**Filtros de actores**

Referencia : http://documentation.bonitasoft.com/creating-actor-filter

Un filtro de actores define la lista de actores que pueden ejecutar una tarea, filtrando la lista de usuarios que están mapeados a esa tarea.

Se usa cuando dentro de la lista de actores que están mapeados a una tarea, queremos dejar unos usuarios definidos.

Bonita nos proporciona un conjunto de filtros estándar:

* Un usuario asigna la tarea a un usuario específico.
* El usuario que ejecuta la tarea se la asigna al usuario que ejecutó la tarea anterior.
* El usuario que inicia la tarea se la asigna al usuario que inició el proceso.
* El gerente (Manager: en bonita, opcionalmente se puede definir un gerente para cada usuario) del usuario que inició la tarea se la asigna al gerente del usuario que inició el proceso.
* Un gerente de usuario asigna la tarea al gerente de usuario que ejecutaría la tarea si no hubiera ningún filtro.

Si la lista de definiciones de filtros anterior no conviene, Bonita nos crear definiciones de filtros personalizados.

La utilización de filtros de actores en Bonita se hace en dos pasos:

* + Definición del filtro.
  + Implementación del filtro

Esto, nos permite cambiar la implementación de los filtros sin cambiar la definición.

**Definición de filtros de actores:**

Para definir un filtro de actores, se tienen que crear siguientes ficheros:

* + Un fichero XML, que define el wizard de configuración.
  + Un fichero imagen, que contendrá el icono del filtro.
  + Un fichero properties para cada idioma que soporta el wizard.

**1-XML de definición del wizard de configuración:**

Tiene que tener los siguientes campos:

* **id**: obligatorio. Identificador de la definición del filtro de actores. Tiene que ser único.
* **version**: obligatorio. La version de la definición del filtro de actores.
* **icono**: opcional. contiene el nombre del fichero imagen que se usará para el icono del filtro.
* **category**: obligatorio. la categoría a la que pertenece el filtro de actores. En el Studio de Bonita, los filtros de actores están agrupados en categorías. Opcionalmente puede tener un icono.
* **input**: opcional, múltiple. Datos de entrada que se le pasan al filtro de actores. Cada input tiene un nombre único y un tipo Java.
* **output**: opcional, múltiple. Datos de salida que se le pasan al proceso desde el filtro de actores. Cada output tiene un nombre único y un tipo Java.
* **page**: opcional, múltiplo. Una página dentro del wizard. una página debe tener un identificador único. Una página (page) consiste en uno o varios widgets. Una página se construye a partir de los widgets que tiene definidos en el orden en el que viene.
* **widget**: uno o más por página: Un elemento dentro de la página. Debe tener un identificador (id) único, un nombre y un tipo. Los tipos siguientes están disponibles:
* text
* password
* text area
* check box
* radio button group
* select
* array
* group
* script editor
* list
* -jarDependency: opcional, múltiple. Una dependencia que debe ser añadida al wizard

**2-Fichero properties de idiomas:**

Define los labels que se muestran en la página. Se definen las siguientes properties:

* process.category = Process
* connectorDefinitionLabel = (label que se muestra para este filtro. ej. Filtro1)
* connectorDefinitionDescription = (Descripción del filtro. ej. Necesitamos que sólo pueden ejecutar las tareas aquellos usuarios cuyo nombre comience por A).
* config.pageTitle = (Titulo de página de configuración. ej. configuración)
* config.pageDescription = (Descripción de página de configuración. ej. descripción)
* connexionConfigPage.pageDescription = (Descripción de página de configuración de página. ej. Elegir aquí si se quiere asignar automáticamente la tarea a este usuario)
* autoAssign.label = (Label del botón de asignar la tarea. ej. Asignar la tarea automáticamente)
* autoAssign.description = (Label de confirmación. ej. La tarea estará asignada automáticamente al usuario resuelto)

Implementación de filtros de actores:

Para implementar un filtro de actor, se necesitan dos artefactos:

* + Un fichero de recursos, que indica los identificadores de la definición e implementación del filtro.
  + Una clase Java, que es la que define la lista de usuarios permitidos.

**1. Fichero de recursos:**

Un fichero XML, que debe seguir un esquema definido en http://www.bonitasoft.org/ns/connector/implementation/6.0 y que debe declarar los siguientes campos:

* id y versión de la definición del filtro.
* id y versión de la implementación del filtro.
* El conjunto de dependencias que son requeridas para la implementación del filtro.

**2. Clase Java de la implementación:**

Debe extender la clase **AbstractUserFilter** del propio Bonita y sobrescribir los siguientes métodos:

* *public void validateInputParameters()* : comprobar que la configuración del filtro de actores está bien definida.
* *public List<Long> filter(final String actorName)* : Devuelve una lista de los identificadores de los usuarios que se desea permitir para un nombre de usuario específico.
* *public boolean shouldAutoAssignTaskIfSingleResult()* : para asignar la tarea al usuario si el resultado contiene un solo id (sólo un usuario).

**Ejemplo de un filtro de actores:**

**1. Definición del filtro de actores:**

-XML de definición del wizard de configuración:

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

<definition:ConnectorDefinition xmlns:definition="http://www.bonitasoft.org/ns/connector/definition/6.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<id>Filtro1-definición</id>

<version>1.0.0</version>

<icon>filtro1.png</icon>

<category id="process" icon="process.gif" ></category>

<input name="autoAsignar" type="java.lang.Boolean" mandatory="false" defaultValue="true"/>

<page id="config">

<widget id="autoAsignado" inputName="autoAsignado" xsi:type="definition:Checkbox"></widget>

</page>

</definition:ConnectorDefinition>

-Fichero properties de idiomas:

process.category = Proceso

connectorDefinitionLabel = Gerente de usuario

connectorDefinitionDescription = Un gerente de usuario puede ejecutar su tarea.

config.pageTitle = Configuración

config.pageDescription = Configuración del filtro Gerente-Usuario del proceso.

connexionConfigPage.pageDescription = Elija aquí si desea asignar la tarea automáticamente a este usuario.

autoAssign.label = Asignar tarea automáticamente.

autoAssign.description = La tarea será asignada al usuario resuelto.

**2. Implementación del filtro de actores**

-fichero de recursos: debe seguir el esquema definido en "http://www.bonitasoft.org/ns/connector/implementation/6.0"

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

<implementation:connectorImplementation xmlns:implementation="http://www.bonitasoft.org/ns/connector/implementation/6.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<definitionId>Filtro1-definicion</definitionId> //fijarse aquí que se tiene que referir a la definición que se quiere emplear en esta implementación

<definitionVersion>1.0.0</definitionVersion> <implementationClassname>org.bonitasoft.userfilter.initiator.manager.ProcessinitiatorManagerUserFilter</implementationClassname> //Clase Java creada para esta implementación

<implementationId>Filtro1-implementacion</implementationId>

<implementationVersion>1.0.0</implementationVersion>

<jarDependencies>

<jarDependency>filtro1-impl-1.0.0-SNAPSHOT.jar</jarDependency>

</jarDependencies>

</implementation:connectorImplementation>

-Clase Java para definir los usuarios permitidos: Por ejemplo para asignar la tarea al gerente del usuario que creó la tarea.

public class ProcessinitiatorManagerUserFilter extends AbstractUserFilter{

*@Override*

public void *validateInputParameters()* throws ConnectorValidationException{}

*@Override*

public List<Long> *filter(final String actorName)* throws UserFilterException {

try {

final long processInstanceId = getExecutionContext().getParentProcessInstanceId(); //sacar el id de esta instancia(expediente) del proceso

long processInitiator = getAPIAccessor().getProcessAPI().getProcessInstance(processInstanceId).getStartedBy(); //sacar el id del usuario que inició el proceso.

return Arrays.asList( getAPIAccessor().getIdentityAPI().getUser(processInitiator).getManagerUserId()); //devuelve la lista de los id's de los gerentes del usuario iniciador del proceso

} catch (final BonitaException e) {

throw new UserFilterException(e);

}

}

*@Override*

public boolean *shouldAutoAssignTaskIfSingleResult()* {

final Boolean autoAssignO = (Boolean) getInputParameter("autoAsignado"); //checkBox definido en la página(page) de la definición del filto

return autoAssignO == null ? true : autoAssignO; //si es null(no se definición en la página) será automáticamente asignada. Si no, coge el valor del checkbox.

}

}

Utilización del filtro

Una vez finalizada la definición e implementación del filtro, se tiene que comprimir en un fichero Zip e importar desde el Studio para añadirlo al proceso deseado.

-Se recomiendo utilizar maven para la construcción de tanto la definición como la implementación del filtro. Se tienen que crear dos proyectos maven y crear los ficheros mencionados antes para cada proyecto (definición e implementación).

-Una vez creados los proyectos y los ficheros que deben contener, se ejecuta el siguiente comando de mvn para construir el Zip: mvn clean install.

-importar el Zip al Studio en : Desarrollo --> fitros de actores --> import y finalmente probarlo generando en .bar e importarlo al portal.

-A veces por un problema de empaquetado del filtro, surgen problemas a la hora de importarlo como el mensaje de error de que no hay un fichero de definición. En este caso, tenemos que hacerlo a mano desde el Bonita Studio.

En desarrollo/filtro de actores hacemos una nueva definición (con los datos que tenemos en el fichero de definición previamente hecho: id, versión, nombre…etc). Luego, le damos a nueva implementación donde hacemos lo mismo. Al final, nos sale una página de eclipse dónde hacemos la clase Java. Copiamos la clase que tenemos hecha o simplemente modificar los métodos que nos salen.

Al final, aplicar el filtro en las calles(sendas que deseemos dentro del proceso). Hacer click en una Senda, en General/actores hacemos click en definir. Nos saldrán los filtros definidos en Bonita más los filtros que hemos hecho nosotros. Seleccionamos el filtro que queremos y listo. Ya se puede desplegar el proceso.