Laboratorio 7

Contenidos

Objetivos

En este laboratorio modificaremos nuestro modelo de proceso `XXX_Add_Vehicle`, incluyéndole una llamada a otro proceso (`XXX_Review_Vehicle`) y una puerta de decisión en función de los datos recibidos desde el proceso llamado.

Si no pudimos completar en el laboratorio anterior nuestro proceso `XXX_Review_Vehicle`, podemos hacer uso del ya existente en la aplicación con la solución (`Open_Vehicle Fleet Management`). En este caso, el modelo de proceso a llamar en sería `Open_Review_Vehicle`.

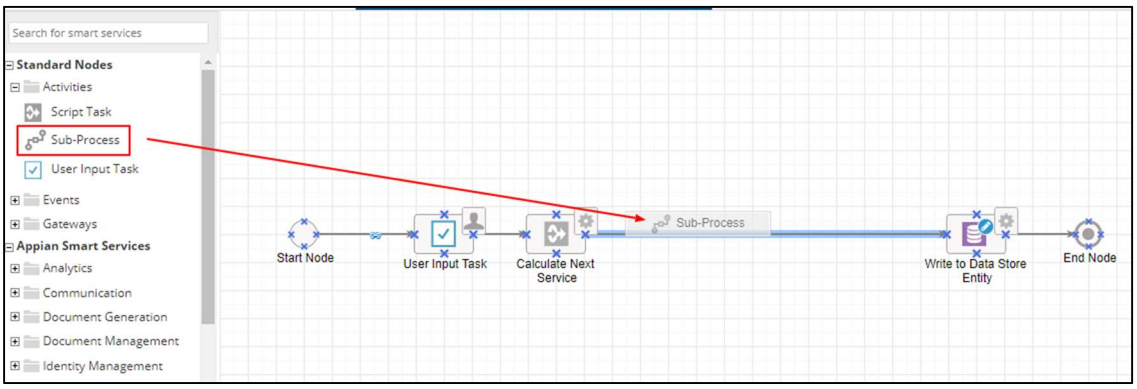
Inclusión de Subproceso

Descripción:

A nuestro proceso `XXX_Add_Vehicle` le añadiremos una llamada al proceso `XXX_Review_Vehicle`.

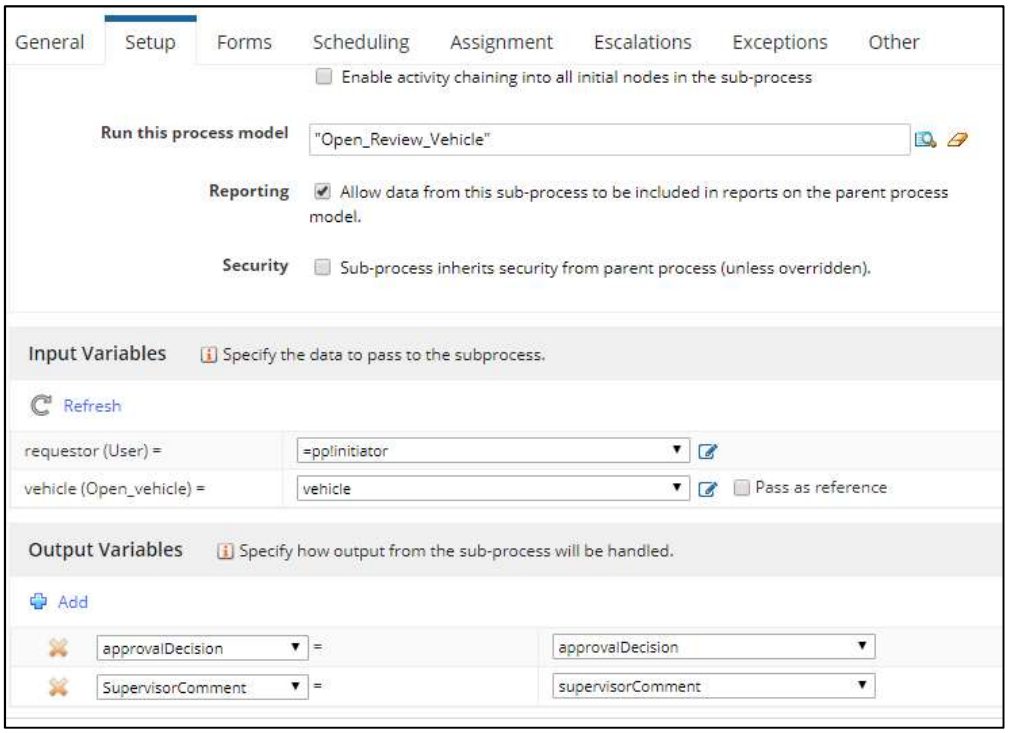
Pasos a realizar:

- Dentro de nuestra aplicación `XXX_Vehicle Fleet Management`, en la carpeta `Process` creada con anterioridad, localizamos nuestro proceso `XXX_Add_Vehicle`, y pulsamos sobre él. Esto nos abrirá el entorno Appian Process Modeler, a través del cual modificaremos nuestro modelo de proceso.
- En el menú de la izquierda, localizar el elemento `Sub-Process` (`Standard Nodes – Activities – Sub-Process`) y lo arrastramos sobre la línea que conecta el script "`Calculate Next Service`" y "`Write To Data Store Entity`", tal y como se muestra en la imagen siguiente:



- Pulsamos con el botón derecho - `Properties` sobre este elemento `Sub-Process` recién agregado. Completaremos algunas de las pestañas que nos aparecen.
- Pestaña General
    - **Name**: `Review Vehicle`
    - **Task Display Name**: `Review Vehicle`
- Pestaña Setup

- **Run this process model**: `XXX_Review_Vehicle`
- Completamos los valores para las variables de entrada y las de salida, para que el resultado quede como sigue:



- Cuando definimos nuestro modelo de proceso, le indicamos 2 parámetros de entrada: `requestor` y `vehicle`. Por tanto, al invocarlo, debemos asignar valores a dichos parámetros de entrada.
- Definimos unos parámetros de salida para recoger los datos que nos devuelva en el modelo de proceso invocado: `approvalDecision` y `supervisorComments`.
- Pulsamos OK y cerramos la ventana emergente.

Creación de Rule Expression

Definición:

Una regla de expresión es una expresión almacenada que se puede llamar desde otras expresiones. Al igual que las funciones de Appian, las reglas de expresión siempre devuelven un valor que puede estar influenciado por una o más entradas. as reglas de expresión pueden invocarse desde cualquier expresión, por lo que pueden reutilizarse en múltiples objetos en todo el sistema.

Descripción:

Crear una Rule Expression que nos indique si un vehículo debe ser denegado o no. Tendrá como parámetros de entrada: `counter` (`Number:integer`) y `decision` (`text`). Tanto si el parámetro `counter` es mayor o igual a `2`, como si el parámetro decision es `Reject`, en ambos casos, nuestra ER devolverá `true`. En otro caso, debe devolver `false`.

Pasos a realizar:

- Situarnos dentro de `XXX_Vehicle Fleet Management` (carpeta `XXX_Rules_And_Constants`).
- Pulsamos `New - Expression Rule`
    - **Name**: `XXX_IsVehicleDenied`
    - **Description**: ER que devolverá un booleano con valor `true` si un vehículo debe ser denegado o `false` en caso contrario
    - **Save in**: `XXX_Rules_And_Constants`
- Agregarle 2 rule inputs:
    - `Counter` (`Number: integer`)
    - `Decision` (`text`)
- Completar nuestra regla para que cumpla el objetivo indicado en la descripción.
- Validar la regla, apoyándonos con el panel central, dando valores a las pruebas inputs.
- Validaremos nuestra ER con 3 sencillas pruebas:
    - `counter=1, decisión="More Info" => salida=false`
    - `counter=2, decisión="More Info" => salida=true`
    - `counter=1, decisión="Reject" => salida=true`
- Una vez validado, pulsamos en botón `Save`.

Nota: si no podemos completar correctamente nuestra Expression Rule, podemos basarnos en la ya existente en la solución `Open_Fleet_Management`. Concretamente, la ER se llama `Open_IsVehicleDenied`.

Inclusión de Puerta de decisión

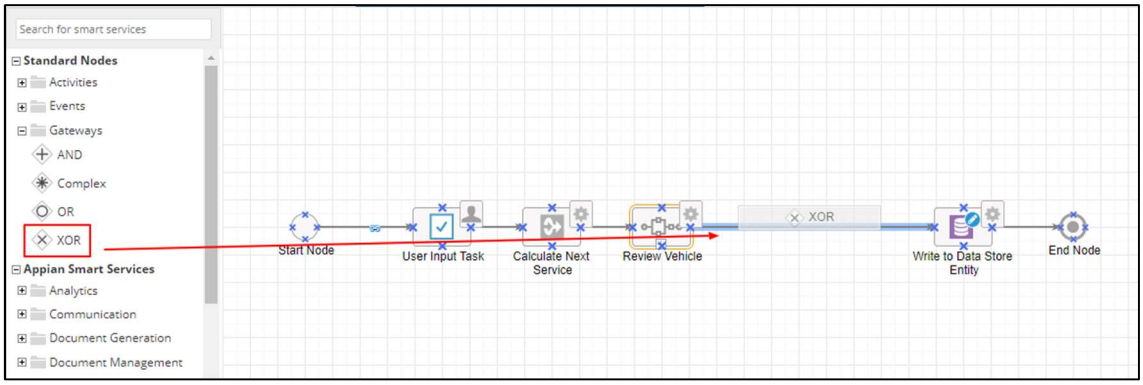
Descripción:

Incluiremos en nuestro flujo una puerta lógica, para enrutar hacia un camino u otro según la decisión tomada por el supervisor. Básicamente, tendremos 3 posibles opciones:

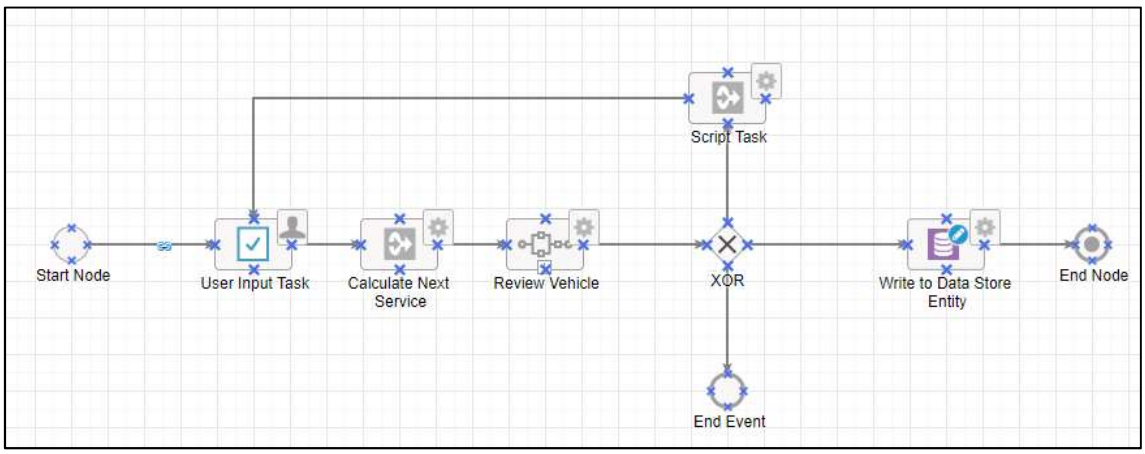
- El supervisor aprueba el vehículo. En tal caso, persistiremos el vehículo en la base de datos.
- El supervisor solicita más información sobre el vehículo, y esta solicitud todavía no la ha hecho más de 3 veces. En tal caso, redirigiremos nuevamente al interfaz de alta.
- El supervisor rechaza el vehículo, o bien ha solicitado más de 3 veces información de este. En tal caso, el vehículo será rechazado.

Pasos a realizar:

- En el menú de la izquierda, localizar el elemento `XOR` (`Standard Nodes – Gateways – XOR`) y lo arrastramos sobre la línea que conecta el script "`Review Vehicle`" y "`Write To Data Store Entity`" tal y como se muestra en la siguiente imagen:

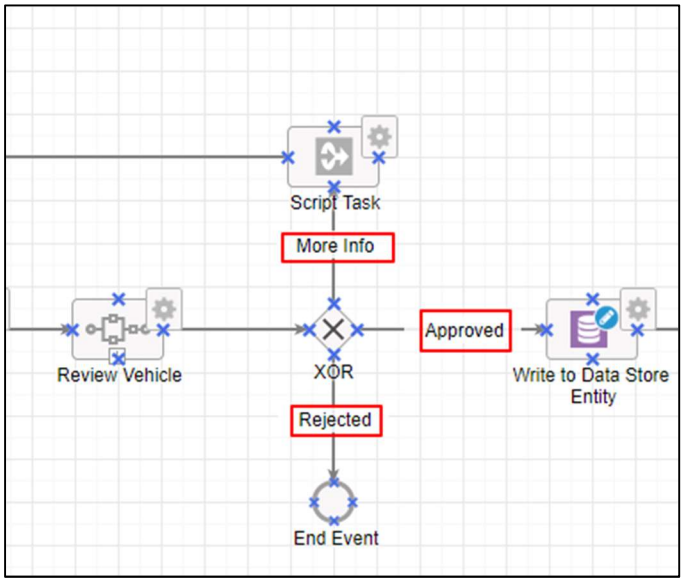


- En el menú de la izquierda, localizar el elemento `Script Task` (`Standard Nodes – Activities – Script Task`) y lo colocamos en nuestro panel central (de momento quedará sin conectar con ningún otro elemento).
- En el menú de la izquierda, localizar el elemento `End Event` (`Standard Nodes – Events – End Event`) y lo colocamos en nuestro panel central (de momento quedará sin conectar con ningún otro elemento). Este elemento `end Event` lo usaremos para ofrecer otra forma de finalizar el proceso, además del ya existente.
- Conectar los elementos recién incorporados para que queden como sigue.
- Para añadir nuevas líneas, buscar en el menú superior el siguiente icono:  .
- Una vez, agregadas las líneas podemos volver a seleccionar componentes pulsando el icono  situado en el menú superior.

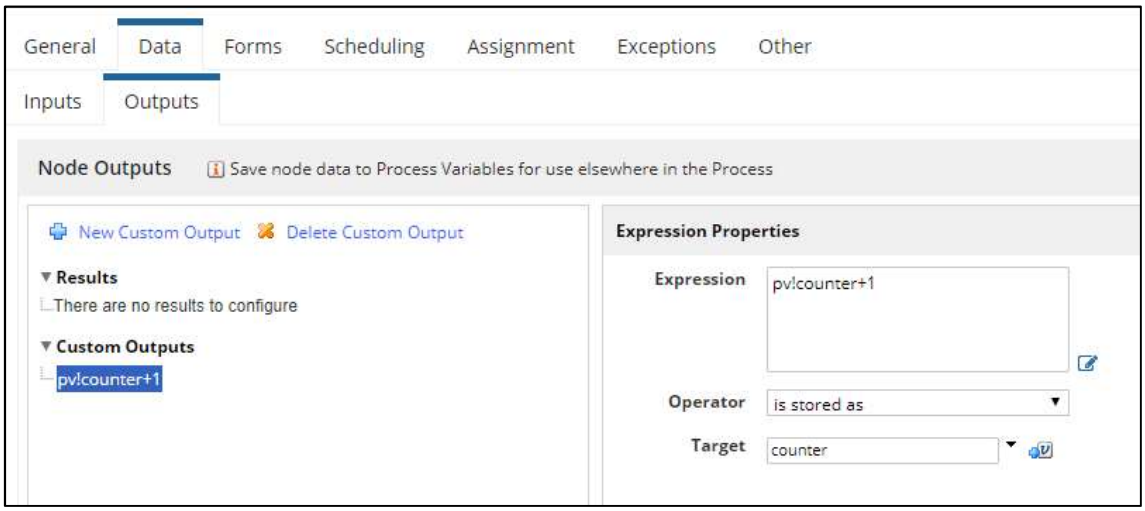


- Renombrar las líneas que salen del componente `XOR`. Para esto, pulsar directamente sobre cada una de las líneas y completar el campo `label`.

- El resultado debe ser el siguiente:



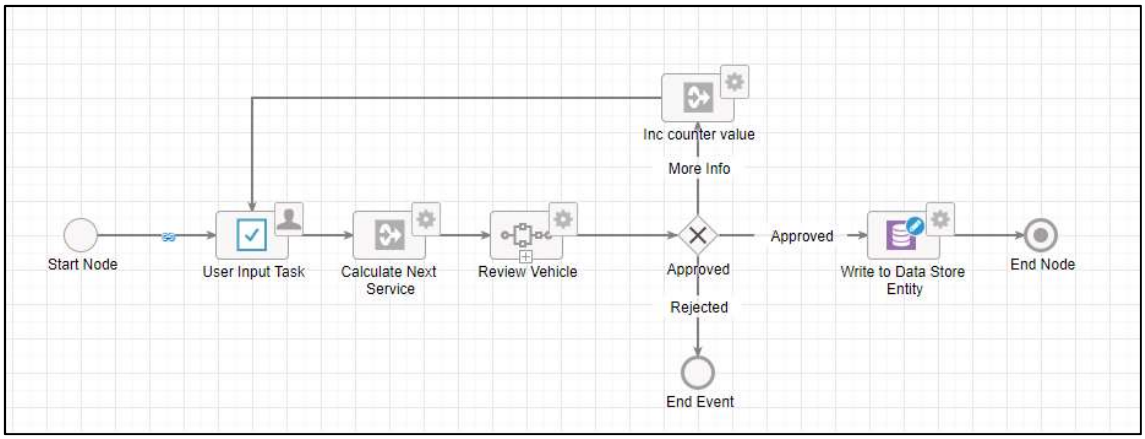
- Pulsando el botón derecho – `Properties` sobre el elemento añadido `Script Task`. Este `scriptTask` lo usaremos para incrementar el valor de nuestro contador (`counter`). Así sabremos el número de veces que se ha solicitado más información sobre el vehículo.
- Pestaña General
    - **Name**: `Inc Counter Value`
    - **Task Display Name**: `Inc Counter Value`
- Pestaña Data
    - Añadir un nuevo nodo output con los siguientes valores:



Este script task, lo que hace es incrementar en 1 el valor de nuestra variable de proceso (pv) `counter`.

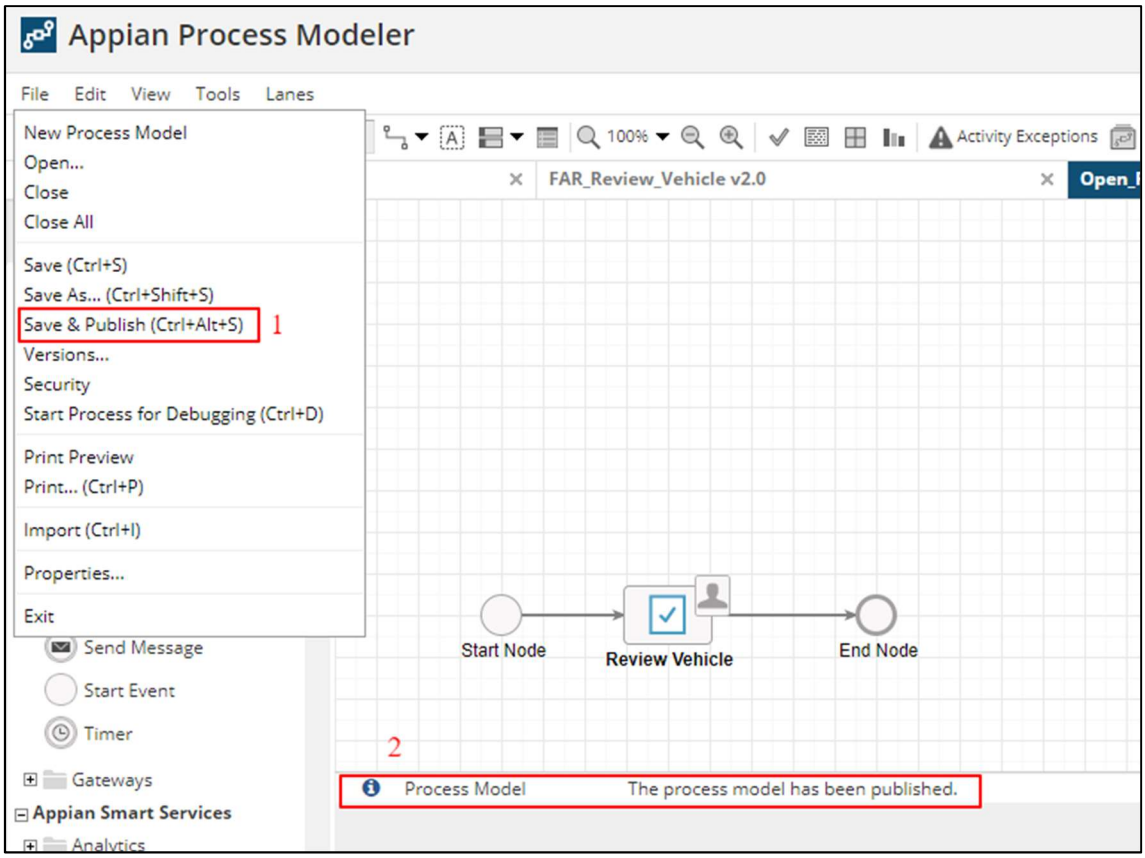
- Pulsando el botón derecho – `Properties` sobre el elemento `XOR` recién añadido.
- Pestaña General
    - Name: Approved
    - Task Display Name: Approved
- Pestaña Decision
    - Pulsamos el botón `New Condition`.
        - **Condition**: `=pv!approvalDecision="Approved"`.
        - **Go to**: `Write To Data Store Entity`.
        - El flujo continuará por esta rama en caso de que el supervisor haya tomado la opción de aprobar el vehículo. Solo en ese caso, lo persistimos en la base de datos.
    - Pulsamos el botón `New Condition`.
        - **Condition**: `=rule!XXX_IsVehicleDenied(counter:pv!counter , decision:pv!approvalDecision)`.
        - **Go to**: `End Event`.
        - Para configurar esta rama nos apoyamos en una Expression Rule ya existente en el entorno. Esta regla devuelve un booleano: será `true`, si el contador es `>= 2` o si la decisión del supervisor es distinta a `Approved`.
        - Si se cumple la regla, el flujo continuará hacia el nodo "`End Event`".
        - `Else If non are true, go to: Inc counter value`
        - Si no se cumple ninguna de las condiciones anteriores, el flujo continuará hacia el script "`Inc counter value`".
    - El resultado de la pestaña `Decision` será el siguiente:

El resultado de nuestro proceso debe ser como sigue:



- Nos vamos al menú superior – `File – Save and Publish`.
- Una vez terminada la publicación, nos debe aparecer en la parte inferior del panel central un mensaje informativo indicándonos que el process model se ha publicado con éxito.

Cada vez que realicemos alguna modificación sobre un proceso, debemos volver a publicarlo para que sus cambios sean visibles en las próximas pruebas.



Conclusiones

En este laboratorio hemos incluido una llamada a un proceso dentro de otro proceso. También hemos visto como incluir una puerta de decisión, la cual, en función de los criterios que hemos establecido, hará que el flujo del proceso tome un camino u otro.