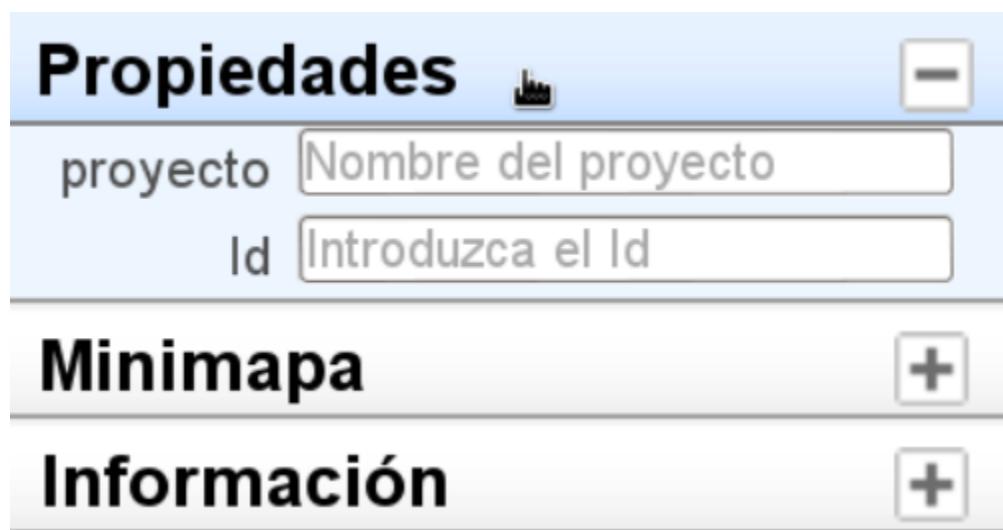


Figura 10.102: Figura 105: Proyecto

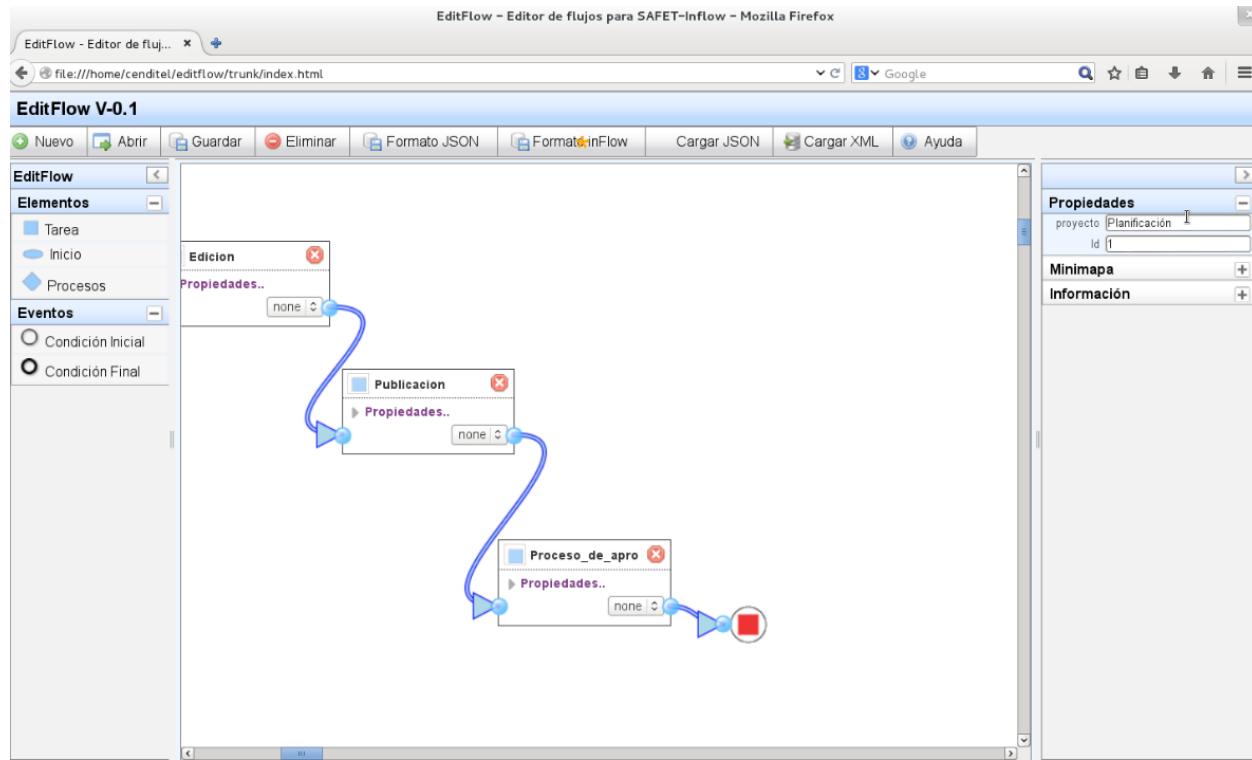


Figura 10.103: **Figura 106: Nombre del flujo**

### C.- Diagrama de flujo en formato JSON

#### 1° PRIMER PASO

- Seleccionamos en el menú la opción (**Formato JSON**), como se muestre la siguiente *Figura 107: Opción*:

**Nota:** Se nos generada un formato (**JSON**) como se muestra en la siguiente: *Figura 108: Formato JSON*:

---

#### 2° SEGUNDO PASO

- Guardamos el formato para ello nos vamos a la barra de herramientas del navegador **Web** y damos click en la opción (**Guardar como**), como se muestre la siguiente *Figura 109: Opción*:

#### 3° TERCER PASO

- Colocamos el nombre del archivo con la extensión (.js), por ejemplo (**Publicación.js**) y pulsamos el botón (**Guardar**), como se muestre la siguiente *Figura 110: Guardar archivo*:

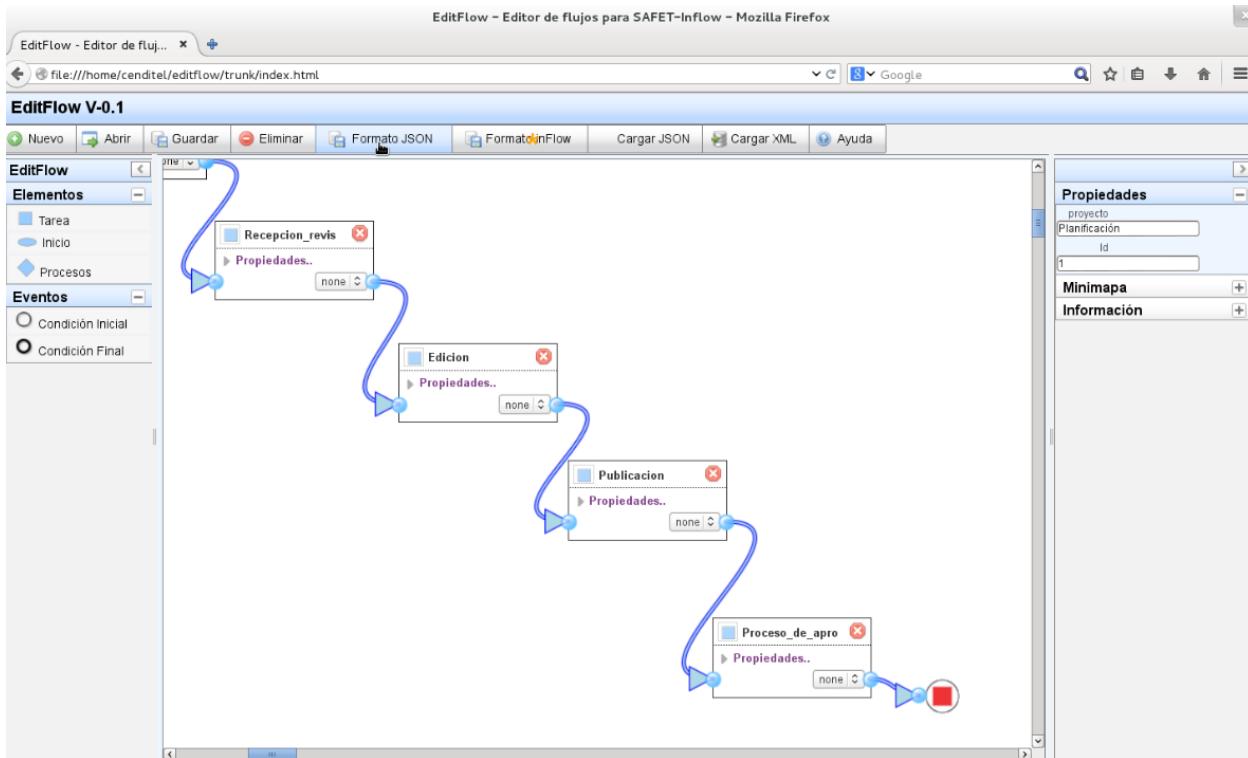


Figura 10.104: Figura 107: Opción

Figura 10.105: Figura 108: Formato JSON

## Documentación de PySafet, Publicación

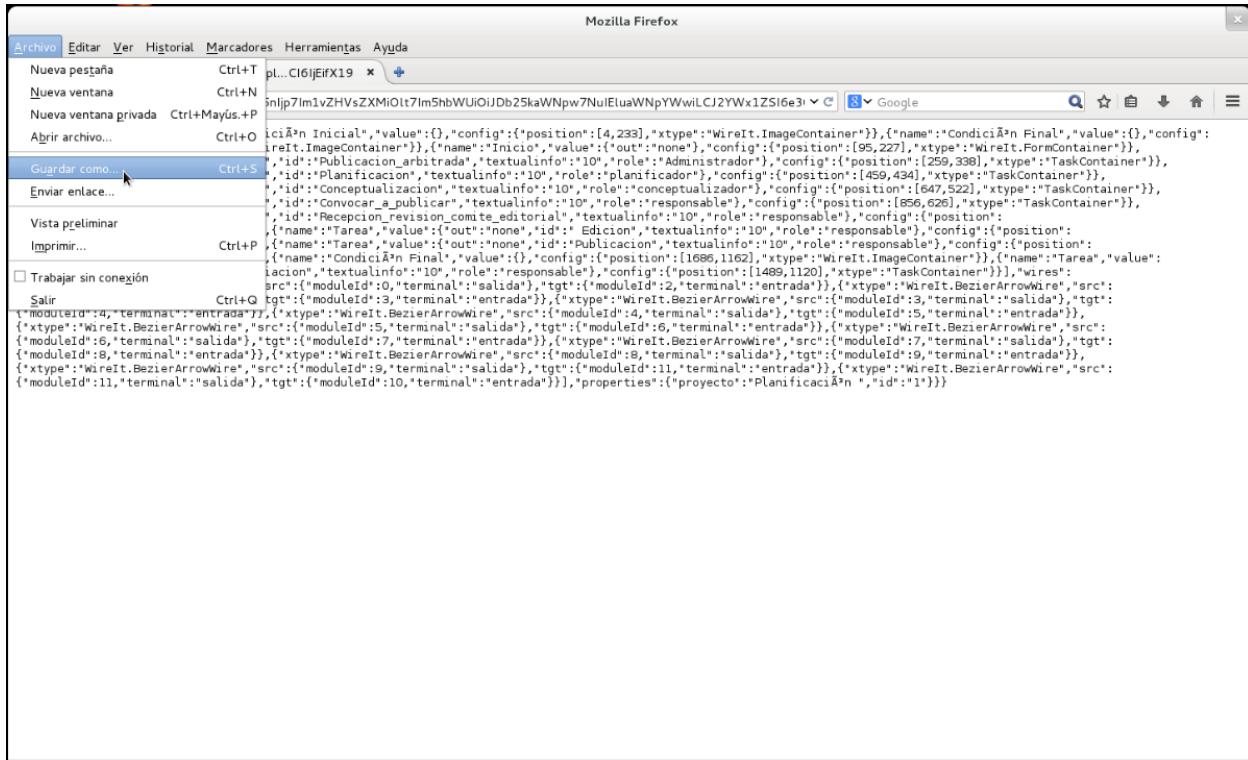


Figura 10.106: Figura 109: Opción

### D.- Diagrama de flujo en formato XML

#### 1° PRIMER PASO

- Seleccionamos en el menú la opción (**Formato inflow**), como se muestre la siguiente *Figura 111: Opción*:

#### 2° SEGUNDO PASO

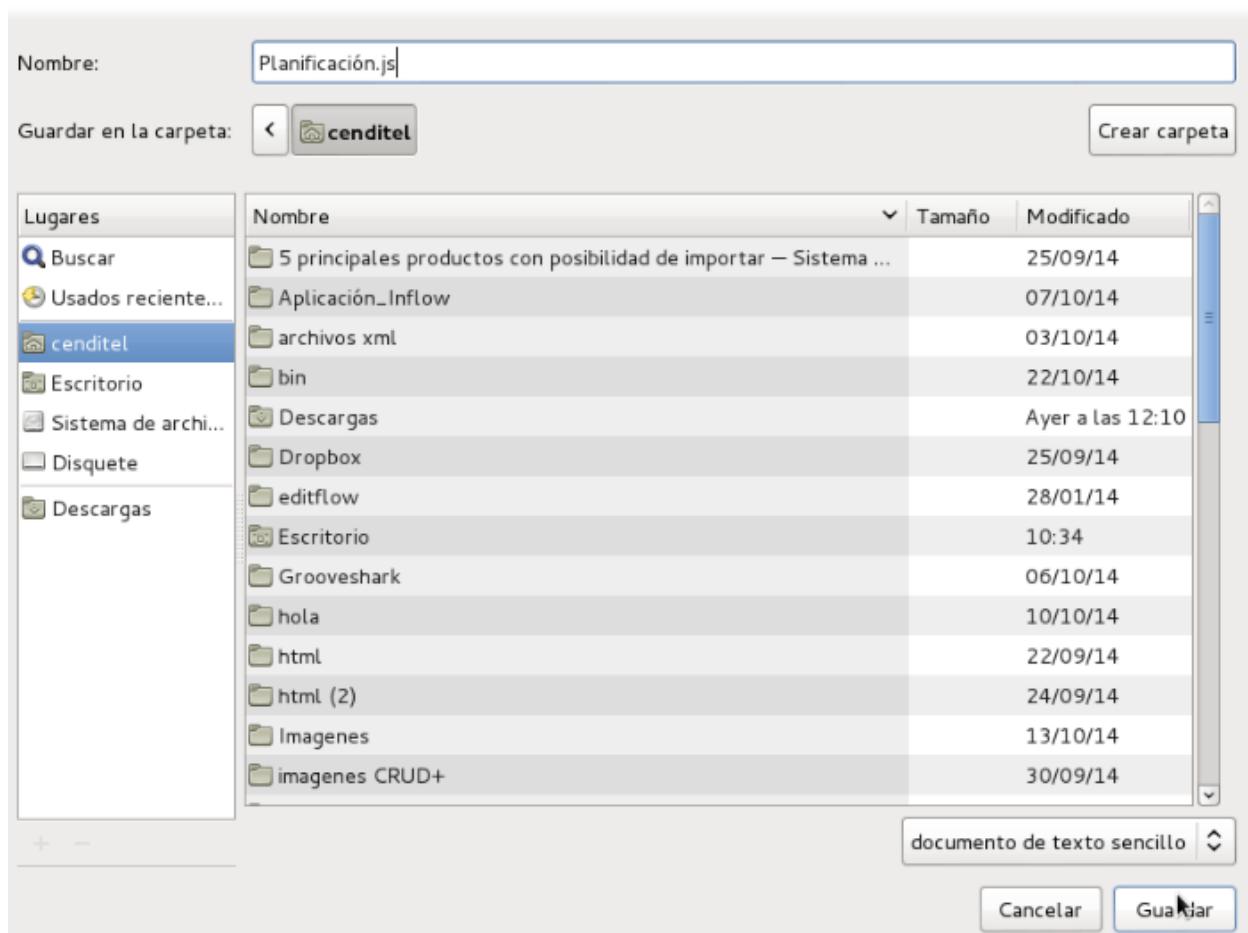
- Se nos mostrara un mensaje del porcentaje del flujo, Pulsamos en el botón (**Aceptar**), como se muestre la siguiente *Figura 112: Mensaje del flujo*:

**Nota:** Se nos generada un formato (XML) como se muestra en la siguiente: *Figura 113: Formato XML*:

■

#### 3° TERCER PASO

- Guardamos el farmota para ello nos vamos a la barra de herramientas del navegador **Web** y damos click en la opción (**Guardar como**), como se muestre la siguiente *Figura 114: Opción*:

Figura 10.107: **Figura 110: Guardar archivo**

## Documentación de PySafet, Publicación

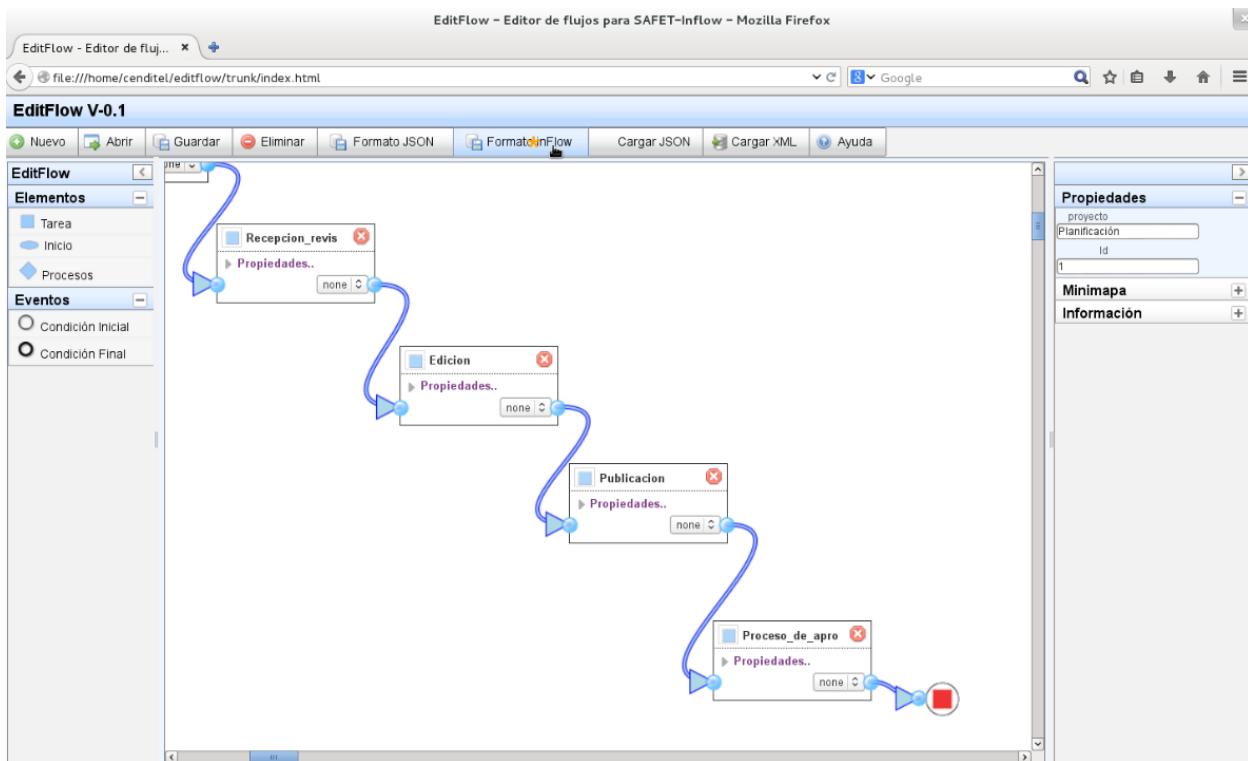


Figura 10.108: Figura 111: Opción

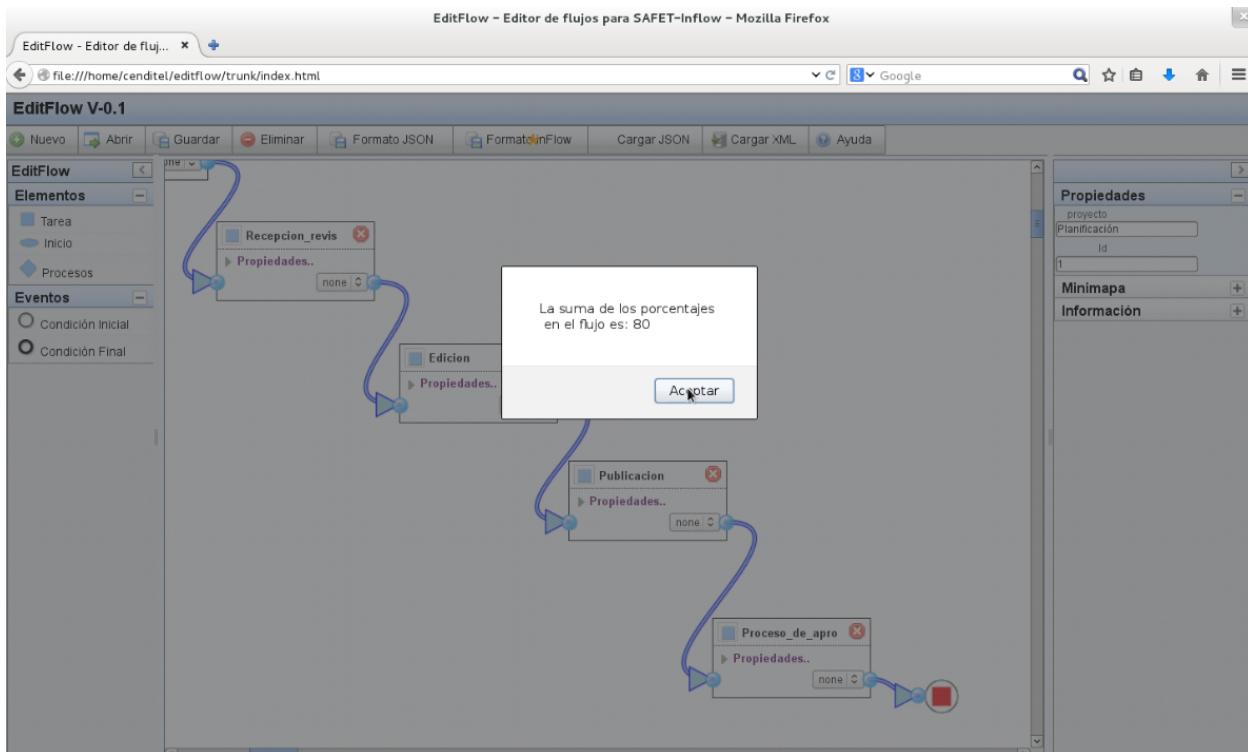
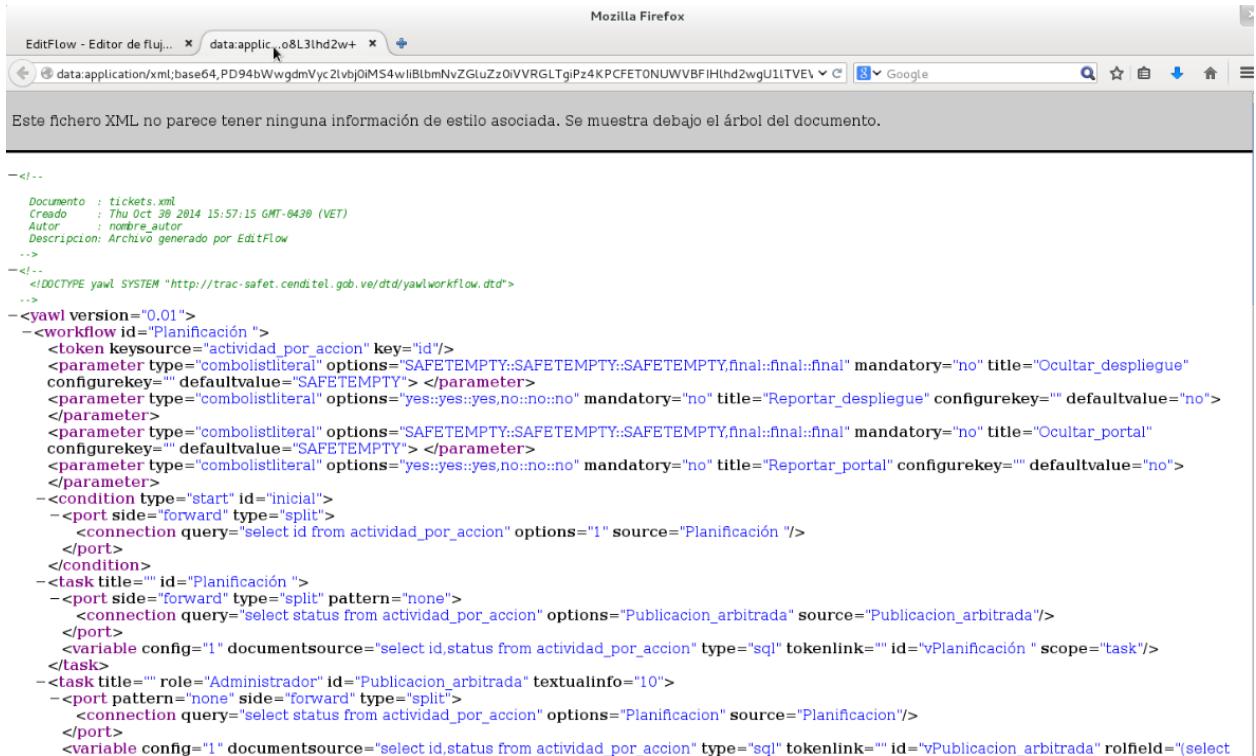


Figura 10.109: Figura 112: Mensaje del flujo

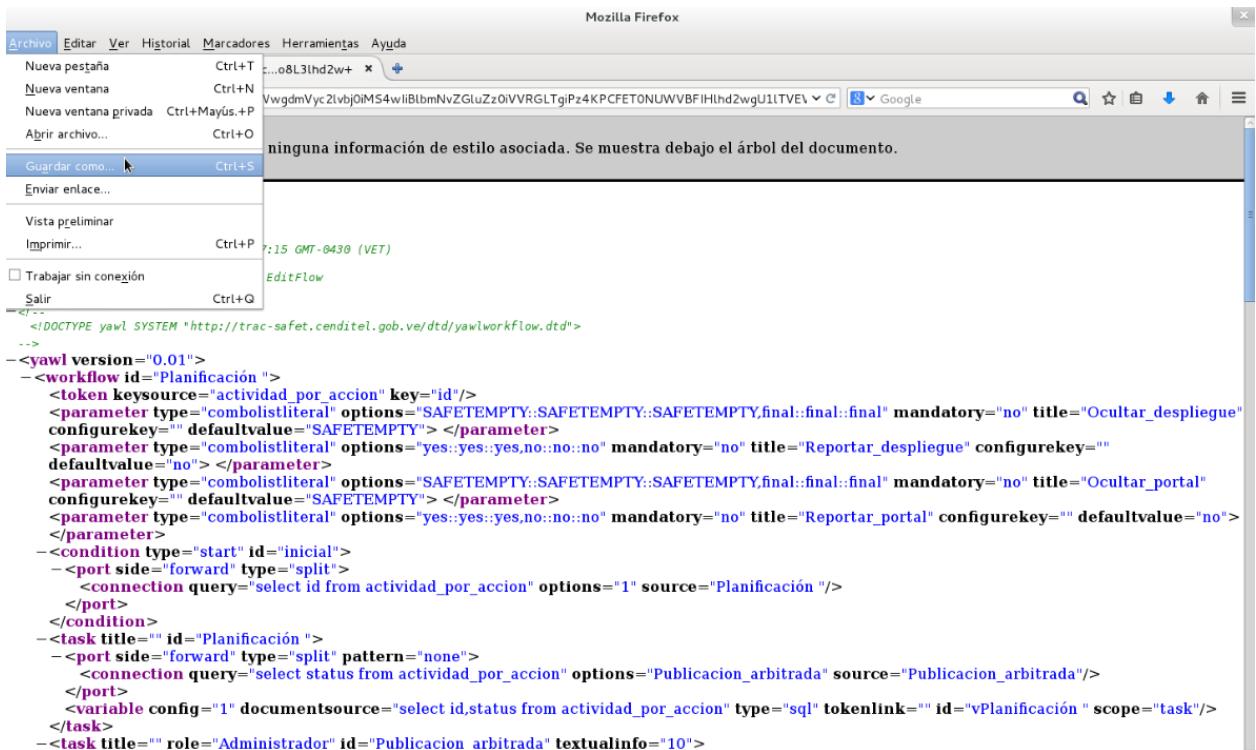


```

<!--<!--
Documento : tickets.xml
Creado : Thu Oct 30 2014 15:57:15 GMT-0430 (VET)
Autor : nombre_autor
Descripción: Archivo generado por EditFlow
-->
--<!--
<!DOCTYPE yawl SYSTEM "http://trac-safet.cenditel.gob.ve/dtd/yawlworkflow.dtd">
-->
-<yawl version="0.01">
-<workflow id="Planificación ">
-<token keysource="actividad_por_accion" key="id"/>
<parameter type="combolistliteral" options="SAFEMPTY:SAFEMPTY:SAFEMPTY:final::final::final" mandatory="no" title="Ocultar_despliegue" configurekey="" defaultvalue="SAFEMPTY"></parameter>
<parameter type="combolistliteral" options="yes::yes::yes,no::no::no" mandatory="no" title="Reportar_despliegue" configurekey="" defaultvalue="no"></parameter>
<parameter type="combolistliteral" options="SAFEMPTY:SAFEMPTY:SAFEMPTY:final::final::final" mandatory="no" title="Ocultar_portal" configurekey="" defaultvalue="SAFEMPTY"></parameter>
<parameter type="combolistliteral" options="yes::yes::yes,no::no::no" mandatory="no" title="Reportar_portal" configurekey="" defaultvalue="no"></parameter>
-<condition type="start" id="initial">
-<port side="forward" type="split">
-<connection query="select id from actividad_por_accion" options="1" source="Planificación "/>
-</port>
-</condition>
-<task title="" id="Planificación ">
-<port side="forward" type="split" pattern="none">
-<connection query="select status from actividad_por_accion" options="Publicacion_arbitrada" source="Publicacion_arbitrada"/>
-</port>
-<variable config="1" documentsource="select id,status from actividad_por_accion" type="sql" tokenlink="" id="vPlanificación " scope="task"/>
-</task>
-<task title="" role="Administrador" id="Publicacion_arbitrada" textualinfo="10">
-<port pattern="none" side="forward" type="split">
-<connection query="select status from actividad_por_accion" options="Planificacion" source="Planificacion"/>
-</port>
-<variable config="1" documentsource="select id,status from actividad_por_accion" type="sql" tokenlink="" id="vPublicacion_arbitrada" rolfield="(select

```

Figura 10.110: Figura 113: Formato XML



```

<!--<!--
Documento : tickets.xml
Creado : Thu Oct 30 2014 15:57:15 GMT-0430 (VET)
Autor : nombre_autor
Descripción: Archivo generado por EditFlow
-->
--<!--
<!DOCTYPE yawl SYSTEM "http://trac-safet.cenditel.gob.ve/dtd/yawlworkflow.dtd">
-->
-<yawl version="0.01">
-<workflow id="Planificación ">
-<token keysource="actividad_por_accion" key="id"/>
<parameter type="combolistliteral" options="SAFEMPTY:SAFEMPTY:SAFEMPTY:final::final::final" mandatory="no" title="Ocultar_despliegue" configurekey="" defaultvalue="SAFEMPTY"></parameter>
<parameter type="combolistliteral" options="yes::yes::yes,no::no::no" mandatory="no" title="Reportar_despliegue" configurekey="" defaultvalue="no"></parameter>
<parameter type="combolistliteral" options="SAFEMPTY:SAFEMPTY:SAFEMPTY:final::final::final" mandatory="no" title="Ocultar_portal" configurekey="" defaultvalue="SAFEMPTY"></parameter>
<parameter type="combolistliteral" options="yes::yes::yes,no::no::no" mandatory="no" title="Reportar_portal" configurekey="" defaultvalue="no"></parameter>
-<condition type="start" id="initial">
-<port side="forward" type="split">
-<connection query="select id from actividad_por_accion" options="1" source="Planificación "/>
-</port>
-</condition>
-<task title="" id="Planificación ">
-<port side="forward" type="split" pattern="none">
-<connection query="select status from actividad_por_accion" options="Publicacion_arbitrada" source="Publicacion_arbitrada"/>
-</port>
-<variable config="1" documentsource="select id,status from actividad_por_accion" type="sql" tokenlink="" id="vPlanificación " scope="task"/>
-</task>
-<task title="" role="Administrador" id="Publicacion_arbitrada" textualinfo="10">
-<port pattern="none" side="forward" type="split">
-<connection query="select status from actividad_por accion" options="Planificacion" source="Planificacion"/>
-</port>
-<variable config="1" documentsource="select id,status from actividad_por accion" type="sql" tokenlink="" id="vPublicacion_arbitrada" rolfield="(select

```

Figura 10.111: Figura 114: Opción

### 4° CUARTO PASO

- Colocamos el nombre del archivo con la extensión (.xml), por ejemplo (**Flujo\_de\_trabajo\_Planificación.xml**) y pulsamos el botón (**Guardar**), como se muestre la siguiente Figura 115: *Guardar archivo*:

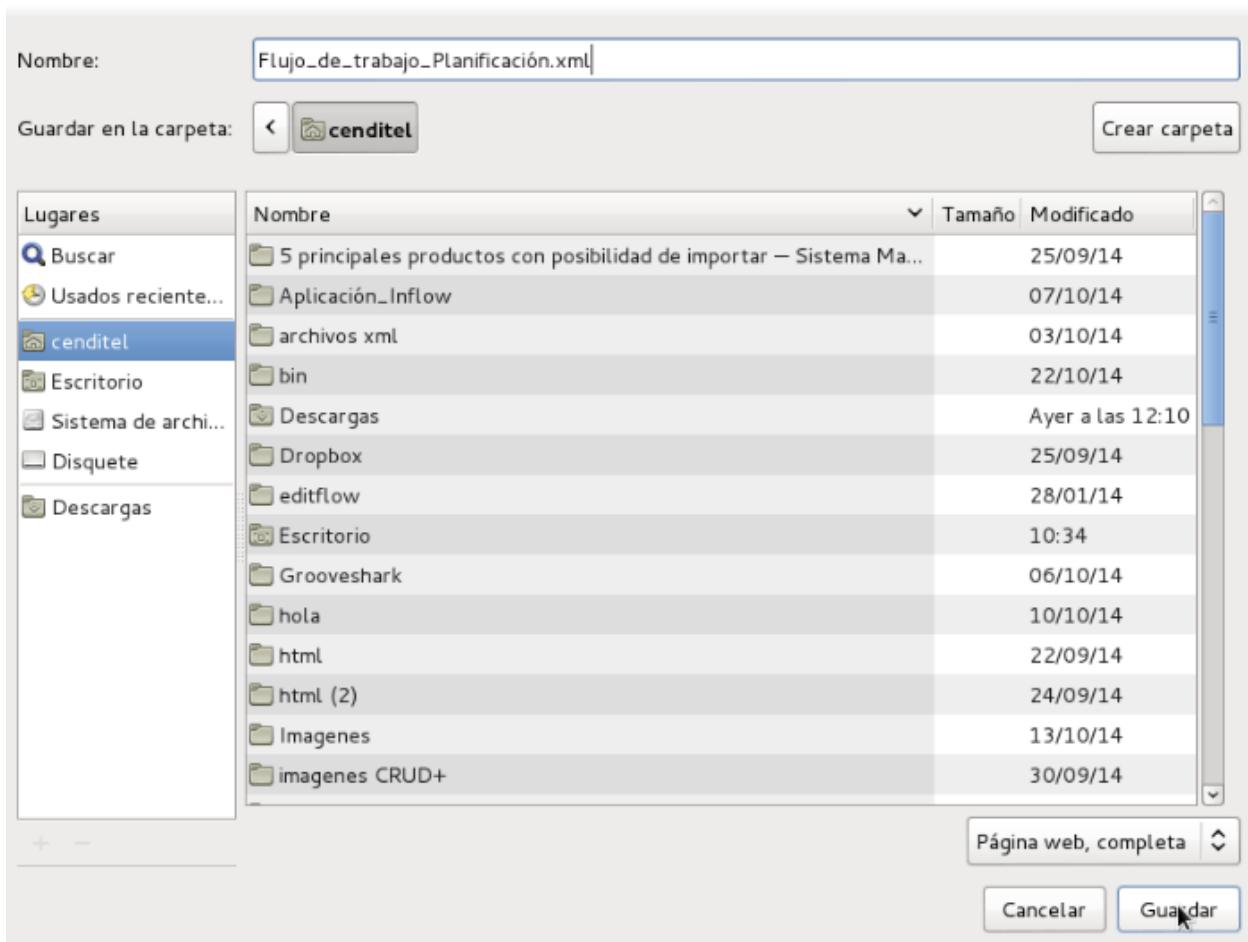


Figura 10.112: **Figura 115: Guardar archivo**

LICENCIA



Figura 10.113: Documentación del sistema POA.pdf



Figura 10.114: **Documentación PySafet by Cenditel Mérida - Venezuela** is licensed under a [Creative Commons Reconocimiento-NoComercial-SinObraDerivada 4.0 Internacional License](#). Creado a partir de la obra en <http://seguridad.cenditel.gob.ve/safet/doc>. Puede hallar permisos más allá de los concedidos con esta licencia en <http://seguridad.cenditel.gob.ve/safet/doc>.