



Guía de Configuración de Workflows

Control de versiones	2
Workflow	3
Variables a evaluar	3
Opciones para configurar un workflow	4
Panel de administración	5
Funciones	5
Editor de workflows	5
Introducción	6
Elementos	6
Acciones	12
Ejemplos de workflows	12
Creación de una entidad (con un único rol validador):	12
Petición de adherencia (con dos roles validadores)	15
Cambiar la unidad organizativa de una entidad (con tres roles validadores)	15
Creación de una entidad con evaluación de atributos para permitir bifurcaciones (con roles validadores dependientes del objeto a validar)	16
XML	17
Elementos del XML	23

Control de versiones

Versión	Fecha de modificación	Responsable	Aprobador	Resumen de cambios
1.0	28/10/2022	Anjana Producto	Anjana Producto	Creación del documento

Workflow

Un workflow es el flujo de validación que tiene que seguir cada entidad o relación de Anjana para quedar aprobada en Anjana y que pueda ser utilizada por otras estructuras. Estas modificaciones pueden ser un cambio de estado, una solicitud de adherencia a un DSA, una modificación de algún atributo...

Para completar un workflow en Anjana es posible contar con tareas de envío de notificaciones, validaciones o evaluación de objetos para permitir bifurcaciones en distintos caminos dentro del mismo workflow.

Variables a evaluar

En los workflows configurados en Anjana es posible comprobar el valor de ciertos parámetros para poder configurar bifurcaciones en ellos.

Estos parámetros son:

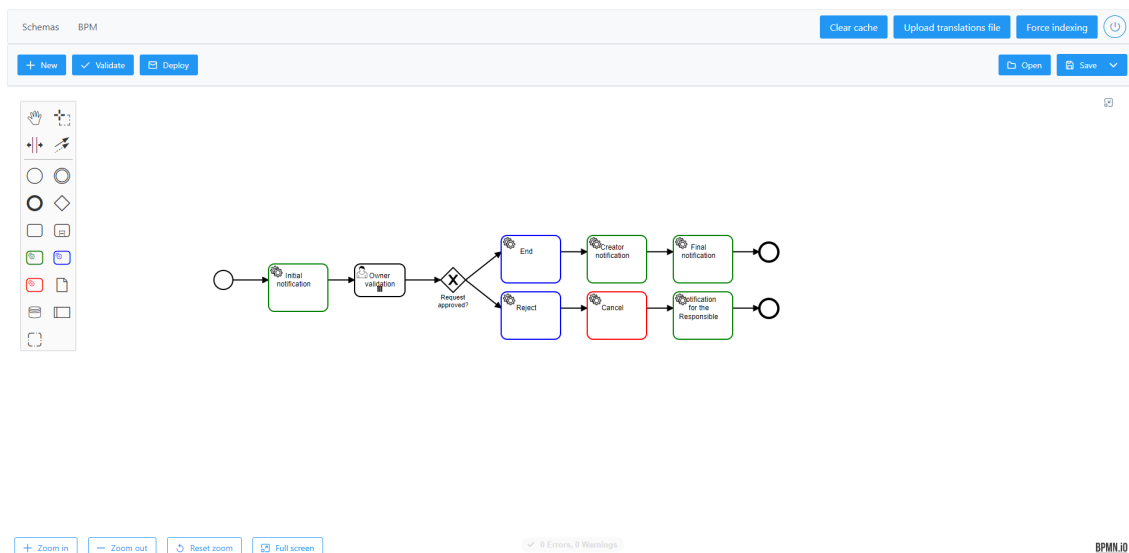
- Cualquier atributo de la plantilla del objeto que está siendo validado. Un ejemplo de comparación es: para el atributo con name='pi', `#{pi == 'true'}`
- IS_VERSION: si el workflow es de versionado (cadena de caracteres así que es 'true' en caso de versionado y 'false' en caso de modificación). Un ejemplo de comparación es: `#{IS_VERSION == 'true'}`
- ORIGINAL_OBJECT_ID: identificador interno del objeto original en caso de que el workflow sea de versionado (número entero). Un ejemplo de comparación es: `#{ORIGINAL_OBJECT_ID >= 10}` y `#{ORIGINAL_OBJECT_ID < 10}`
- OBJECT_ID: identificador interno de Anjana para el objeto (número entero). Un ejemplo de comparación es: `#{OBJECT_ID >= 1}`
- OBJECT_NAME: nombre del objeto (cadena de caracteres). Un ejemplo de comparación es: `#{OBJECT_NAME == "Clientes"}` y `#{OBJECT_NAME != "Clientes"}`
- OBJECT_SUB_TYPE: nombre del subtipo del objeto (cadena de caracteres). Un ejemplo de comparación es: `#{OBJECT_SUB_TYPE == 'DATASET'}` y `#{OBJECT_SUB_TYPE != 'DATASET'}`
- OBJECT_TYPE: ENTITY o RELATIONSHIP (cadena de caracteres). Un ejemplo de comparación es: `#{OBJECT_TYPE == 'ENTITY'}` y `#{OBJECT_TYPE != 'ENTITY'}`

- ORGANIZATIONAL_UNIT: una o dos unidades organizativas a las que pertenece el objeto (una en caso de entidad y dos en caso de relación, por sus dos extremos) (lista de valores). Un ejemplo de comparación es: `${execution.getVariable("ORGANIZATIONAL_UNIT").contains("HQ/Legal")}` y `${!execution.getVariable("ORGANIZATIONAL_UNIT").contains("HQ/Legal")}`
- ROLE_NAME: rol del usuario que lanza el workflow (cadena de caracteres). Un ejemplo de comparación es: `${ROLE_NAME== 'data_owner'}` y `${ROLE_NAME != 'data_owner'}`
- SENDER_USER: usuario nominal que lanza el workflow (cadena de caracteres). Un ejemplo de comparación es: `${ SENDER_USER == 'maria.gonzalez'}` y `${ SENDER_USER != 'maria.gonzalez'}`
- ACCEPTED: aprobación del usuario a la validación (cadena de caracteres) así que es 'true' en caso de aprobación y 'false' de rechazo). Un ejemplo de comparación es: `${ACCEPTED == 'true'}` y `${ACCEPTED == 'false'}`
- WORKFLOW_TYPE: tipo de workflow a ejecutar (valores posibles: 'CREATE', 'MODIFY', 'ACTIVATE', 'DEACTIVATE', 'TRANSFER', 'DEPRECATE', 'ADHERENCE', 'DISADHERENCE'). Un ejemplo de comparación es: `${WORKFLOW_TYPE == 'CREATE'}` y `${WORKFLOW_TYPE != 'CREATE'}`

Opciones para configurar un workflow

Es posible configurar workflows en Anjana de dos formas:

- Desde el panel de administración, en el apartado de BPM, desde donde se accede al editor de workflows: <http://{{hostname}}/admin/bpmn-manager/edit>



- Creando directamente el xml para Activiti

A continuación se exponen ambas opciones.

Panel de administración

Funciones

En el front de administración están disponibles las siguientes opciones:

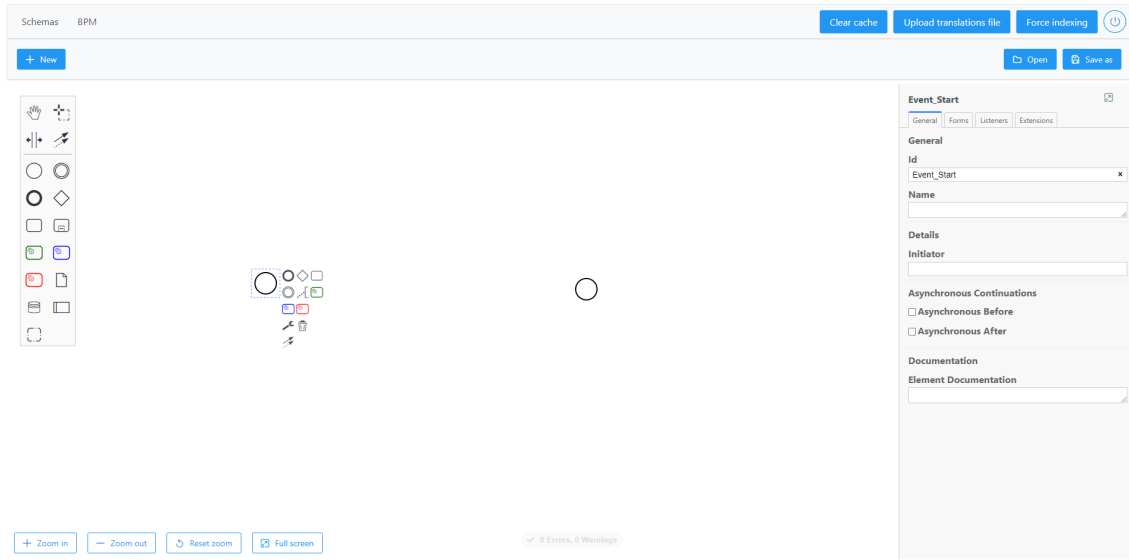
- Open → Permite importar un workflow previamente creado a través de su xml.
- Save as → Guarda el workflow configurado desde el editor de workflow. Esta opción permite guardar el xml o imagen en el Filesystem o desplegarlo en Anjana para poder usarlo.
- New → Abre una pizarra desde donde se puede configurar un nuevo workflow
- Validate → Valida la corrección del workflow configurado. Solamente aparecerán cuando el workflow se haya guardado a través de la API.
- Deploy → Despliega el workflow para poderlo lanzar desde Anjana. Solamente aparecerán cuando el workflow se haya guardado a través de la API.

Editor de workflows

Será el entorno de trabajo desde donde se creará un workflow, donde estarán las herramientas necesarias para incluir las acciones y flujos que formarán la estructura del workflow.

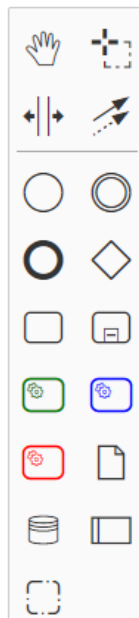
Introducción

La manera más sencilla de empezar un workflow es incluyendo en la pizarra el inicio y final del workflow para, a partir de ahí, ir incluyendo los elementos necesarios para formar el flujo de trabajo.



Elementos

Los elementos que se pueden añadir en un workflow, entre el inicio y el final aparecen a la izquierda de la pizarra de workflow.



Estos elementos son:

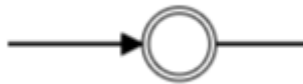
- **StartEvent**

Inicio del workflow, necesario para el correcto funcionamiento del workflow.



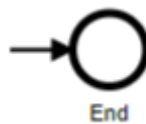
- Intermediate/Boundary Event

Se encargan de realizar las funciones de 'listener', son eventos que no se pueden lanzar, se activan automáticamente cuando detectan un cierto tipo de disparador, de esa manera el workflow se interrumpe y se sigue el flujo de secuencia que sale del evento.



- EndEvent

Final del workflow, necesario para el correcto funcionamiento del workflow.



- Sequence Flow

Son las líneas que unen los distintos elementos, pueden ser de dos tipos:

- Condicionales → Se tiene que cumplir una condición para que pase el flujo de ejecución al siguiente elemento, como por ejemplo, que el usuario valide como OK la task.

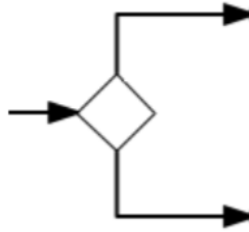


- Incondicionales → Siempre pasan al siguiente elemento del flujo de ejecución.



- Gateway

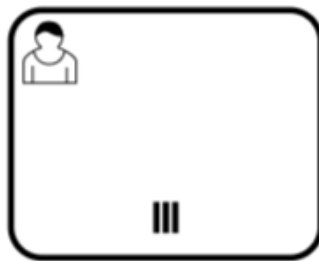
Se usa para diversificar el flujo de ejecución del workflow, permite realizar tareas en paralelo.



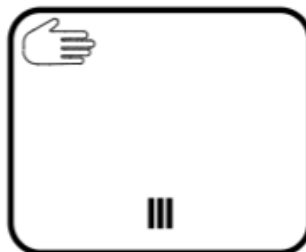
- Task

Tarea individual dentro del workflow, se pueden clasificar en diferentes tipos dependiendo de :

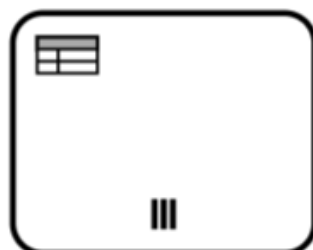
- Su función :
 - User → Necesaria la acción de un usuario para continuar el flujo.



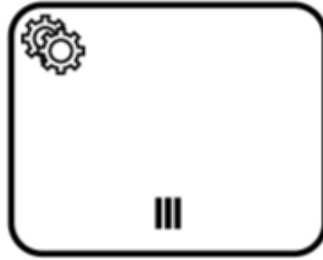
- Manual → Es una tarea externa al flujo interno del motor BPM, por tanto, no se conoce el proceso que realiza internamente, solo se controla su estado para saber si ha terminado.



- Business Rule → Se usa para ejecutar de manera síncrona una o más reglas definidas por el cliente. Actualmente Activiti usa archivos .drl para incluir dichas reglas.



- Service → Lanzan la ejecución de una clase java, dicha clase puede realizar funciones diversas, para Anjana se han creado las siguientes:



- SendNotificationJavaDelegate → Envía una notificación que previamente debe de estar creada en la tabla notification del schema hermes en el modelo de datos de Anjana. Para configurar esta task, sólo es necesario incluir el código de la notificación que se desea enviar en el apartado 'Field Injections'. El uso de este componente como notificación para validación queda deprecado y se eliminará en la siguiente versión.

A screenshot of a configuration interface for a task named 'SEND_INIT_NOTIFICATION'. The interface has several tabs: 'General', 'Listeners', 'Input/Output', 'Field Injections' (which is selected), and 'Extensions'. Under the 'Field Injections' tab, there is a 'Notification Code' dropdown menu. Below that is a 'Fields' list with one entry: '0 : notificationCode'. Underneath the list are fields for 'Name' (containing 'notificationCode'), 'Type' (a dropdown menu set to 'String'), and 'Value' (containing '1').

- Se recomienda fusionar con la validación del rol en una task de usuario configurándose de la siguiente forma. El uso de esta tarea para notificar validaciones de usuario está deprecado.

DATA_OWNER_VALIDATOR

< General Forms Listeners Input/Output **Field Injections** >

Field Injections

Validator Role
 DC - Data Owner

Fields

0 : validatorRole
1 : notificationCode

Name
 notificationCode

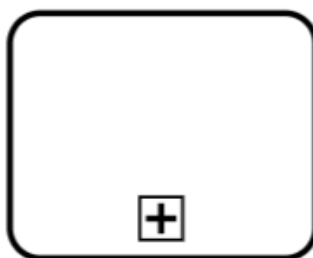
Type
 String

Value
 7

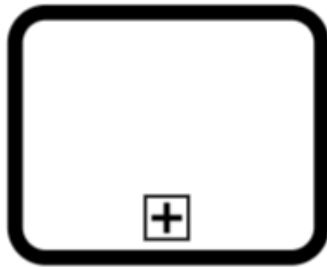
- LastExecutionJavaDelegate → Se usa para que Anjana realice los últimos procesos al finalizar un workflow. Es necesario llamar a esta tarea siempre antes de terminar el workflow, tanto si el resultado es OK como KO.
- CancelTaskWorkflowJavaDelegate → Cuando uno de los validadores no aprueba el workflow el flujo debe de pasar por esta task para indicar a Activi que el proceso ha sido cancelado.
- Script → Cuando el flujo del workflow llega a dicha tarea se ejecuta el script.



- Expanded SubProcess → Sirve para agrupar diferentes task, se suele usar cuando el resultado se quiere que dependa de todas las task que están agrupadas dentro del subprocesso.

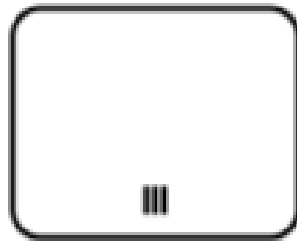


- Call Activity → Es un tipo de subprocesso que agrupa varias task pero este tipo de subprocesso puede ser llamado desde diferentes partes del flujo del workflow.



- Su ejecución :

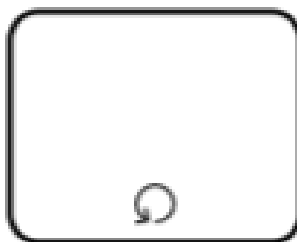
- En paralelo → El proceso de varias task se puede realizar en paralelo.



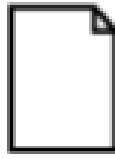
- Secuencial → El flujo no puede avanzar hasta que se realice la task.



- Loop → La tarea se ejecutará n veces.



- DataObjectReference → Activiti permite definir data object como parte de un proceso o subproceso. De esta manera se pueden incluir estructuras XML complejas que se pueden importar desde XSD.



- DataStorage Reference → Se pueden incluir referencias dentro del flujo del workflow a una base de datos.



- Pool/Participant → Permite crear otra pizarra de workflow.
- Group → Permite agrupar diferentes pizarras de workflows.

Acciones

- Change Type → Permite cambiar el tipo del elemento.



- Delete → Borra el elemento.



- Connect using Sequence/MessageFlow or Association → Permite unir elementos entre sí.

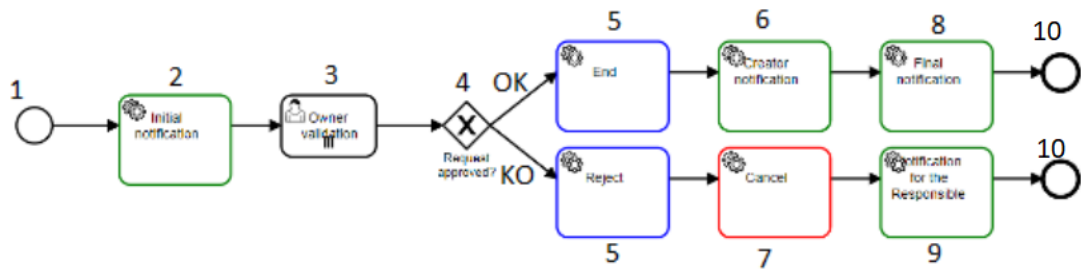


- Append Text Annotation → Sirve para incluir notas a un elemento.



Ejemplos de workflows

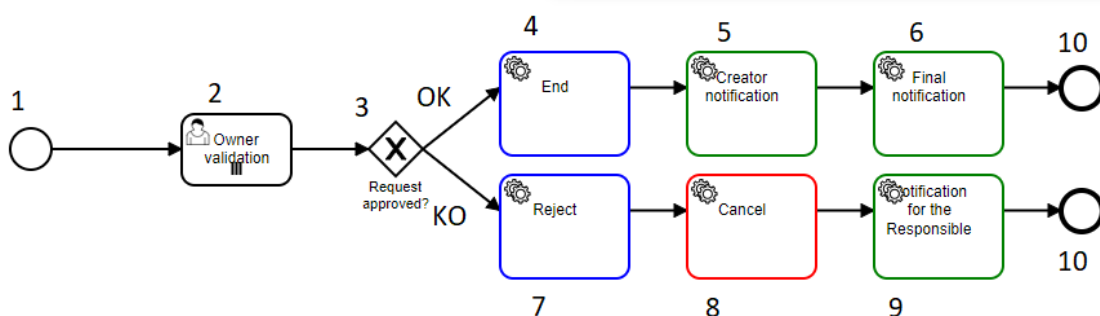
Creación de una entidad (con un único rol validador):



El proceso de este workflow será el siguiente:

1. Se inicia el workflow (startEvent).
2. Envío de notificación al rol owner (Sent Notification Task).
3. Validación del rol owner (User Task).
4. Gateway de validación.
5. Se ejecutan los últimos procesos en Anjana antes de acabar el workflow, tanto si el workflow ha sido aprobado como rechazado (LastExecution Task).
6. Notificación al usuario creador (Sent Notification Task).
7. Se avisa a Activi del workflow cancelado (Cancel Workflow Task).
8. Notificación final de que el workflow ha sido aprobado (Sent Notification Task).
9. Notificación al responsable, donde se le indica que el workflow ha sido rechazado (Sent Notification Task).
10. Final del workflow (End Event).

Creación de una entidad reduciendo las tasks configuradas. En la task de validación se incorpora el envío de la notificación pertinente:



El proceso de este workflow será el siguiente:

1. Se inicia el workflow (startEvent).
2. Envío de notificación al rol owner y validación (User Task). Para el envío de la notificación, en la task se incluye el notificationCode de la misma forma que se indicaba en el tipo de task SentNotification.
3. Gateway de validación.

4. Se ejecutan los últimos procesos en Anjana antes de acabar el workflow, tanto si el workflow ha sido aprobado como rechazado (LastExecution Task).
5. Notificación al usuario creador (Sent Notification Task).
6. Se avisa a Activi del workflow cancelado (Cancel Workflow Task).
7. Notificación final de que el workflow ha sido aprobado (Sent Notification Task).
8. Notificación al responsable, donde se le indica que el workflow ha sido rechazado (Sent Notification Task).
9. Final del workflow (End Event).

Esta reducción de tasks se ha llevado a cabo fusionando el envío de la notificación y la validación del rol en una task de usuario configurándose así:

DATA_OWNER_VALIDATOR

< General Forms Listeners Input/Output **Field Injections** f >

Field Injections

Validator Role

DC - Data Owner

Fields

0 : validatorRole
1 : notificationCode

Name

notificationCode

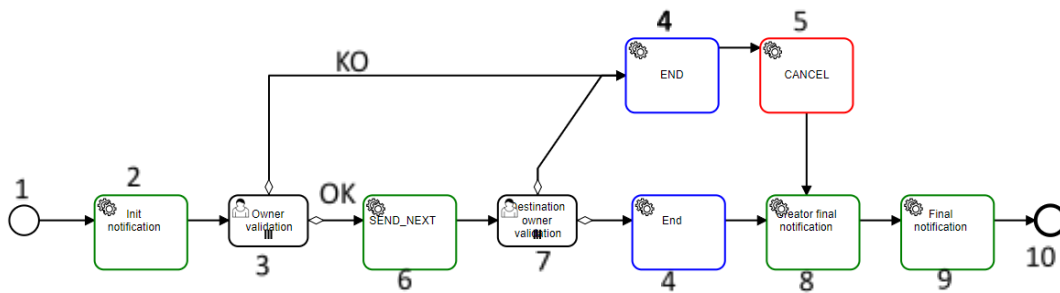
Type

String

Value

7

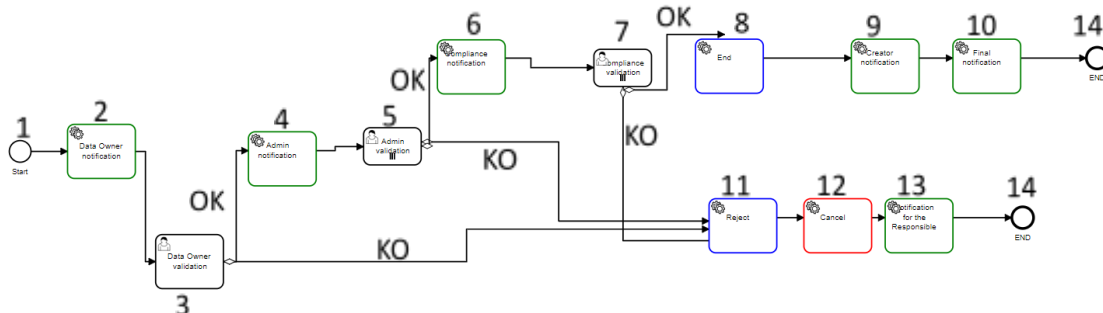
Petición de adherencia (con dos roles validadores)



Pasos del workflow:

1. Inicio del workflow (StartEvent).
2. Envío de notificación al rol owner para validación (Sent Notification Task).
3. Validación del rol owner (User Task).
4. Se ejecutan los últimos procesos en Anjana antes de acabar el workflow, tanto si el workflow ha sido aprobado como rechazado (LastExecution Task).
5. Se informa a Activiti del rechazo (Cancel Workflow Task).
6. Al aprobar el rol owner se envía la notificación al siguiente rol validador, el destination owner (Sent Notification Task).
7. Validación del rol destination owner (User Task).
8. Notificación final al creador (Sent Notification Task).
9. Notificación final a los roles validadores (Sent Notification Task).
10. Final del workflow (End Event).

Cambiar la unidad organizativa de una entidad (con tres roles validadores)

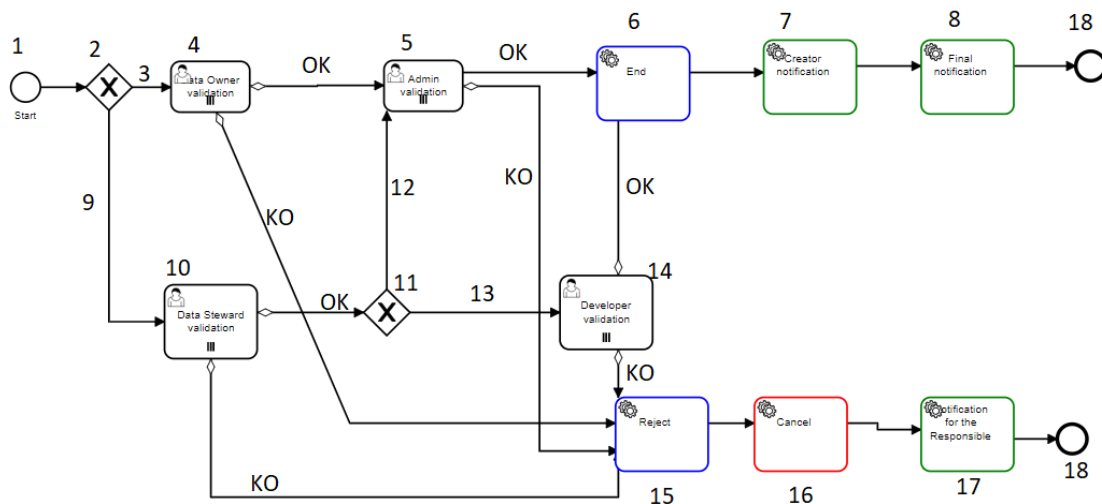


Los pasos del workflow son los siguientes:

1. Inicio del workflow (StartEvent).

2. Envío de notificación al rol data owner para validación (Sent Notification Task).
3. Validación del rol data owner (User Task).
4. Al aprobar el rol data owner, se envía una notificación al rol admin para validación (Sent Notification Task).
5. Validación del rol admin (User Task).
6. Al aprobar el admin el paso del workflow se envía una notificación al último rol implicado, compliance para validación (Sent Notification Task).
7. Validación del rol compliance (User Task).
8. Si el rol compliance valida el último paso se inician los procesos internos de Anjana que cambiarán la unidad organizativa del objeto (Last Execution Task).
9. Notificación final al creador (Sent Notification Task).
10. Notificación final a los roles validadores (Sent Notification Task).
11. Si el rol compliance indica que rechaza el workflow se inician los procesos internos de Anjana (Last Execution Task).
12. Se informa a Activiti de la cancelación del workflow (Cancel Workflow Task).
13. Se envía notificación del KO al responsable del workflow (Sent Notification Task).
14. Final del workflow (End Event).

Creación de una entidad con evaluación de atributos para permitir bifurcaciones (con roles validadores dependientes del objeto a validar)



El proceso de este workflow será el siguiente:

1. Se inicia el workflow (startEvent).
2. Gateway para la bifurcación siguiente

3. Condición que comprueba si el objeto a validar NO es un dataset (`${OBJECT_SUB_TYPE != "DATASET"}`)
4. Envío de notificación al rol data owner y validación (User Task).
5. Envío de notificación al rol admin y validación (User Task).
6. Si el rol admin aprueba se inician los procesos internos de Anjana que cambiarán la unidad organizativa del objeto (Last Execution Task).
7. Notificación final al creador (Sent Notification Task).
8. Notificación final a los roles validadores(Sent Notification Task).
9. Condición que comprueba si el objeto a validar SÍ es un dataset (`${OBJECT_SUB_TYPE=="DATASET"}`)
10. Envío de notificación al rol data steward y validación (User Task).
11. Gateway para la bifurcación siguiente
12. Condición que comprueba si el atributo pi del objeto a validar es true (`${pi == 'true'}`)
13. Condición que comprueba si el atributo pi del objeto a validar es false (`${pi == 'false'}`)
14. Envío de notificación al rol developer y validación (User Task).
15. Si cualquiera de los validadores rechaza el workflow se inician los procesos internos de Anjana (Last Execution Task).
16. Se informa a Activiti de la cancelación del workflow (Cancel Workflow Task).
17. Se envía notificación del KO al responsable del workflow (Sent Notification Task).
18. Final del workflow (End Event).

XML

La configuración de los workflows se puede realizar también de forma manual mediante un fichero XML. Este proceso es de por sí más complicado porque hay que conocer a priori el formato que va a tener la definición de un workflow de Activiti. Por suerte estos ficheros XML suelen seguir un esquema donde tenemos establecido cuál va a ser la estructura de estos ficheros. Para el caso que nos ocupa se puede consultar en <http://bpmn.sourceforge.net/schemas/BPMN20.xsd>.

Observando el contenido del fichero se puede dividir el mismo en dos partes bien diferenciadas.

En primer lugar está la definición de la secuencia de workflows y las tareas que ejecutan cada una de estas secuencias.

En segundo lugar se establece la estructura gráfica que permitirá poder utilizar el front de administración comentado en el punto anterior. Para poder generar esta estructura gráfica es necesario abrirlo y guardarlo en un primer momento con la utilidad Yaoqiang BPMN Editor.

Como muestra de configuración vamos a indicar el formato del diagrama generado con Activiti. Se ven claramente las zonas donde se definen las actividades que realiza cada secuencia del workflow:

```
<serviceTask      id="PROCESSOWNER_NOTIFICATION"      name="P.Owner
notification"
                  implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJ
avaDelegate">
  <extensionElements>
    <activiti:field name="notificationCode" stringValue="7" />
  </extensionElements>
  <incoming>flow1</incoming>
  <outgoing>flowN2</outgoing>
</serviceTask>
```

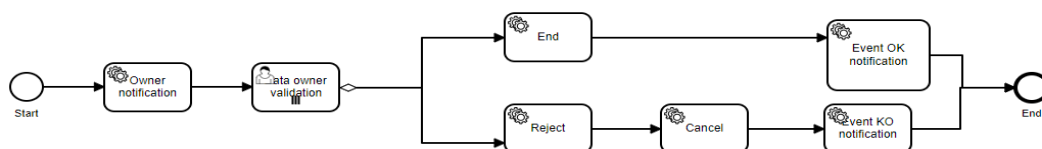
Se define la tarea PROCESSOWNER_NOTIFICATION y la clase Java que se encarga de implementarla. Vemos también que nos indica que quien origina esta tarea es la secuencia del workflow llamada "flow1" y que cuando termine se ejecutará la tarea "flowN2".

Un ejemplo donde se definen estas secuencias del workflow sería:

```
<sequenceFlow    id="flow1"      sourceRef="Event_Start"
targetRef="PROCESSOWNER_NOTIFICATION" />
<sequenceFlow    id="flowN2"    sourceRef="PROCESSOWNER_NOTIFICATION"
targetRef="PROCESSOWNER_VALIDATOR" />
.....
```

Se puede observar que se indica el nombre de la secuencia del workflow, quien lo puede originar (*sourceRef*) y qué tarea se va a ejecutar (*targetRef*).

Para el siguiente workflow



El xml generado sería el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions      xmlns="http://www.omg.org/spec/BPMN/20100524/MODEL"
xmlns:activiti="http://activiti.org/bpmn"
```

```

xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI"
xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
xmlns:di="http://www.omg.org/spec/DD/20100524/DI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:tns="DC"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:yaoqiang="http://bpmn.sourceforge.net" id="_1613572957230" name=""
targetNamespace="DC" exporter="Yaoqiang BPMN Editor" exporterVersion="5.3"
xsi:schemaLocation="http://www.omg.org/spec/BPMN/20100524/MODEL
http://bpmn.sourceforge.net/schemas/BPMN20.xsd">
  <process id="ADHERENCE" name="Request of adherence" processType="None"
isClosed="false" isExecutable="true">
    <extensionElements>
      <yaoqiang:description />
      <yaoqiang:pageFormat height="841.8897637795276"
imageableHeight="831.8897637795276" imageableWidth="588.1102362204724"
imageableX="5.0" imageableY="5.0" orientation="0" width="598.1102362204724" />
      <yaoqiang:page background="#FFFFFF" horizontalCount="1" verticalCount="1" />
    </extensionElements>
    <startEvent id="Event_Start" name="Start">
      <outgoing>flow1</outgoing>
      <outputSet />
    </startEvent>
    <serviceTask id="DATA_OWNER_NOTIFICATION" name="Owner notification"
implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJavaDelegate">
      <extensionElements>
        <activiti:field name="notificationCode" stringValue="20" />
      </extensionElements>
      <incoming>flow1</incoming>
      <outgoing>flow2</outgoing>
    </serviceTask>
    <userTask id="DATA_OWNER_VALIDATOR" name="Data owner validation"
implementation="##unspecified">
      <extensionElements>
        <activiti:field name="validatorRole" stringValue="data_owner" />
      </extensionElements>
      <incoming>flow2</incoming>
      <outgoing>Flow3_OK</outgoing>
      <outgoing>Flow3_KO</outgoing>
      <multiInstanceLoopCharacteristics
activiti:collection="MODULE_ORGANIZATIONAL_UNIT">
        <completionCondition xsi:type="tFormalExpression">${ACCEPTED !=
'true'}</completionCondition>
      </multiInstanceLoopCharacteristics>
    </userTask>
    <serviceTask id="END_KO_NOTIFICATION" name="Event KO notification"
implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJavaDelegate">
      <extensionElements>
        <activiti:field name="notificationCode" stringValue="11" />
      </extensionElements>
      <incoming>CANCEL</incoming>
      <outgoing>Flow_065hvgf</outgoing>

```

```

</serviceTask>
    <serviceTask id="FINAL_EXEC" name="End" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.LastExecutionJavaDelegate">
    <incoming>Flow3_OK</incoming>
    <outgoing>Flow_1uee4uh</outgoing>
</serviceTask>
    <serviceTask id="FINAL_REJECT_EXEC" name="Reject" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.LastExecutionJavaDelegate">
    <incoming>Flow3_KO</incoming>
    <outgoing>KO</outgoing>
</serviceTask>
    <serviceTask id="CANCEL_EXEC" name="Cancel" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.CancelTaskWorkflowDelegate">
    <incoming>KO</incoming>
    <outgoing>CANCEL</outgoing>
</serviceTask>
<endEvent id="Event_End" name="End">
    <incoming>Flow_065hvgf</incoming>
    <incoming>Flow_03tyub3</incoming>
    <inputSet />
</endEvent>
        <sequenceFlow id="flow1" sourceRef="Event_Start"
targetRef="DATA_OWNER_NOTIFICATION" />
        <sequenceFlow id="flow2" sourceRef="DATA_OWNER_NOTIFICATION"
targetRef="DATA_OWNER_VALIDATOR" />
        <sequenceFlow id="Flow3_OK" sourceRef="DATA_OWNER_VALIDATOR"
targetRef="FINAL_EXEC">
            <conditionExpression xsi:type="tFormalExpression">${ACCEPTED ==
'true'}</conditionExpression>
        </sequenceFlow>
        <sequenceFlow id="Flow3_KO" sourceRef="DATA_OWNER_VALIDATOR"
targetRef="FINAL_REJECT_EXEC">
            <conditionExpression xsi:type="tFormalExpression">${ACCEPTED ==
'false'}</conditionExpression>
        </sequenceFlow>
        <sequenceFlow id="KO" sourceRef="FINAL_REJECT_EXEC" targetRef="CANCEL_EXEC" />
        <sequenceFlow id="CANCEL" sourceRef="CANCEL_EXEC" targetRef="END_KO_NOTIFICATION"
/>
        <sequenceFlow id="Flow_1uee4uh" sourceRef="FINAL_EXEC"
targetRef="END_OK_NOTIFICATION" />
        <sequenceFlow id="Flow_065hvgf" sourceRef="END_KO_NOTIFICATION"
targetRef="Event_End" />
        <serviceTask id="END_OK_NOTIFICATION" name="Event OK notification"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJavaDelegate">
        <extensionElements>
            <activiti:field name="notificationCode" stringValue="10" />
        </extensionElements>
        <incoming>Flow_1uee4uh</incoming>
        <outgoing>Flow_03tyub3</outgoing>
    </serviceTask>
        <sequenceFlow id="Flow_03tyub3" sourceRef="END_OK_NOTIFICATION"
targetRef="Event_End" />
</process>

```

```

    <bpmndi:BPMNDiagram id="Yaoqiang_Diagram-ADHERENCE_-_ARCHITECT" name="Untitled
Diagram" resolution="96">
    <bpmndi:BPMNPlane bpmnElement="ADHERENCE">
    <bpmndi:BPMNEdge id="Yaoqiang-CANCEL" bpmnElement="CANCEL">
    <di:waypoint x="742" y="264" />
    <di:waypoint x="817" y="264" />
    <bpmndi:BPMNLabel>
    <dc:Bounds x="906.33" y="253.58" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="Yaoqiang-KO" bpmnElement="KO">
    <di:waypoint x="589" y="264" />
    <di:waypoint x="657" y="264" />
    <bpmndi:BPMNLabel>
    <dc:Bounds x="761.32" y="253.58" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="Yaoqiang-Flow3_KO" bpmnElement="Flow3_KO">
    <di:waypoint x="342" y="217" />
    <di:waypoint x="423" y="217" />
    <di:waypoint x="423" y="280" />
    <di:waypoint x="504" y="280" />
    <bpmndi:BPMNLabel>
    <dc:Bounds x="414.5" y="230.08" width="119" height="19.84" />
    </bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="Yaoqiang-Flow3_OK" bpmnElement="Flow3_OK">
    <di:waypoint x="342" y="217" />
    <di:waypoint x="423" y="217" />
    <di:waypoint x="423" y="160" />
    <di:waypoint x="504" y="160" />
    <bpmndi:BPMNLabel>
    <dc:Bounds x="417" y="182.08" width="114" height="19.84" />
    </bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="Yaoqiang-flow2" bpmnElement="flow2">
    <di:waypoint x="197.32352941176464" y="216" />
    <di:waypoint x="257.32352941176464" y="216" />
    <bpmndi:BPMNLabel>
    <dc:Bounds x="224.32" y="206.08" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="Yaoqiang-flow1" bpmnElement="flow1">
    <di:waypoint x="52.32285357569175" y="216" />
    <di:waypoint x="112.32352941176464" y="216" />
    <bpmndi:BPMNLabel>
    <dc:Bounds x="79.32" y="206.08" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
    </bpmndi:BPMNEdge>
    <bpmndi:BPMNEdge id="Flow_luee4uh_di" bpmnElement="Flow_luee4uh">
    <di:waypoint x="589" y="160" />
    <di:waypoint x="820" y="160" />
    </bpmndi:BPMNEdge>

```

```

<bpmndi:BPMNEdge id="Flow_065hvgf_di" bpmnElement="Flow_065hvgf">
  <di:waypoint x="902" y="264" />
  <di:waypoint x="950" y="264" />
  <di:waypoint x="950" y="217" />
  <di:waypoint x="1004" y="217" />
</bpmndi:BPMNEdge>
<bpmndi:BPMNEdge id="Flow_03tyub3_di" bpmnElement="Flow_03tyub3">
  <di:waypoint x="920" y="180" />
  <di:waypoint x="952" y="180" />
  <di:waypoint x="952" y="217" />
  <di:waypoint x="1004" y="217" />
</bpmndi:BPMNEdge>
<bpmndi:BPMNShape id="Yaoqiang-Event_Start" bpmnElement="Event_Start">
  <dc:Bounds x="20" y="200" width="32" height="32" />
  <bpmndi:BPMNLabel>
    <dc:Bounds x="21.5" y="240.08" width="29" height="19.84" />
  </bpmndi:BPMNLabel>
</bpmndi:BPMNShape>
      <bpmndi:BPMNShape      id="Yaoqiang-DATA_OWNER_NOTIFICATION"
bpmnElement="DATA_OWNER_NOTIFICATION">
  <dc:Bounds x="112" y="188.5" width="85" height="55" />
  <bpmndi:BPMNLabel>
    <dc:Bounds x="125.5" y="200.58" width="58" height="34.84" />
  </bpmndi:BPMNLabel>
</bpmndi:BPMNShape>
      <bpmndi:BPMNShape      id="Yaoqiang-DATA_OWNER_VALIDATOR"
bpmnElement="DATA_OWNER_VALIDATOR">
  <dc:Bounds x="257" y="188.5" width="85" height="55" />
  <bpmndi:BPMNLabel>
    <dc:Bounds x="268" y="200.58" width="63" height="34.84" />
  </bpmndi:BPMNLabel>
</bpmndi:BPMNShape>
      <bpmndi:BPMNShape      id="Yaoqiang-FINAL_REJECT_EXEC"
bpmnElement="FINAL_REJECT_EXEC">
  <dc:Bounds x="504" y="236" width="85" height="55" />
  <bpmndi:BPMNLabel>
    <dc:Bounds x="673.5" y="255.58" width="36" height="19.84" />
  </bpmndi:BPMNLabel>
</bpmndi:BPMNShape>
<bpmndi:BPMNShape id="Yaoqiang-CANCEL_EXEC" bpmnElement="CANCEL_EXEC">
  <dc:Bounds x="657" y="236" width="85" height="55" />
  <bpmndi:BPMNLabel>
    <dc:Bounds x="817" y="255.58" width="39" height="19.84" />
  </bpmndi:BPMNLabel>
</bpmndi:BPMNShape>
<bpmndi:BPMNShape id="Yaoqiang-FINAL_EXEC" bpmnElement="FINAL_EXEC">
  <dc:Bounds x="504" y="132" width="85" height="55" />
  <bpmndi:BPMNLabel>
    <dc:Bounds x="679.5" y="160.58" width="24" height="19.84" />
  </bpmndi:BPMNLabel>
</bpmndi:BPMNShape>
      <bpmndi:BPMNShape      id="Yaoqiang-END_OK_NOTIFICATION"
bpmnElement="END_KO_NOTIFICATION">

```

```

    <dc:Bounds x="817" y="236" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="517.5" y="153.08" width="58" height="34.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
  <bpmndi:BPMNShape id="Activity_082ilpu_di" bpmnElement="END_OK_NOTIFICATION">
    <dc:Bounds x="820" y="140" width="100" height="80" />
  </bpmndi:BPMNShape>
  <bpmndi:BPMNShape id="Yaoqiang-Event_End_KO" bpmnElement="Event_End">
    <dc:Bounds x="1004" y="201" width="32" height="32" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="1010" y="240" width="20" height="14" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
</bpmndi:BPMNPlane>
</bpmndi:BPMNDiagram>
</definitions>

```

Elementos del XML

- Secuencia y tareas. Se pueden identificar los siguientes elementos en xml:
 - Tareas que se ejecutan de forma automática por el sistema, identificadas dentro del xml con las etiquetas *serviceTask*.
 - Tareas que se ejecutan mediante la interacción humana. Estas tareas están representadas dentro de las etiquetas *userTask*.
 - Definición de secuencias de ejecución, identificadas con las etiquetas *sequenceFlow*.

```

    <process id="ADHERENCE" name="Request of adherence" processType="None"
isClosed="false" isExecutable="true">
    <extensionElements>
      <yaoqiang:description />
      <yaoqiang:pageFormat height="841.8897637795276"
imageableHeight="831.8897637795276" imageableWidth="588.1102362204724"
imageableX="5.0" imageableY="5.0" orientation="0" width="598.1102362204724" />
      <yaoqiang:page background="#FFFFFF" horizontalCount="1" verticalCount="1" />
    </extensionElements>
    <startEvent id="Event_Start" name="Start">
      <outgoing>flow1</outgoing>
      <outputSet />
    </startEvent>
    <serviceTask id="DATA_OWNER_NOTIFICATION" name="Owner notification"
implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJavaDelegate">
      <extensionElements>
        <activiti:field name="notificationCode" stringValue="20" />
      </extensionElements>
      <incoming>flow1</incoming>
      <outgoing>flow2</outgoing>
    </serviceTask>
  </process>

```



```

</serviceTask>
    <userTask id="DATA_OWNER_VALIDATOR" name="Data owner validation"
implementation="##unspecified">
    <extensionElements>
        <activiti:field name="validatorRole" stringValue="data_owner" />
    </extensionElements>
    <incoming>flow2</incoming>
    <outgoing>Flow3_OK</outgoing>
    <outgoing>Flow3_KO</outgoing>
    <multiInstanceLoopCharacteristics
activiti:collection="MODULE_ORGANIZATIONAL_UNIT">
        <completionCondition xsi:type="tFormalExpression">${ACCEPTED !=
'true'}</completionCondition>
    </multiInstanceLoopCharacteristics>
</userTask>
    <serviceTask id="END_KO_NOTIFICATION" name="Event KO notification"
implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJavaDelegate">
    <extensionElements>
        <activiti:field name="notificationCode" stringValue="11" />
    </extensionElements>
    <incoming>CANCEL</incoming>
    <outgoing>Flow_065hvgf</outgoing>
</serviceTask>
    <serviceTask id="FINAL_EXEC" name="End" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.LastExecutionJavaDelegate">
    <incoming>Flow3_OK</incoming>
    <outgoing>Flow_luee4uh</outgoing>
</serviceTask>
    <serviceTask id="FINAL_REJECT_EXEC" name="Reject" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.LastExecutionJavaDelegate">
    <incoming>Flow3_KO</incoming>
    <outgoing>KO</outgoing>
</serviceTask>
    <serviceTask id="CANCEL_EXEC" name="Cancel" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.CancelTaskWorkflowDelegate">
    <incoming>KO</incoming>
    <outgoing>CANCEL</outgoing>
</serviceTask>
<endEvent id="Event_End" name="End">
    <incoming>Flow_065hvgf</incoming>
    <incoming>Flow_03tyub3</incoming>
    <inputSet />
</endEvent>
    <sequenceFlow id="flow1" sourceRef="Event_Start"
targetRef="DATA_OWNER_NOTIFICATION" />
    <sequenceFlow id="flow2" sourceRef="DATA_OWNER_NOTIFICATION"
targetRef="DATA_OWNER_VALIDATOR" />
    <sequenceFlow id="Flow3_OK" sourceRef="DATA_OWNER_VALIDATOR"
targetRef="FINAL_EXEC">
        <conditionExpression xsi:type="tFormalExpression">${ACCEPTED ==
'true'}</conditionExpression>
    </sequenceFlow>

```

```

        <sequenceFlow id="Flow3_KO" sourceRef="DATA_OWNER_VALIDATOR"
targetRef="FINAL_REJECT_EXEC">
        <conditionExpression xsi:type="tFormalExpression">${ACCEPTED ==
'false'}</conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="KO" sourceRef="FINAL_REJECT_EXEC" targetRef="CANCEL_EXEC" />
    <sequenceFlow id="CANCEL" sourceRef="CANCEL_EXEC" targetRef="END_KO_NOTIFICATION"
/>
        <sequenceFlow id="Flow_1uee4uh" sourceRef="FINAL_EXEC"
targetRef="END_OK_NOTIFICATION" />
        <sequenceFlow id="Flow_065hvgf" sourceRef="END_KO_NOTIFICATION"
targetRef="Event_End" />
        <serviceTask id="END_OK_NOTIFICATION" name="Event OK notification"
activiti:class="com.anjana.hermes.functionalities.SentNotificationJavaDelegate">
    <extensionElements>
        <activiti:field name="notificationCode" stringValue="10" />
    </extensionElements>
    <incoming>Flow_1uee4uh</incoming>
    <outgoing>Flow_03tyub3</outgoing>
</serviceTask>
        <sequenceFlow id="Flow_03tyub3" sourceRef="END_OK_NOTIFICATION"
targetRef="Event_End" />
</process>

```

Entre las distintas tareas que se pueden definir en este momento hay implementadas las siguientes:

- **CancelTaskWorkflowDelegate.** Esta tarea cancela el workflow indicado. Puede aceptar los siguientes parámetros:
 - WORKFLOW_EXECUTION_ID (entrada). Workflow sobre el que trabajar.
 - ACCEPTED (salida). Establece con valor “false” esta variable.

- **LastExecutionJavaDelegate.** Llama al endpoint que se encarga de realizar la última ejecución del workflow dependiendo del objeto y tipo de objeto afectado.
 - WORKFLOW_EXECUTION_ID (entrada). Workflow sobre el que trabajar.
 - OBJECT_ID (entrada). Identificador del objeto afectado por el workflow.
 - OBJECT_SUB_TYPE (entrada). Tipo del objeto afectado por el workflow.

- **SentNotificationJavaDelegate.** Esta tarea se encarga de enviar notificaciones según la tarea de la que se trata y el estado de la misma. Los parámetros que utiliza son:
 - WORKFLOW_EXECUTION_ID (entrada). Workflow sobre el que trabajar.
 - ACCEPTED (salida). Establece con valor “false” esta variable.
 - notificationCode (entrada). Código de la notificación que se quiere enviar.

Otra características de estas tareas es que se indica que secuencia del workflow es la que lo activa y cual es la siguiente secuencia a ejecutar. Esto se indica con las etiquetas *incoming* y *outcoming*.

```

<serviceTask id="FINAL_EXEC" name="End" implementation="##WebService"
activiti:class="com.anjana.hermes.functionalities.LastExecutionJavaDelegate">
    <incoming>Flow3_OK</incoming>
    <outgoing>Flow_1uee4uh</outgoing>

```

```
</serviceTask>
```

En la definición de las secuencias del workflow se puede especificar una condición que cuando sea verdadera se ejecutará esa secuencia. Esta característica junto con la anterior de las tareas donde se especifica el siguiente paso a ejecutar nos permitirá poder definir las bifurcaciones dentro del workflow.

```
<sequenceFlow id="Flow3_OK" sourceRef="DATA_OWNER_VALIDATOR" targetRef="FINAL_EXEC">  
  <conditionExpression xsi:type="tFormalExpression">${ACCEPTED ==  
'true'}</conditionExpression>  
</sequenceFlow>
```

Las tareas de usuario se utilizan para poder tener la validación previa de una determinada tarea automática si así fuera necesario. Los elementos de esta tarea son los siguientes:

```
<userTask id="DATA_OWNER_VALIDATOR" name="Data Owner validation" implementation="##unspecified">  
  <extensionElements>  
    <activiti:field name="validatorRole" stringValue="data_owner" />  
  </extensionElements>  
  <incoming>flow2</incoming>  
  <outgoing>Flow2_OK</outgoing>  
  <outgoing>Flow2_KO</outgoing>  
  <multiInstanceLoopCharacteristics activiti:collection="MODULE_ORGANIZATIONAL_UNIT">  
    <completionCondition xsi:type="tFormalExpression">${ACCEPTED != 'true'}</completionCondition>  
  </multiInstanceLoopCharacteristics>  
</userTask>
```

- activity:field name="validatorRole". Indica el role de quien tiene que realizar la validación.
- incoming. Indica la secuencia del workflow que origina esta tarea.
- outgoing. Indica que secuencia del workflow se generará según la respuesta dada. En el ejemplo definimos dos posibilidades que se corresponden con una validación correcta o incorrecta.
- completionCondition. Condición que se tiene que cumplir para considerar la tarea como finalizada.

La forma de orquestar las distintas tareas, tanto automáticas como no automáticas se realiza mediante las secuencias del workflow explicada más arriba.

- Definición de estructura gráfica. Recordamos que esta definición se establece abriendo y guardando el xml mediante la aplicación [Yaoqiang BPMN Editor](#).

```
<bpmndi:BPMNDiagram id="Yaoqiang_Diagram-ADHERENCE_-_ARCHITECT" name="Untitled  
Diagram" resolution="96">  
  <bpmndi:BPMNPlane bpmnElement="ADHERENCE">  
    <bpmndi:BPMNEdge id="Yaoqiang-CANCEL" bpmnElement="CANCEL">  
      <di:waypoint x="742" y="264" />  
      <di:waypoint x="817" y="264" />  
      <bpmndi:BPMNLabel>  
        <dc:Bounds x="906.33" y="253.58" width="6" height="19.84" />  
      </bpmndi:BPMNLabel>  
    </bpmndi:BPMNEdge>  
    <bpmndi:BPMNEdge id="Yaoqiang-KO" bpmnElement="KO">  
      <di:waypoint x="589" y="264" />  
      <di:waypoint x="657" y="264" />
```

```

    <bpmndi:BPMNLabel>
      <dc:Bounds x="761.32" y="253.58" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Yaoqiang-Flow3_KO" bpmnElement="Flow3_KO">
    <di:waypoint x="342" y="217" />
    <di:waypoint x="423" y="217" />
    <di:waypoint x="423" y="280" />
    <di:waypoint x="504" y="280" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="414.5" y="230.08" width="119" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Yaoqiang-Flow3_OK" bpmnElement="Flow3_OK">
    <di:waypoint x="342" y="217" />
    <di:waypoint x="423" y="217" />
    <di:waypoint x="423" y="160" />
    <di:waypoint x="504" y="160" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="417" y="182.08" width="114" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Yaoqiang-flow2" bpmnElement="flow2">
    <di:waypoint x="197.32352941176464" y="216" />
    <di:waypoint x="257.32352941176464" y="216" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="224.32" y="206.08" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Yaoqiang-flow1" bpmnElement="flow1">
    <di:waypoint x="52.32285357569175" y="216" />
    <di:waypoint x="112.32352941176464" y="216" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="79.32" y="206.08" width="6" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Flow_luee4uh_di" bpmnElement="Flow_luee4uh">
    <di:waypoint x="589" y="160" />
    <di:waypoint x="820" y="160" />
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Flow_065hvgf_di" bpmnElement="Flow_065hvgf">
    <di:waypoint x="902" y="264" />
    <di:waypoint x="950" y="264" />
    <di:waypoint x="950" y="217" />
    <di:waypoint x="1004" y="217" />
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNEdge id="Flow_03tyub3_di" bpmnElement="Flow_03tyub3">
    <di:waypoint x="920" y="180" />
    <di:waypoint x="952" y="180" />
    <di:waypoint x="952" y="217" />
    <di:waypoint x="1004" y="217" />
  </bpmndi:BPMNEdge>
  <bpmndi:BPMNShape id="Yaoqiang-Event_Start" bpmnElement="Event_Start">

```

```

    <dc:Bounds x="20" y="200" width="32" height="32" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="21.5" y="240.08" width="29" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
    <bpmndi:BPMNShape id="Yaoqiang-DATA_OWNER_NOTIFICATION"
bpmnElement="DATA_OWNER_NOTIFICATION">
    <dc:Bounds x="112" y="188.5" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="125.5" y="200.58" width="58" height="34.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
    <bpmndi:BPMNShape id="Yaoqiang-DATA_OWNER_VALIDATOR"
bpmnElement="DATA_OWNER_VALIDATOR">
    <dc:Bounds x="257" y="188.5" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="268" y="200.58" width="63" height="34.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
    <bpmndi:BPMNShape id="Yaoqiang-FINAL_REJECT_EXEC"
bpmnElement="FINAL_REJECT_EXEC">
    <dc:Bounds x="504" y="236" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="673.5" y="255.58" width="36" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
  <bpmndi:BPMNShape id="Yaoqiang-CANCEL_EXEC" bpmnElement="CANCEL_EXEC">
    <dc:Bounds x="657" y="236" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="817" y="255.58" width="39" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
  <bpmndi:BPMNShape id="Yaoqiang-FINAL_EXEC" bpmnElement="FINAL_EXEC">
    <dc:Bounds x="504" y="132" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="679.5" y="160.58" width="24" height="19.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
    <bpmndi:BPMNShape id="Yaoqiang-END_OK_NOTIFICATION"
bpmnElement="END_KO_NOTIFICATION">
    <dc:Bounds x="817" y="236" width="85" height="55" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="517.5" y="153.08" width="58" height="34.84" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNShape>
  <bpmndi:BPMNShape id="Activity_082ilpu_di" bpmnElement="END_OK_NOTIFICATION">
    <dc:Bounds x="820" y="140" width="100" height="80" />
  </bpmndi:BPMNShape>
  <bpmndi:BPMNShape id="Yaoqiang-Event_End_KO" bpmnElement="Event_End">
    <dc:Bounds x="1004" y="201" width="32" height="32" />
    <bpmndi:BPMNLabel>
      <dc:Bounds x="1010" y="240" width="20" height="14" />
    </bpmndi:BPMNLabel>
  </bpmndi:BPMNLabel>

```

```
</bpmndi:BPMNShape>  
</bpmndi:BPMNPlane>  
</bpmndi:BPMNDiagram>
```