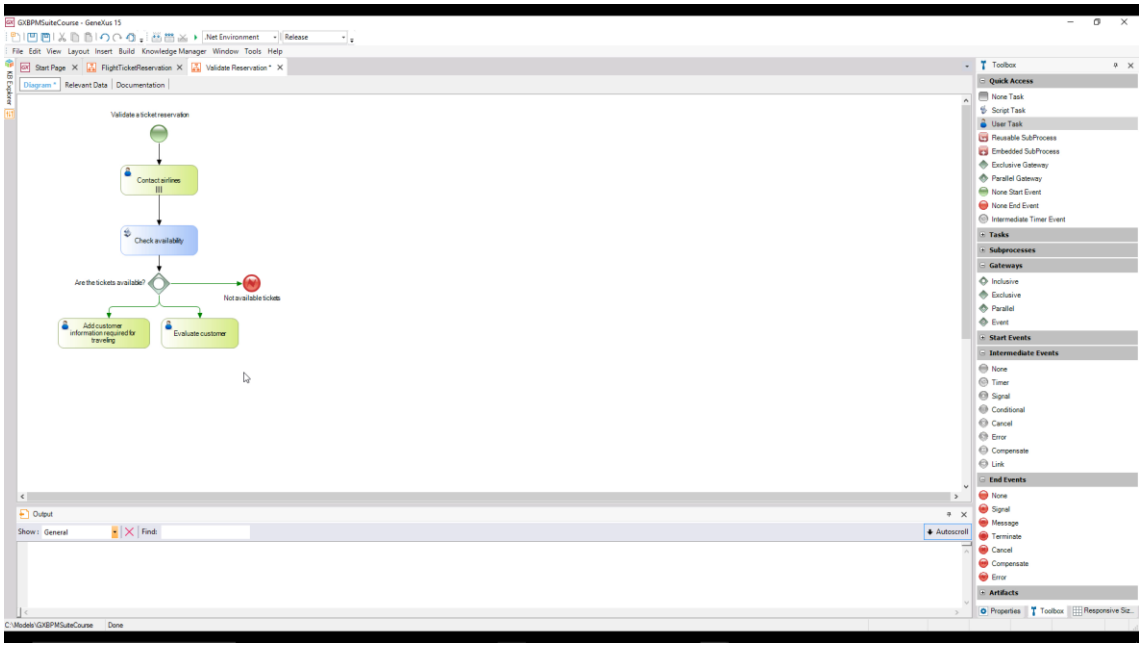
Bifurcación y unión de caminos, generación de avisos periódicos y manejo de señales

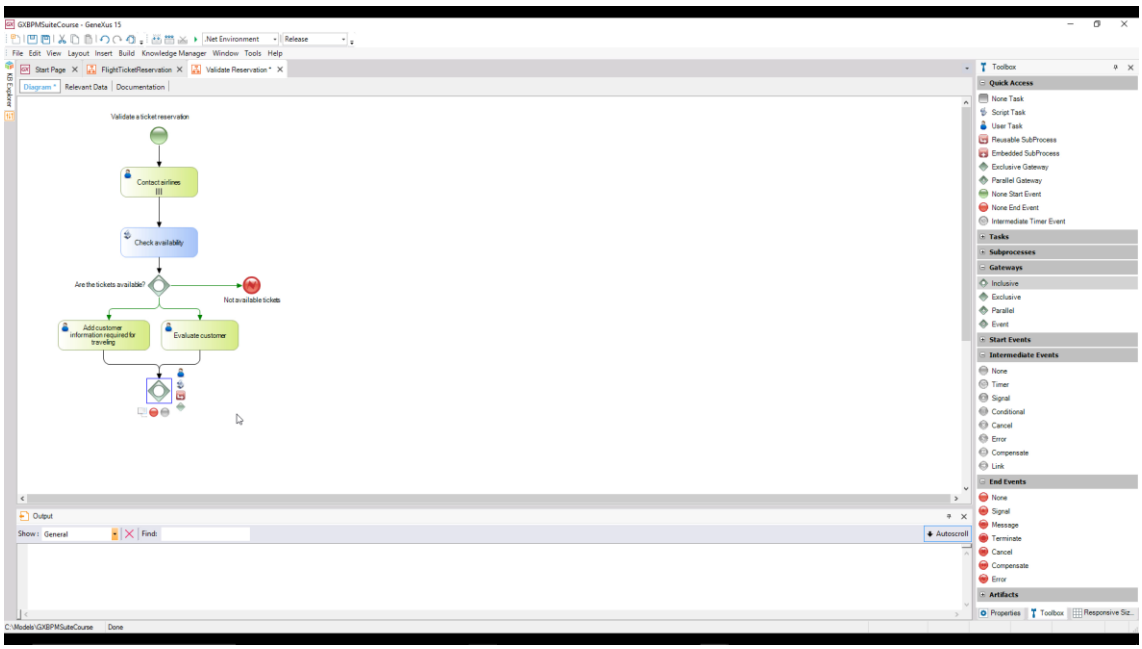
Volviendo al modelo del proceso de validación de la reserva, si había disponibilidad de tickets debíamos continuar por 2 caminos simultáneamente. Agregamos las tareas interactivas “Add customer information required for traveling” y “Evaluate Customer”, conectando ambas desde el inclusive gateway.



En este caso, estamos utilizando el inclusive gateway para ramificar caminos. Cada camino tiene definido una condición y el proceso seguirá por uno o más caminos a la vez, dependiendo de la cuáles sean las condiciones que se cumplan.

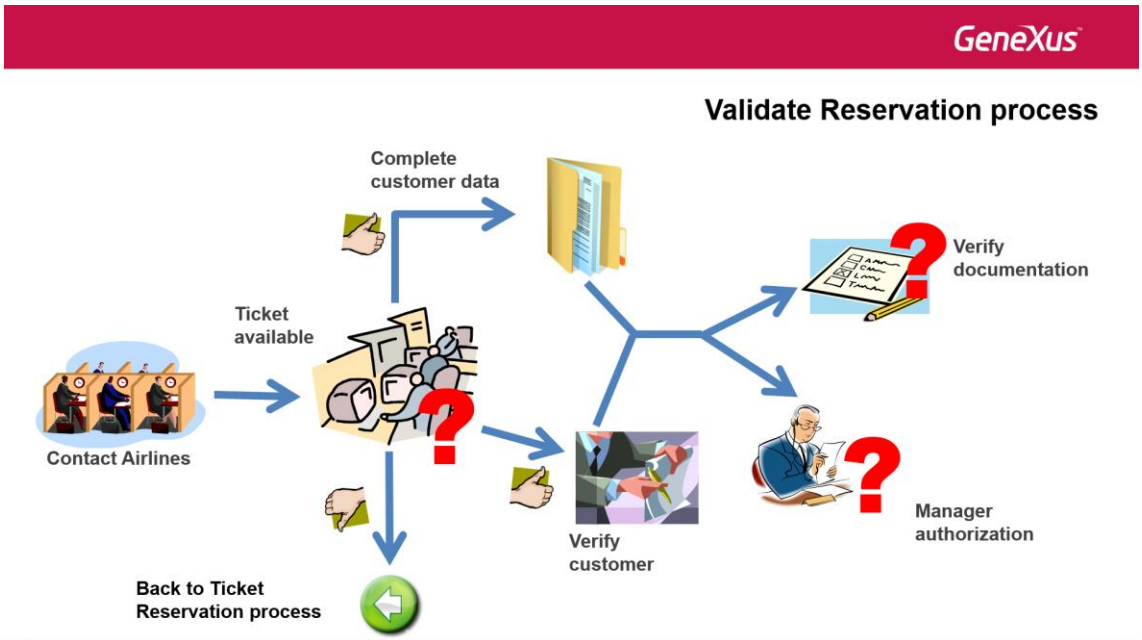
Una vez terminadas las tareas de obtener información para el viaje y verificar situación financiera del cliente, se debe continuar con el proceso, pero para poder continuar, ambas tareas deben estar finalizadas.

Es decir que si una tarea termina antes que la otra, se deberá esperar a que la segunda termine para continuar el proceso. Para conseguir eso, usamos también un inclusive gateway, así que lo arrastramos desde la barra de herramientas y lo conectamos desde ambas tareas.

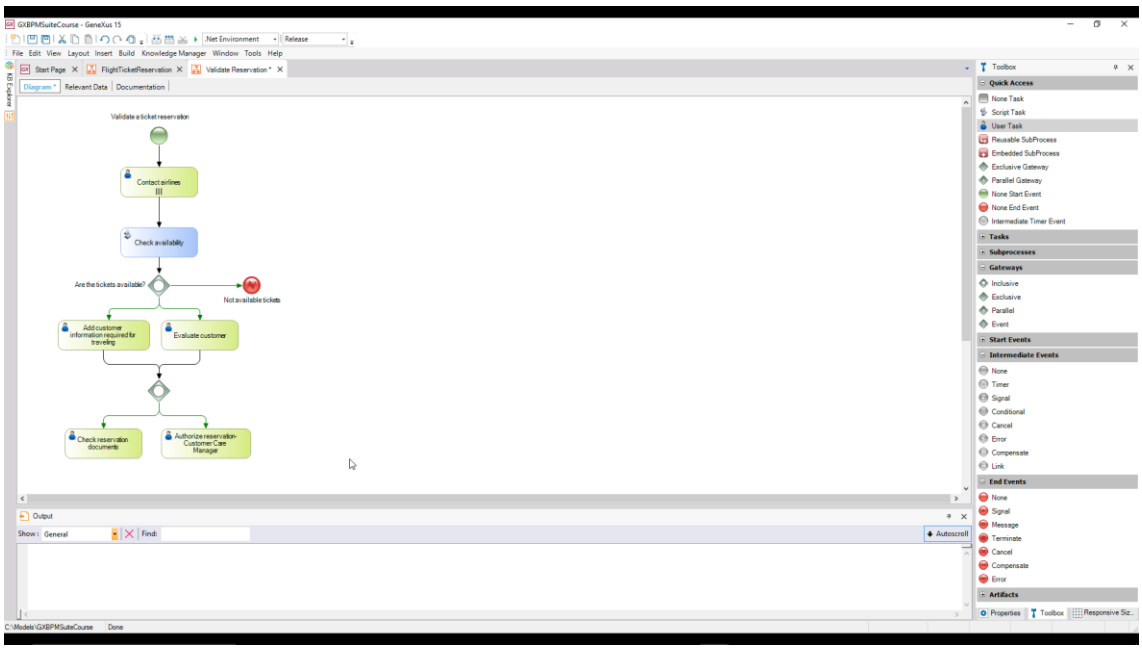


En la unificación de caminos, el inclusive gateway realiza una sincronización de los caminos por donde realmente avanzó el proceso, que no son necesariamente todos los caminos que llegan al gateway.

En nuestro caso, el proceso puede avanzar a partir de cualquiera de las 2 tareas y son los únicos caminos que llegan al inclusive gateway, así que una vez que ambas finalizan, se continuará con nuestro proceso que también implica la ejecución de 2 actividades: la verificación de los datos necesarios para el viaje y la autorización del gerente de atención al cliente.



Para modelar la divergencia del flujo en 2 caminos, utilizaremos un inclusive gateway como en el caso anterior, así que aprovechamos el mismo inclusive gateway que habíamos usado para unir los caminos y creamos las tareas Check reservation documents y Authorize Reservation -Customer Care Manager, conectándolas desde dicho nodo.

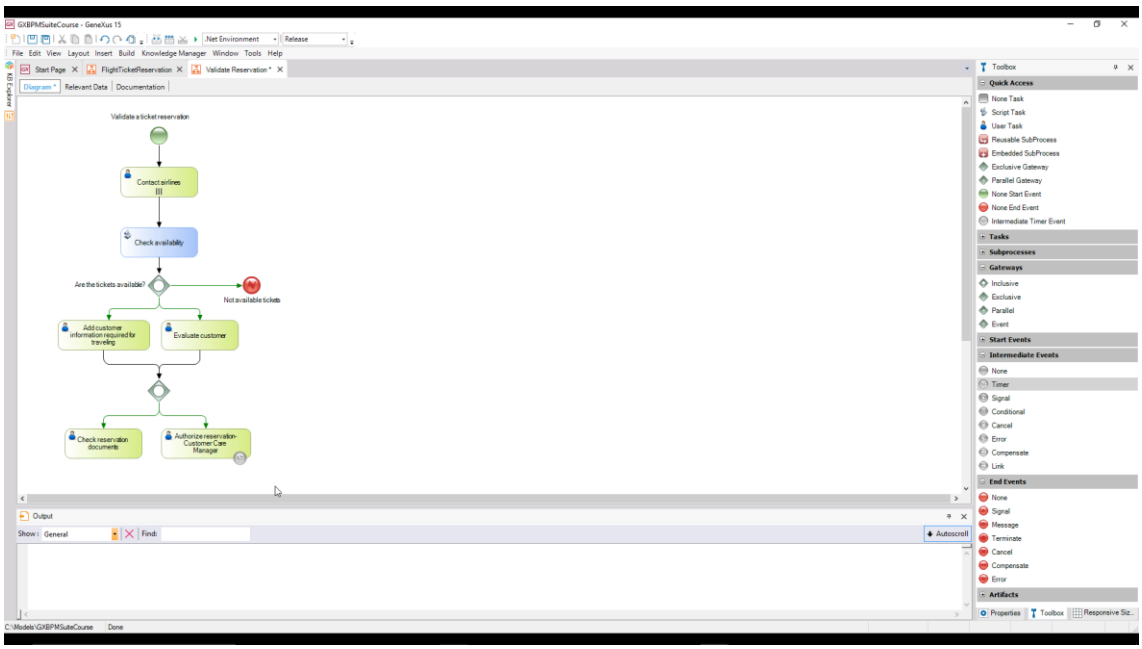


Ahora bien…La tarea que ejecuta el gerente de atención al cliente debe tener un tiempo límite y queremos que el sistema pueda avisarle al mismo, en forma periódica y repetitiva, que tiene una reserva pendiente de autorizar.

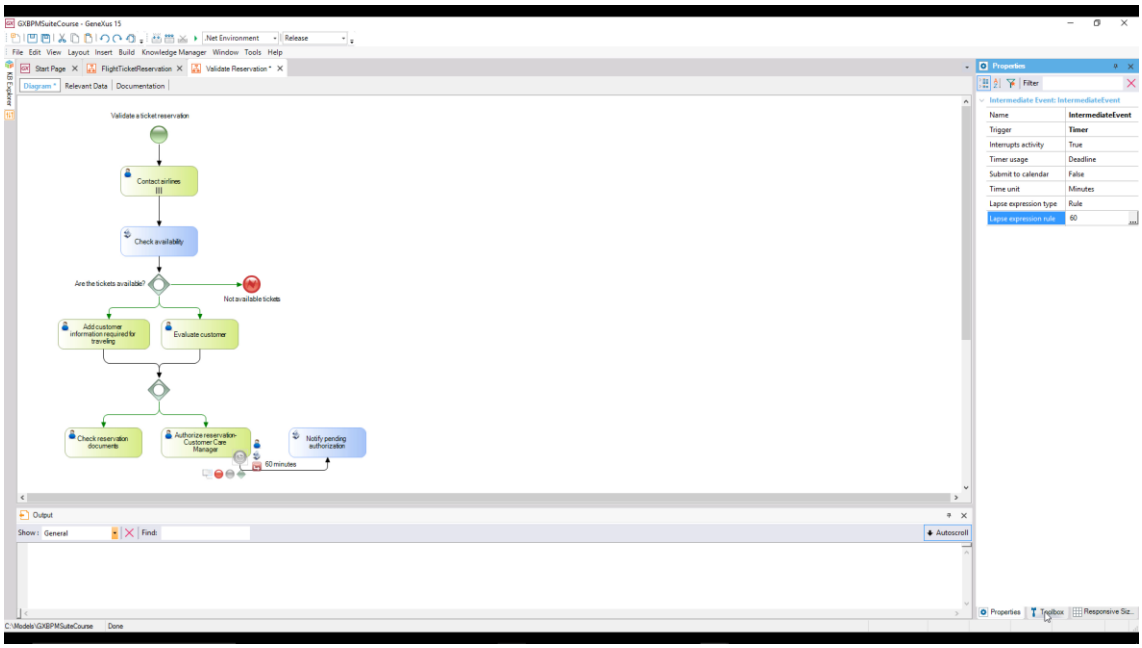
Para detectar el transcurso de un cierto tiempo, agregamos un intermediate event del tipo timer a la tarea.

Un evento es un suceso que ocurre durante la ejecución de un proceso y puede iniciar, pausar, interrumpir o finalizar dicho proceso. Según la naturaleza del evento, lo podemos clasificar en eventos de inicio, intermedios o de fin. Ya hemos utilizado los casos más sencillos de eventos de comienzo y de fin, como el none start event y el none end event respectivamente.

Ahora utilizaremos un evento intermedio, del tipo **timer**. El evento timer asociado a una tarea, dependiendo de su configuración, interrumpirá la ejecución de la misma una vez que ha transcurrido un cierto tiempo, logrando así limitar la duración de dicha tarea.

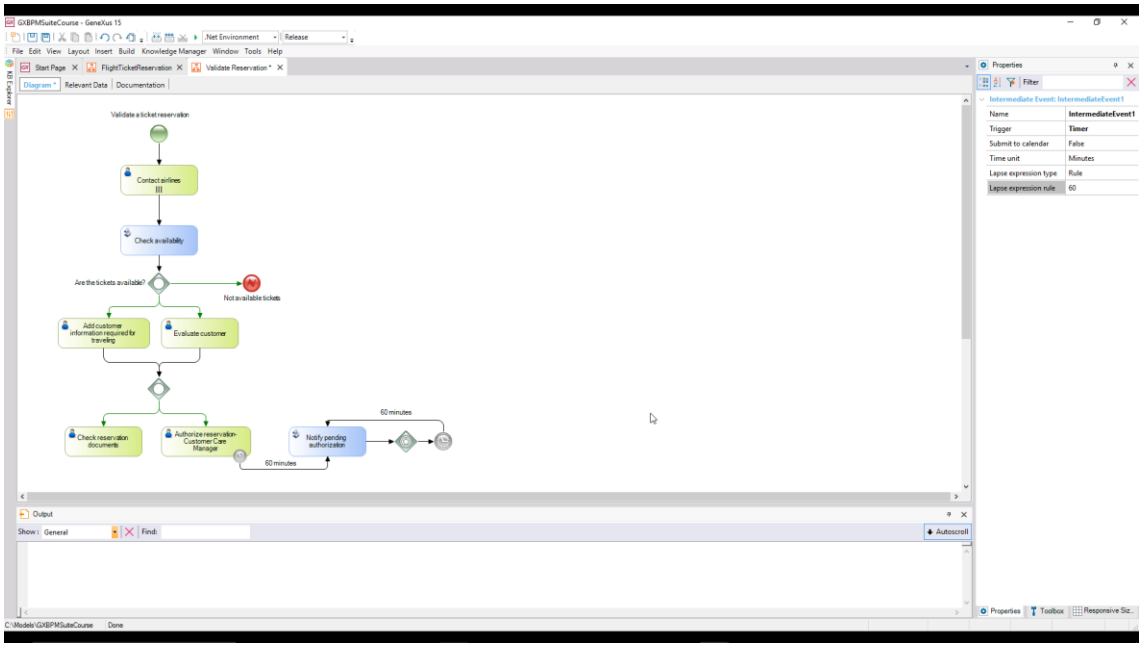


La notificación es automática, así que agregamos una tarea del tipo script con la descripción "Notify pending authorization" y la conectamos desde el timer event. A esa conexión la etiquetamos con el tiempo entre avisos periódicos, en este caso 60 minutos.



También vamos a las propiedades del intermediate timer event y en la propiedad Lapse expression rule escribimos 60.

Una vez que se produzca el aviso, queremos que el mismo se repita cada 60 minutos. Para lograr esto utilizamos un Event Gateway y un nuevo timer cuya salida conectamos a la tarea batch. A esta conexión le ponemos también 60 minutos y ajustamos la propiedad del timer para ello.

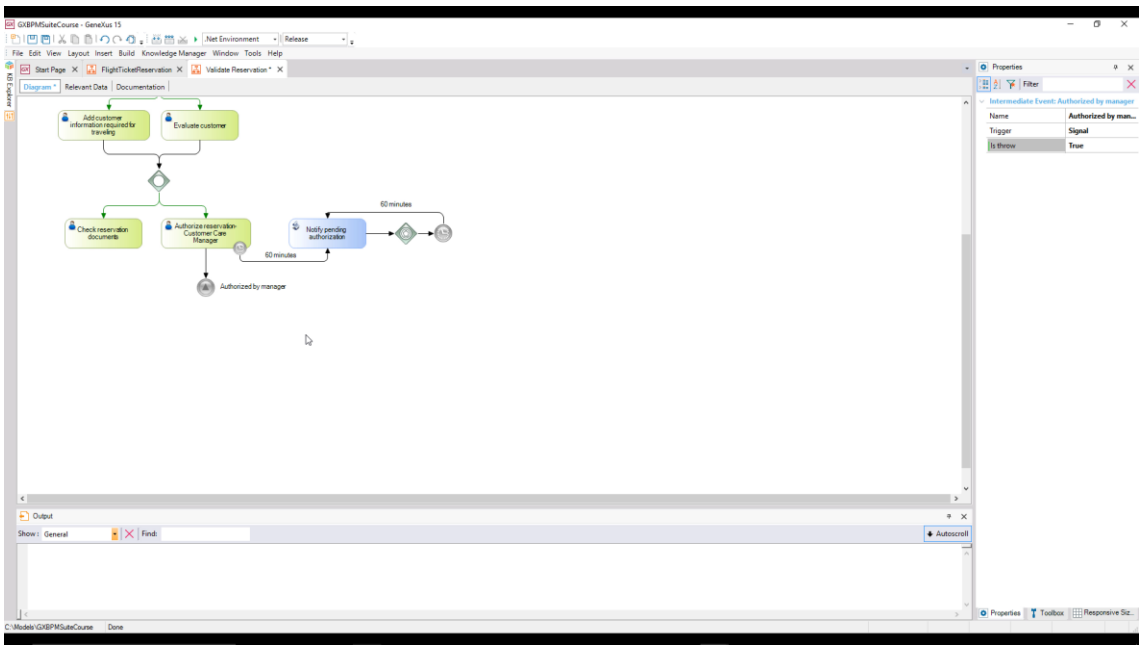


El event gateway se utiliza para que el proceso continúe por un determinado camino solamente si se produce un evento específico. En un caso particular del exclusive gateway porque solamente se podrá seguir por un camino, pero que en este caso no depende de una condición sino de la ocurrencia de un evento.

Continuando con nuestro modelo, en el caso de que el gerente autorice la reserva, se debe interrumpir este ciclo de notificaciones. Para generar un aviso, necesitamos que se dispare una señal y que la misma pueda ser detectada para interrumpir las notificaciones.

Para modelar un evento que **dispare** una señal, agregamos un símbolo de "signal intermediate event", le ponemos la descripción "Authorized by manager" y lo conectamos desde la tarea "Authorize reservation - Customer Care Manager".

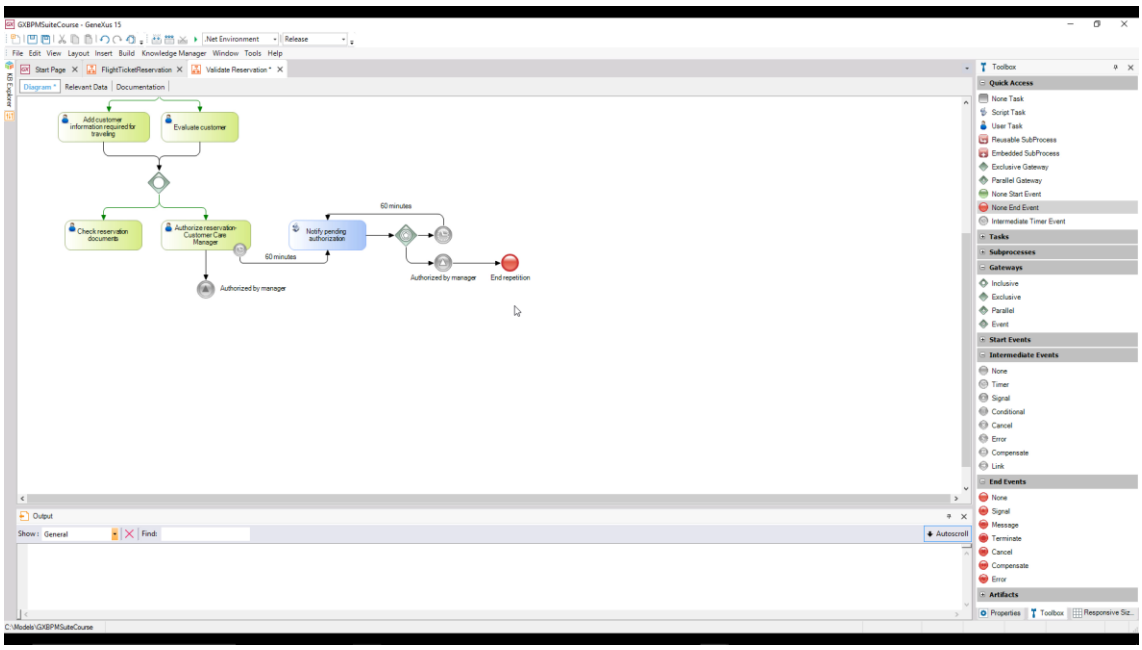
Como queremos indicar que la señal sea generada, modificamos la propiedad **Is Throw** y le ponemos el valor True.



Ahora debemos **atrapar** esta señal en el proceso de la generación de notificaciones.

Para eso agregamos otro "signal intermediate event" con la misma descripción que el que genera la señal "Authorized by manager" y lo conectamos desde el event Gateway. Luego agregamos un none end event y lo conectamos desde el signal event.

Como este signal intermadiate event tiene como función capturar la seña, dejamos su propiedad Is Throw en el valor False.

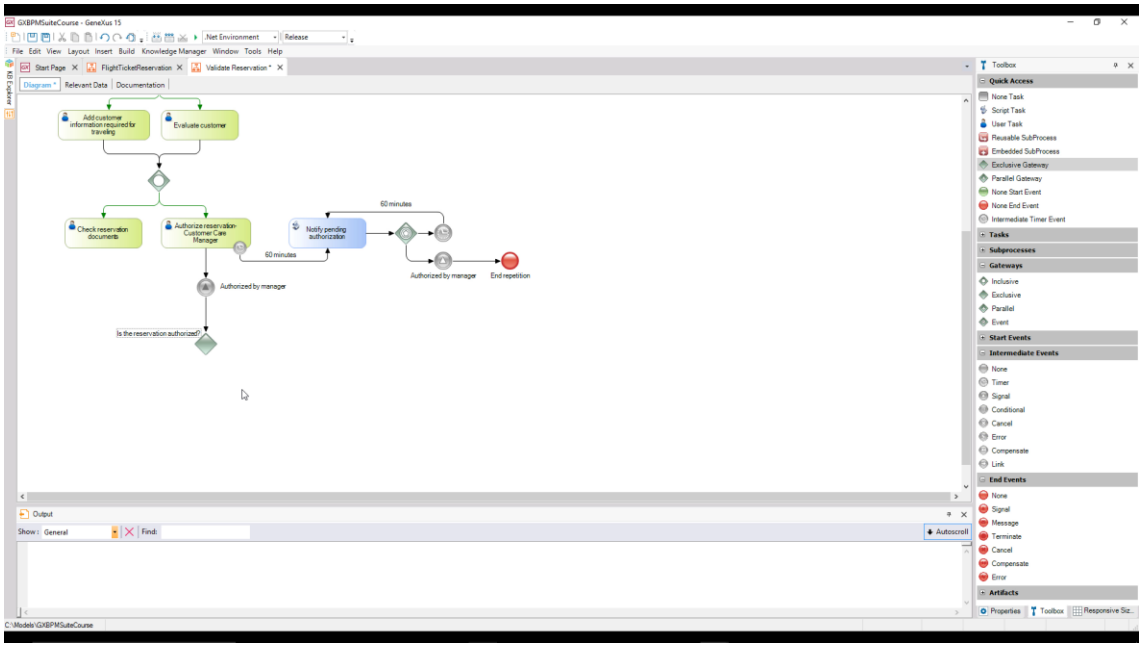


Todos los tipos de eventos intermedios pueden **atrapar** eventos (en inglés "*catch*"), algunos a su vez pueden **dispararlos** (en inglés "*throw*"). Cuando el flujo de un proceso llega a un evento de tipo *catch*, el proceso se detiene hasta que ocurra el evento esperado. En cambio cuando el flujo llega a un evento de tipo *throw* el evento es disparado inmediatamente y el flujo continúa.

Los tipos de eventos que atrapan eventos muestran su icono sin relleno mientras que los que disparan eventos muestran su icono con relleno, lo que vemos reflejado en los dos eventos signal que habíamos agregado.

Continuando con el proceso luego de que el gerente de atención al cliente cumpliera con el estudio de la autorización de la reserva, puede pasar que la misma sea autorizada o no.

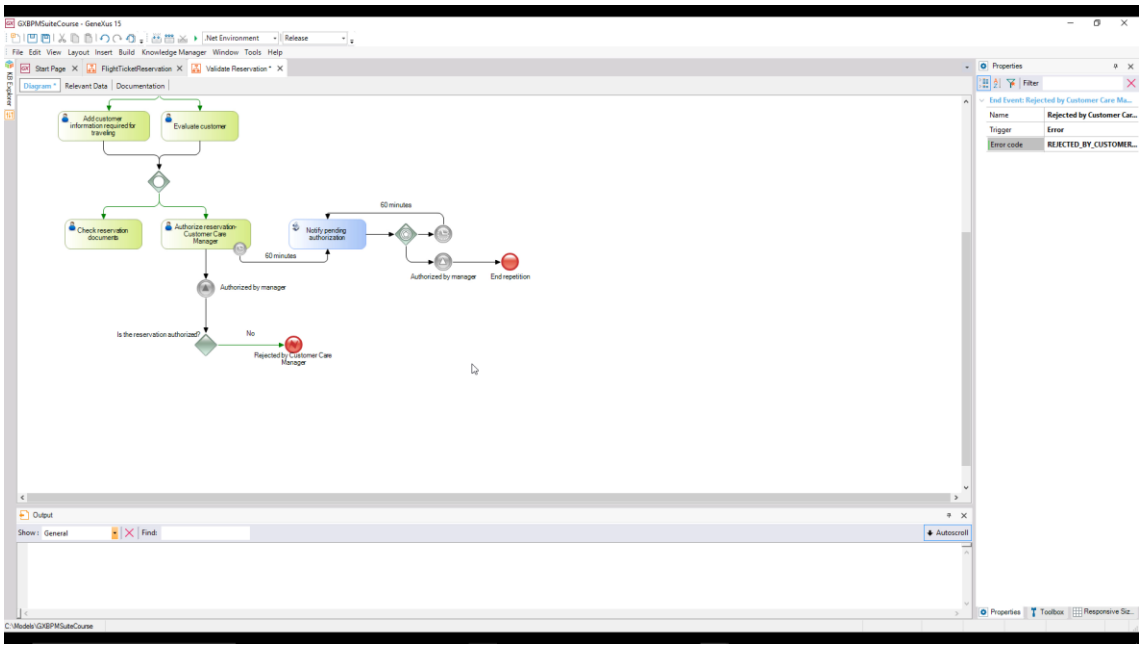
Para reflejar esto agregamos un exclusive gateway con la descripción "Is the reservation auhorized?" y lo conectamos desde el signal intermediate event.



Si la autorización es rechazada, debemos enviar una señal de error al proceso FlightTicketReservation, para que éste notifique al cliente del rechazo. Para eso agregamos un error end event que conectamos desde el

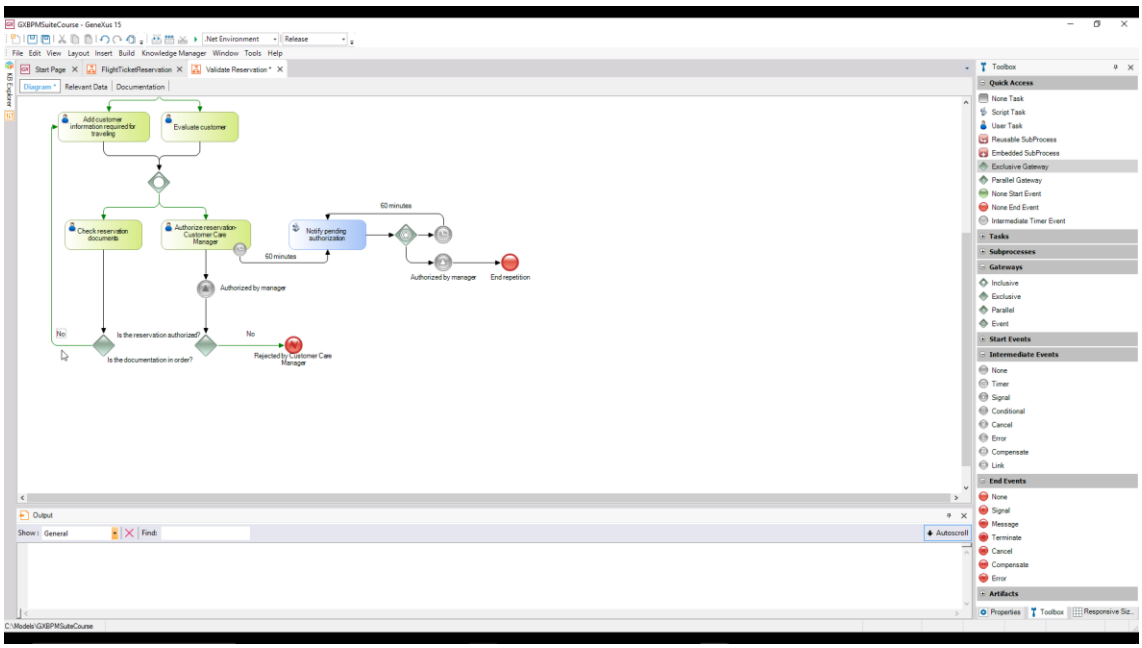
exclusive Gateway, a la conexión le ponemos la etiqueta “No” y al end event le ponemos como descripción “Rejected by Customer Care Manager”.

En su propiedad Error Code agregamos **REJECTED_BY_CUSTOMER_CARE_MANAGER**.

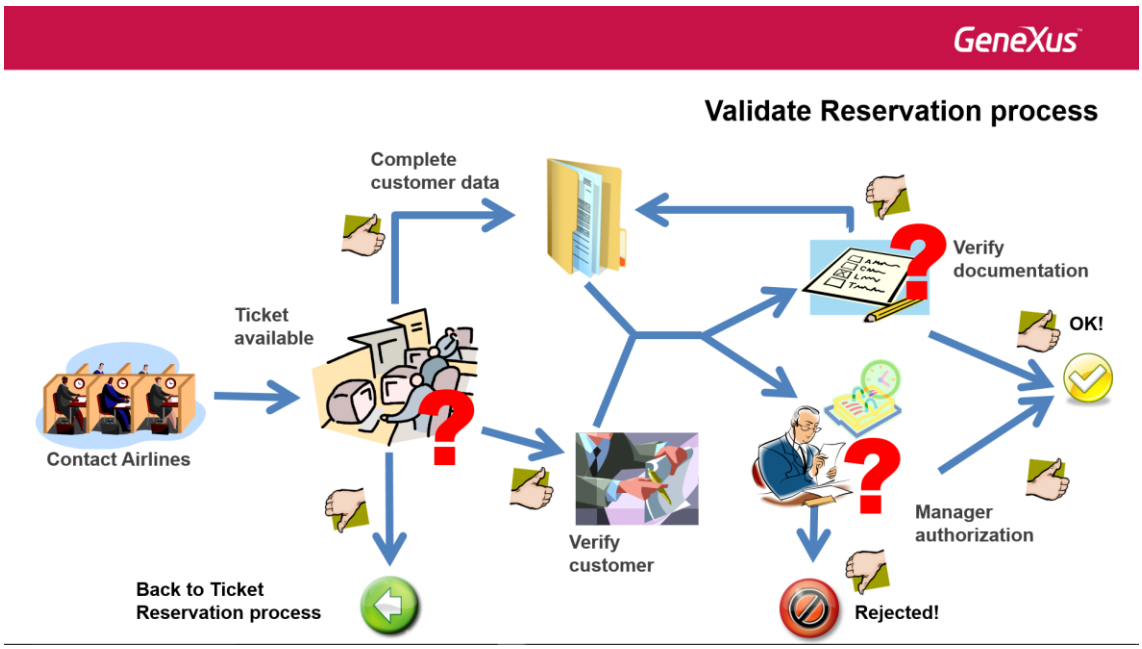


Luego de cumplida la tarea de verificación de la documentación, puede pasar que la documentación no esté en orden, en cuyo caso debe volverse a la tarea de colección de información. Para hacer este chequeo, agregamos un exclusive gateway con la descripción “Is the documentation in order?” y lo conectamos desde la tarea “Check reservation documents”.

Por último conectamos el gateway a la tarea “Add customer information required for traveling” y al conector le ponemos la etiqueta “No”.

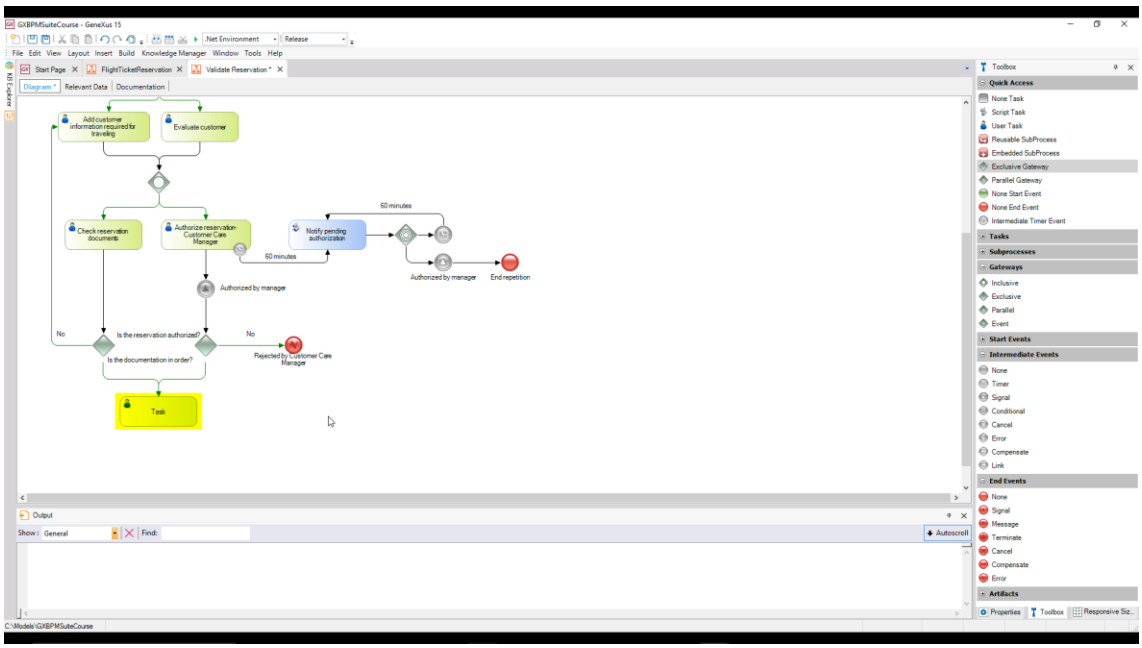


Una vez aprobados los controles de la documentación y otorgada la autorización del gerente de atención al cliente, se debe dar como válida a la reserva.



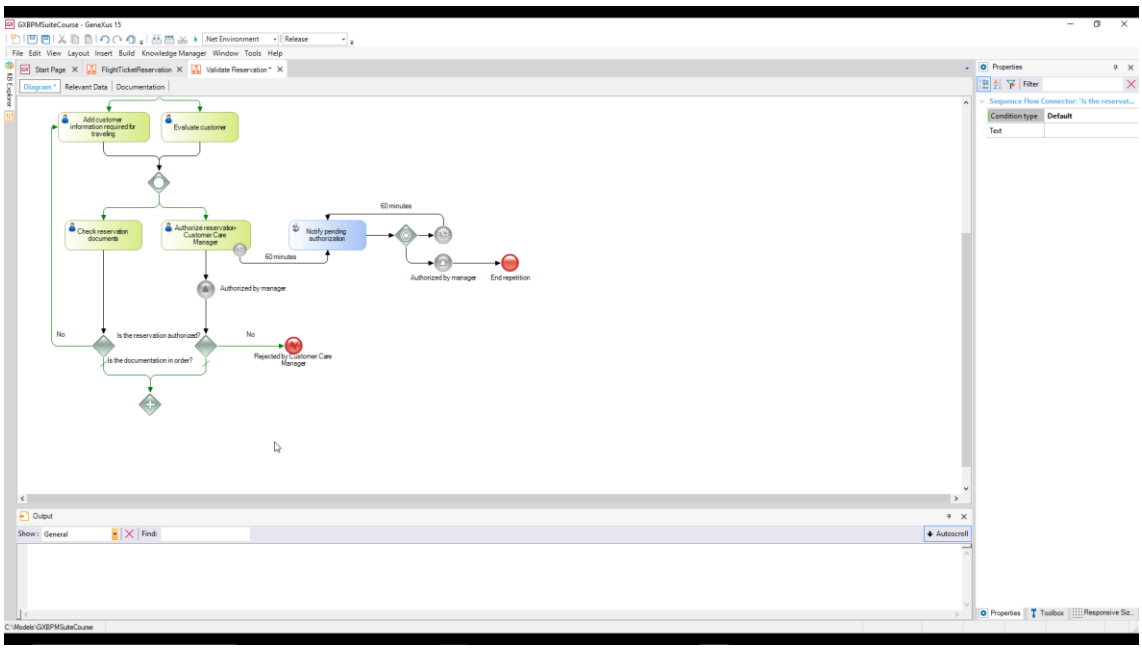
Para poder continuar, ambas tareas deben estar finalizadas y si es necesario debe esperarse por una de ellas hasta que esto suceda.

La sincronización es necesaria, porque si simplemente juntáramos ambos caminos…



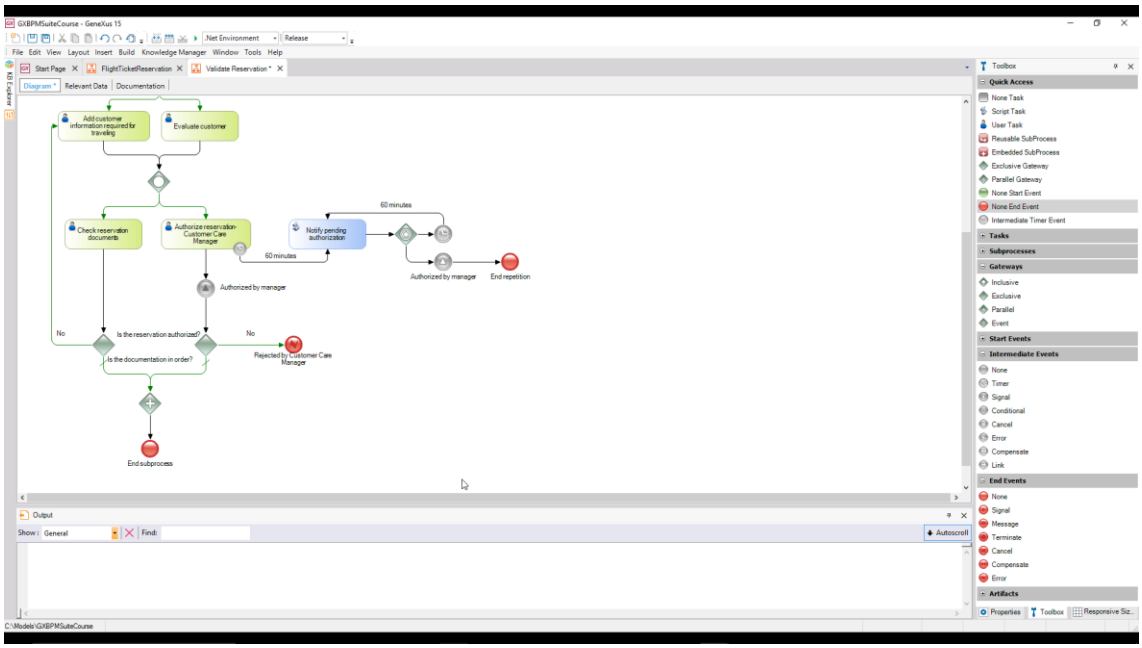
… puede pasar que una de las etapas anteriores se ejecute primero, con lo cual se ejecutaría la tarea siguiente y luego, al ejecutarse la etapa que estaba pendiente, la tarea siguiente a la unión se ejecutaría por segunda vez!

Para hacer esta sincronización, en la que se debe esperar por ambos caminos, agregamos un **parallel exclusive gateway** y lo unimos desde los exclusive gateways.



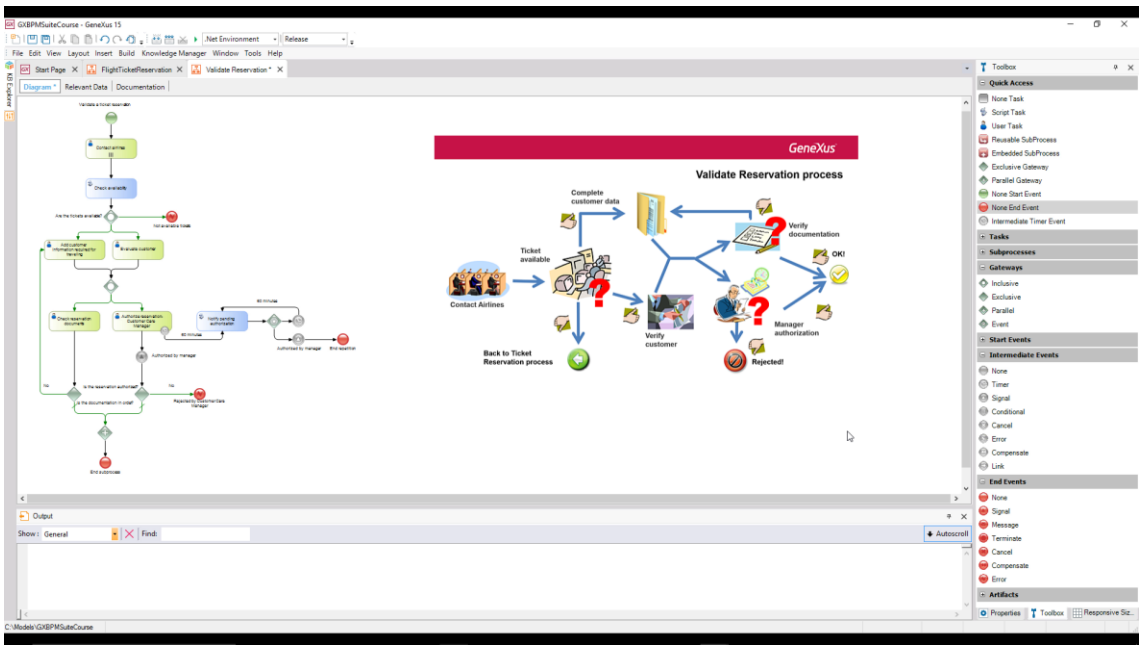
Para señalizar que ambos caminos que parten de los exclusive gateways son los caminos por defecto, ponemos el valor Default a sus respectivas propiedades Condition type,

Una vez que ambos caminos finalizan, debe finalizar el proceso de validación de reservas de tickets, por lo que insertamos un none end event y lo conectamos desde el parallel Gateway.

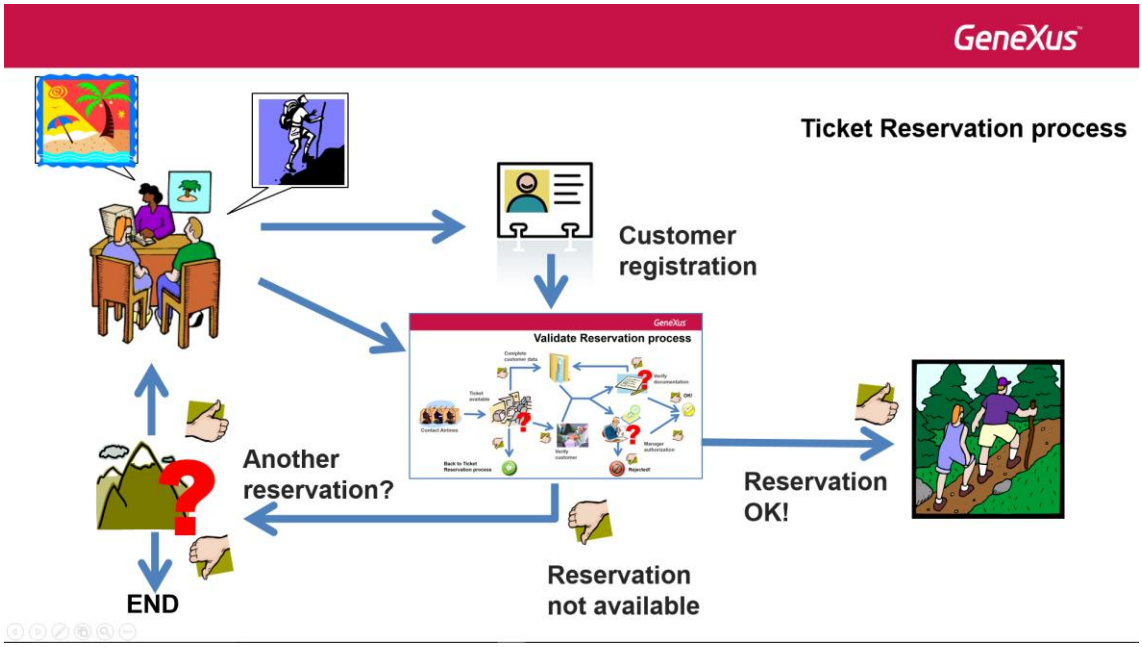


Presionando botón derecho sobre el diagrama, ajustamos el zoom con Zoom Out hasta que podamos ver el diagrama completo.

Con esto damos por terminado el diagrama del subproceso de validación de reservas de tickets,



que integra el proceso principal de reserva de tickets aéreos.



Este video junto al de Modelado - Primera Parte nos brindó un rápido vistazo al modelado de procesos, una breve introducción al estándar BPMN y las facilidades que GeneXus nos proporciona para construir diagramas que cumplan con dicho estándar.

En videos siguientes veremos otros aspectos del modelado de procesos de negocios, como su automatización, monitoreo, optimización, puesta en producción y mantenimiento de los mismos.