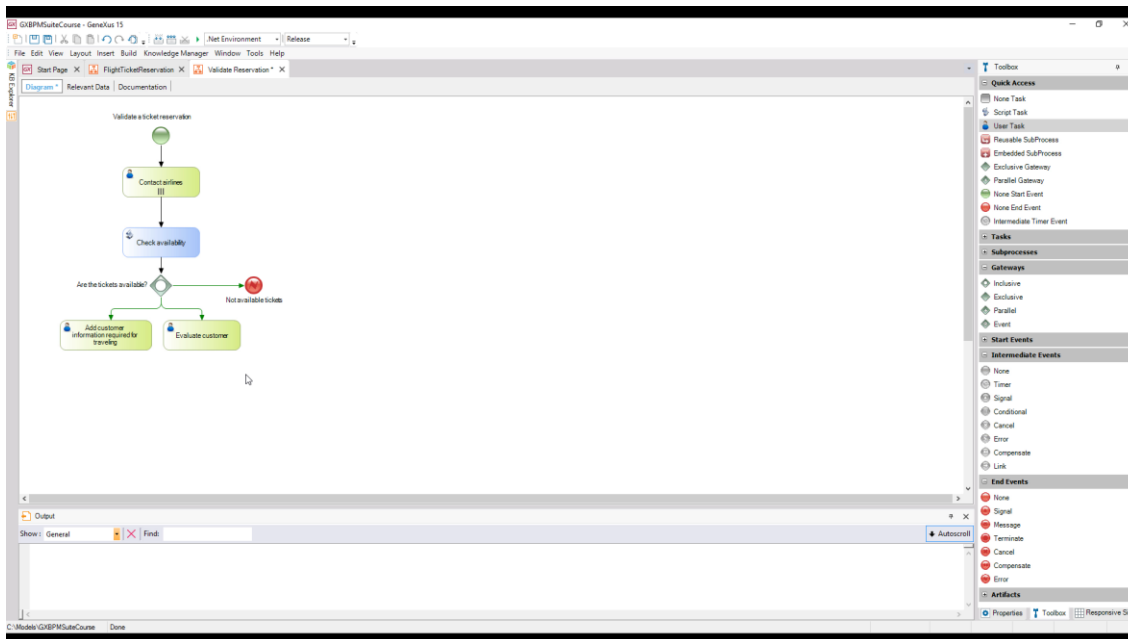


Bifurcação e união de caminhos, geração de avisos periódicos e gestão de sinais

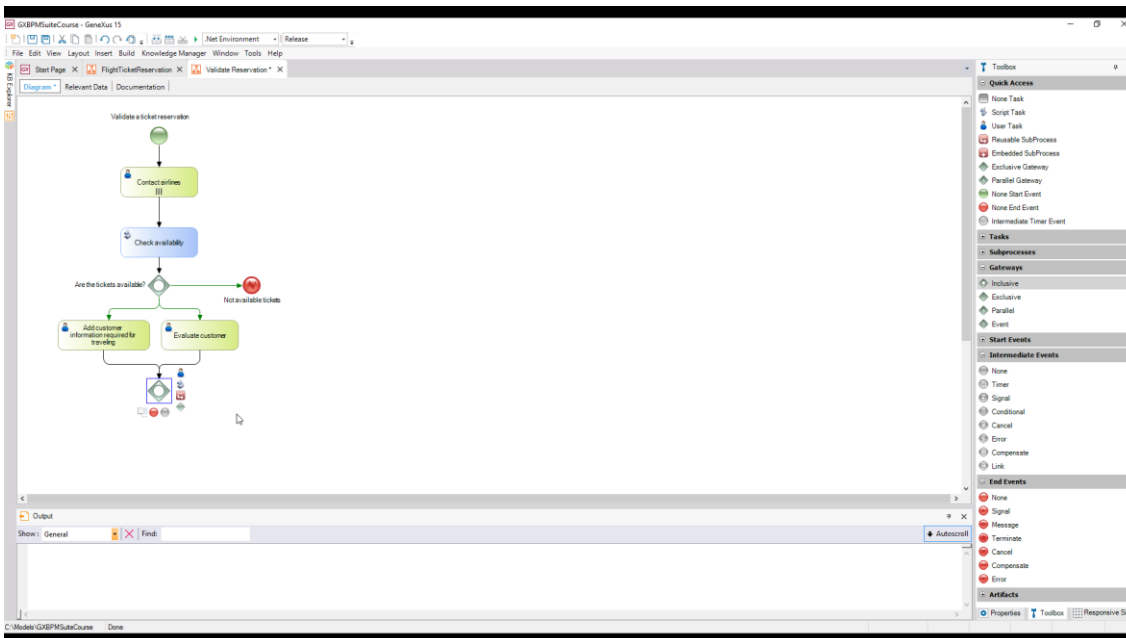
Voltando ao modelo do processo de validação da reserva, se houver disponibilidade de passagens, devemos continuar por 2 caminhos simultaneamente. Adicionamos as tarefas interativas “Add customer information required for traveling” e “Evaluate Customer”, conectando ambas partindo do inclusive gateway.



Neste caso, estamos utilizando o inclusive gateway para ramificar caminhos. Cada caminho tem uma condição definida e o processo seguirá por um ou mais caminhos simultâneos, dependendo das condições cumpridas.

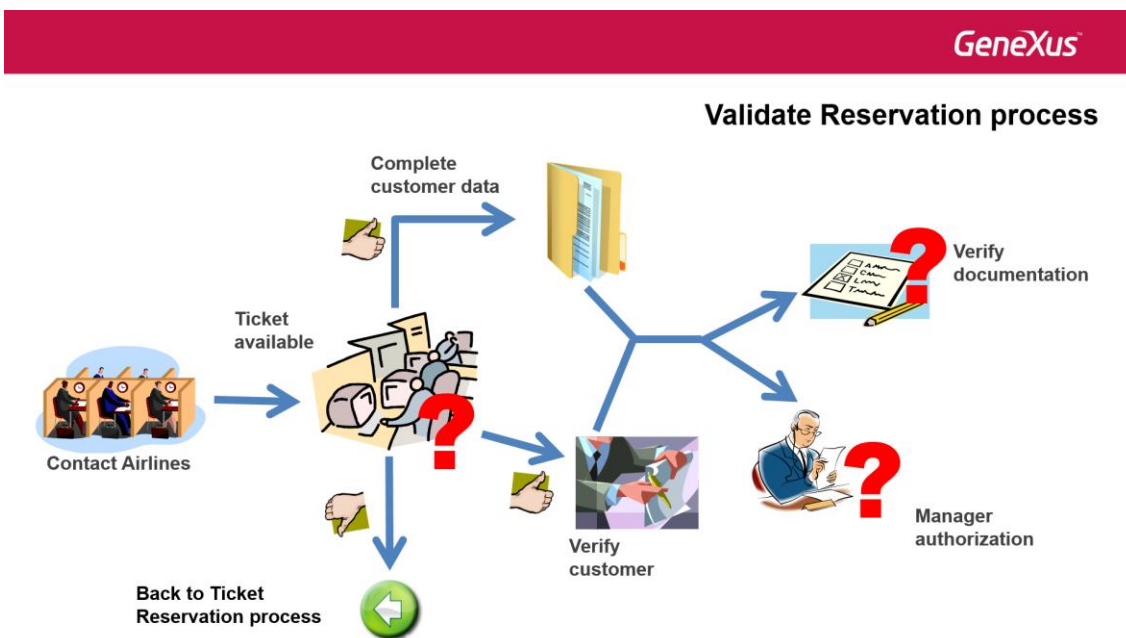
Depois de obter a informação para a viagem e verificar a situação financeira do cliente, é preciso continuar com o processo, mas para poder continuar, ambas as tarefas precisam estar concluídas.

Ou seja, se uma tarefa termina antes que outra, é preciso esperar que a segunda termine para continuar o processo. Para obter isso, usamos também um inclusive gateway, que o arrastamos da barra de ferramentas e conectamos o processo partindo das duas tarefas.

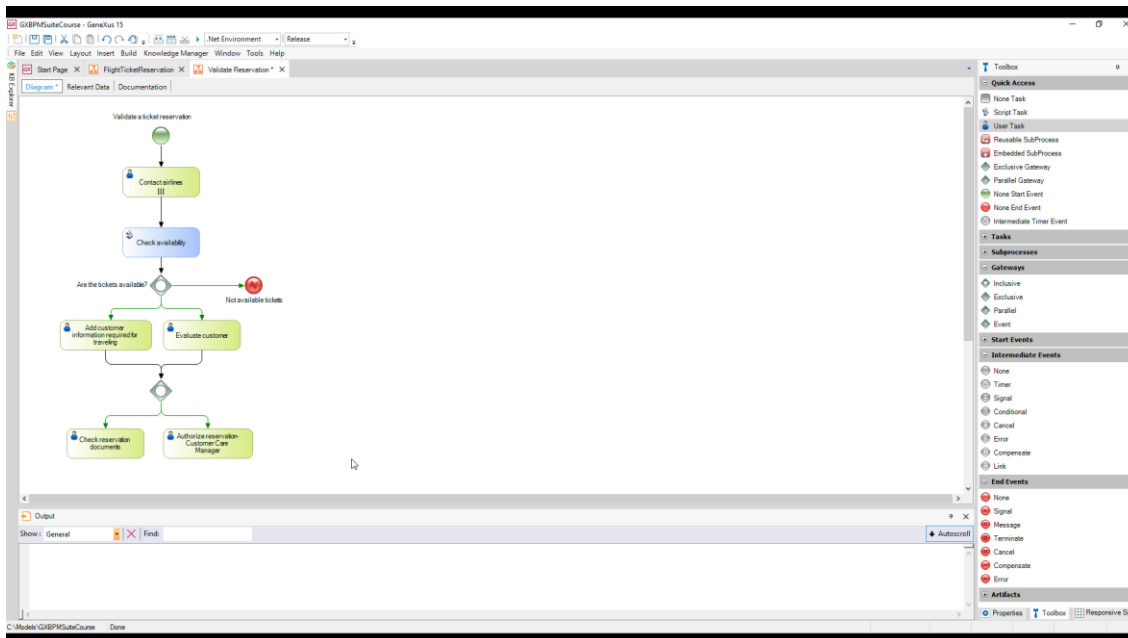


Na unificação de caminhos, o inclusive gateway realiza uma sincronização dos caminhos por onde o processo avançou, que não são necessariamente todos os caminhos que chegam ao gateway.

Em nosso caso, o processo pode avançar a partir de qualquer uma das 2 tarefas e os dois são únicos caminhos que chegam ao inclusive gateway. Depois que as duas terminarem, continuaremos com o processo, que também implica na execução de 2 atividades: a verificação dos dados para a viagem e a autorização do gerente de atendimento ao cliente.



Para modelar a divergência do fluxo em 2 caminhos, utilizaremos um inclusive gateway como no caso anterior, de modo que aproveitamos o mesmo inclusive gateway que usamos para unir os caminhos e criamos as tarefas Check reservation documents e Authorize Reservation - Customer Care Manager, conectando-as a partir deste nó.



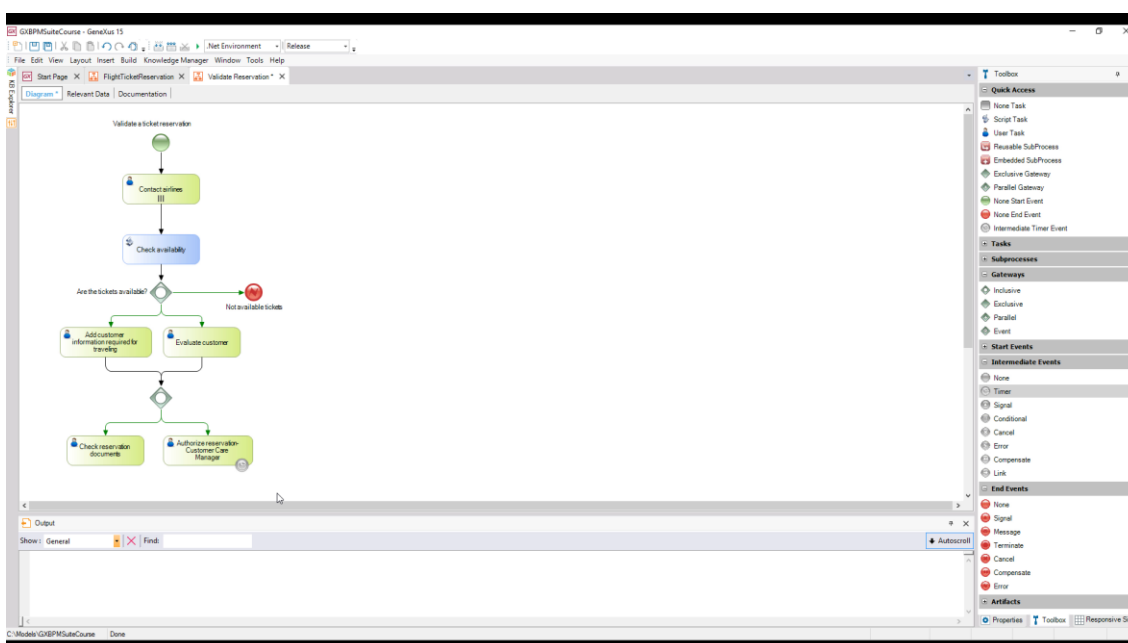
A tarefa que o gerente de atendimento ao cliente executa deve ter um tempo limite e queremos que o sistema possa avisá-lo periodicamente que falta autorizar uma reserva.

Para detectar a passagem de determinado tempo, adicionamos um intermediate event do tipo timer à tarefa.

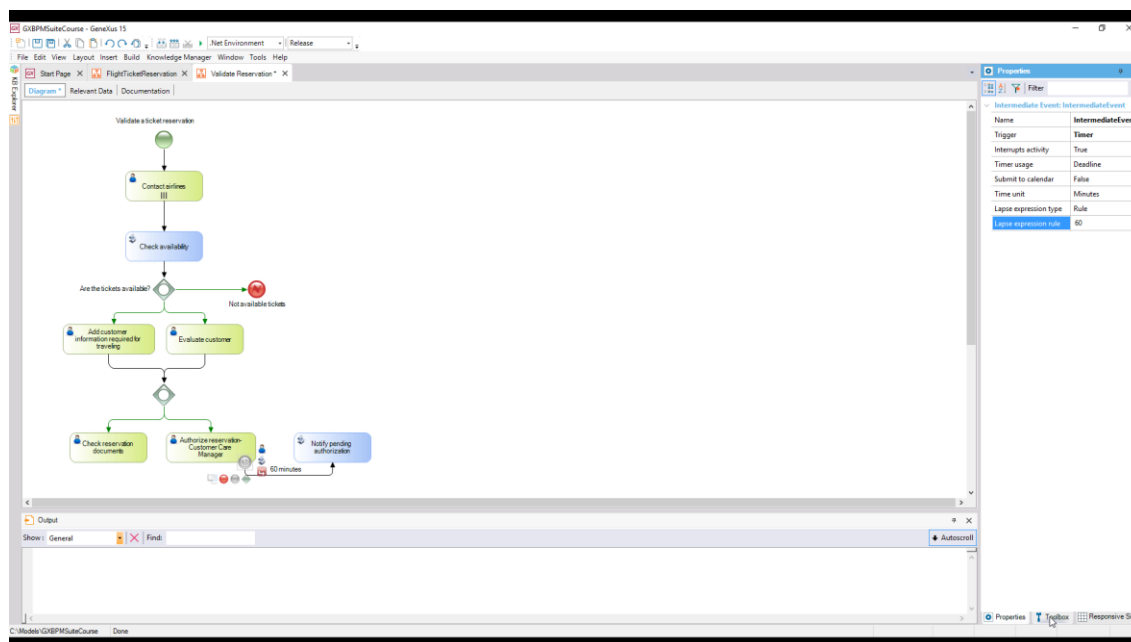
Um evento é um acontecimento que ocorre durante a execução de um processo e pode iniciar, pausar, interromper ou finalizar este processo. Segundo a natureza do evento, podemos classificá-lo como evento de início, intermediário ou de fim.

Já utilizamos os casos mais simples de eventos de começo e de fim, como o none start event e o none end event, respectivamente. Agora, utilizaremos um evento intermediário do tipo **timer**.

O evento timer, associado a uma tarefa, dependendo de sua configuração, interromperá a execução dela assim que seu tempo se esgotar, conseguindo dessa forma, limitar a duração dessa tarefa.

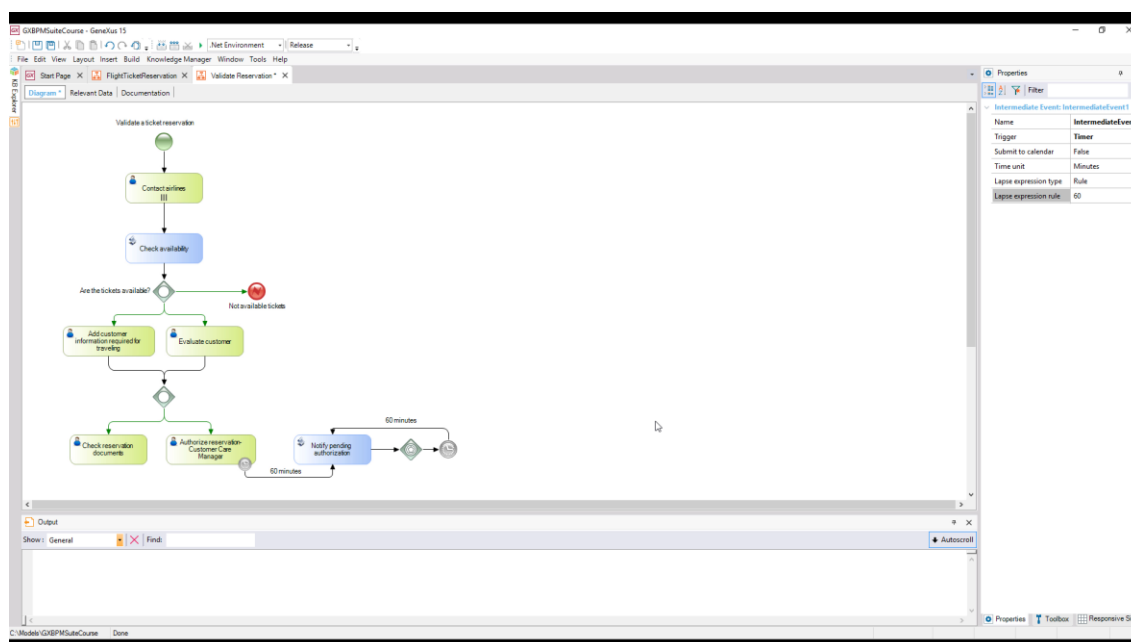


A notificação é automática. Adicionamos uma tarefa do tipo script com a descrição “Notify pending authorization” e a conectamos partindo do timer event. Rotulamos essa conexão com o tempo entre avisos periódicos, neste caso, 60 minutos.



Também vamos às propriedades do intermediate timer event, na propriedade Lapse expression rule e escrevemos 60.

Uma vez que seja feito o aviso, queremos que ele se repita a cada 60 minutos. Para conseguir isso, utilizamos um Event Gateway e um novo timer cuja saída conectamos à tarefa batch. A essa conexão também colocamos 60 minutos e ajustamos a propriedade do timer para isso.

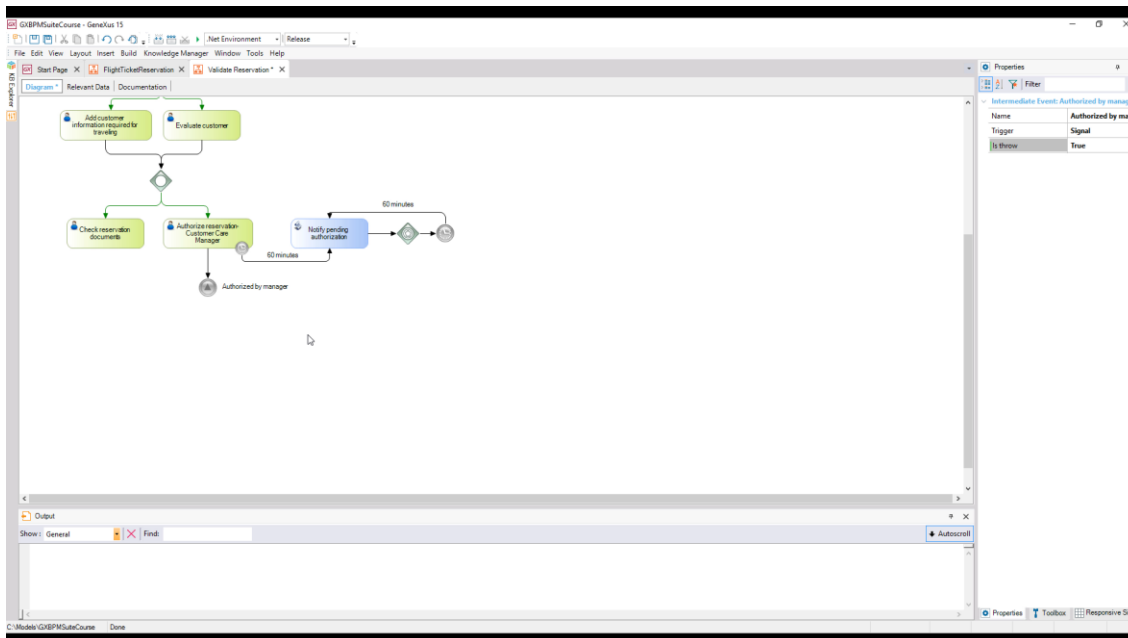


O evento gateway é utilizado para que o processo continue por um determinado caminho apenas se um evento específico for produzido. É um caso particular do exclusive gateway, porque poderá seguir somente por um caminho, mas que neste caso não depende de uma condição, mas da ocorrência de um evento.

Continuando com nosso modelo, no caso de o gerente autorizar a reserva, deve-se interromper este ciclo de notificações. Para gerar um aviso, precisamos que se dispare um sinal e que ele possa ser detectado para interromper as notificações.

Para modelar um evento que **dispare** um sinal, adicionamos um símbolo de “signal intermediate event”, colocamos a descrição “Authorized by manager”, e o conectamos partindo da tarefa “Authorize reservation - Customer Care Manager”.

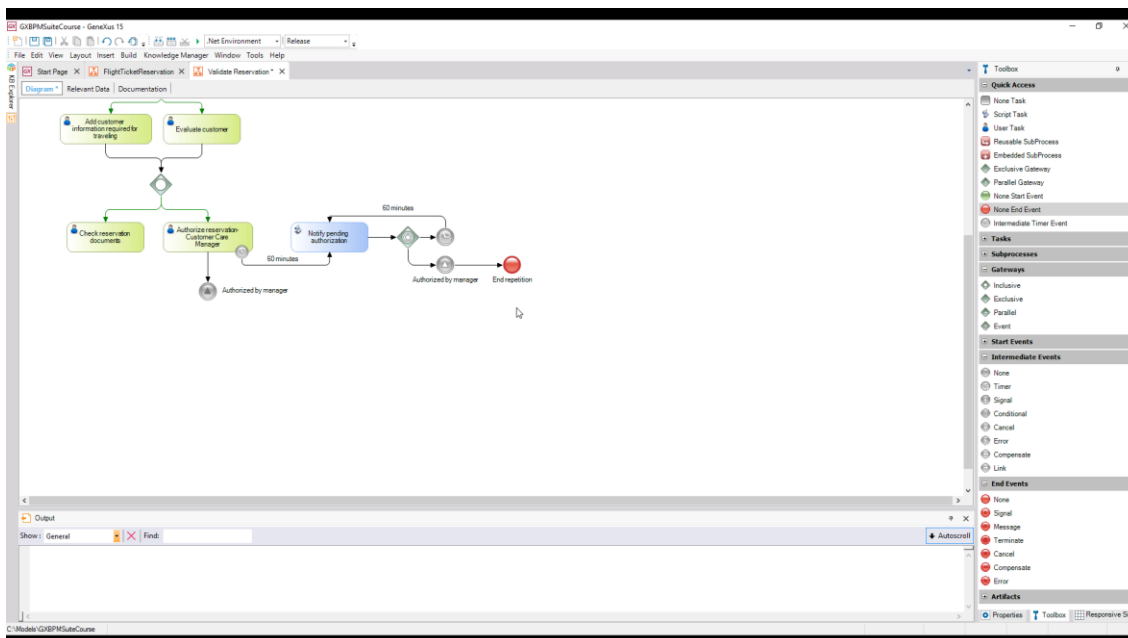
Como queremos indicar que o sinal seja gerado, modificamos a propriedade **Is Throw** e colocamos o valor True.



Agora devemos **pegar** esse sinal no processo de geração de notificações.

Para isso, adicionamos outro “signal intermediate event” com a mesma descrição que o que gera o sinal “Authorized by manager” e o conectamos partindo do evento Gateway. Depois, adicionamos um None end event e conectamos partindo do signal event.

Como este signal intermediate event tem como função capturar o sinal, deixamos sua propriedade Is Throw no valor False.

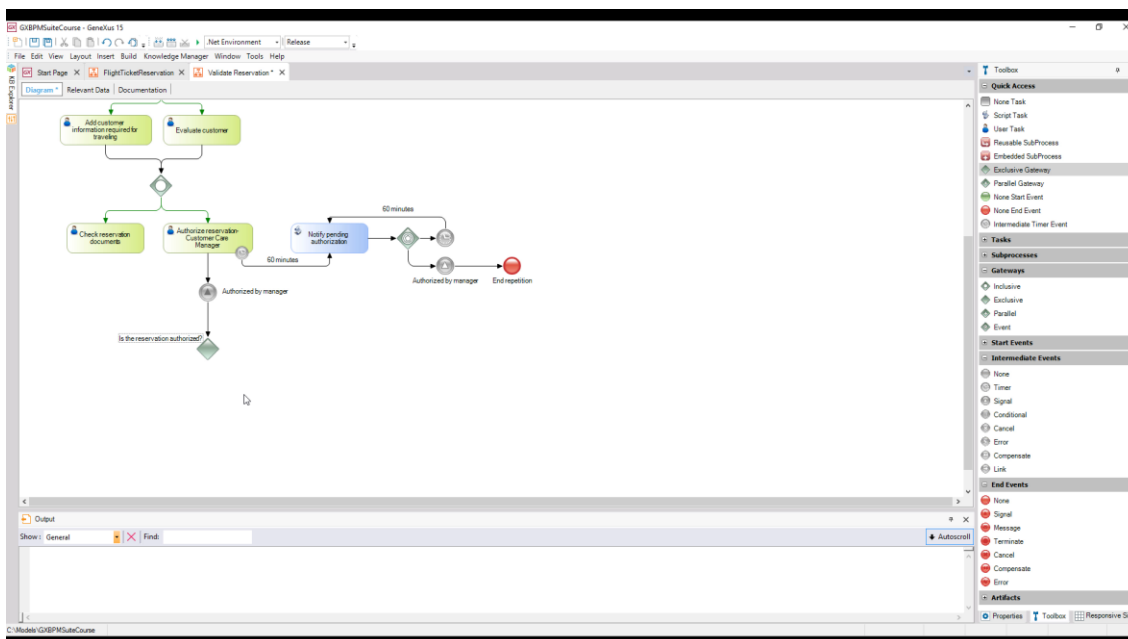


Todos os tipos de eventos intermediários podem **pegar** eventos (*catch*, em inglês), alguns podem **dispará-los** (em inglês, *throw*). Quando o fluxo de um processo chega a um evento do tipo *catch*, o processo detém-se até que ocorra o evento esperado. Porém, quando o fluxo chega a um evento do tipo *throw*, o evento é disparado imediatamente e o fluxo continua.

Os tipos de eventos que pegam eventos mostram seu ícone vazio enquanto os que disparam mostram seu ícone preenchido, o que vemos refletido nos dois eventos signal que tínhamos adicionado.

Continuando com o processo, depois que o gerente de atendimento ao cliente estudar a autorização da reserva, ele pode dar uma resposta positiva ou negativa.

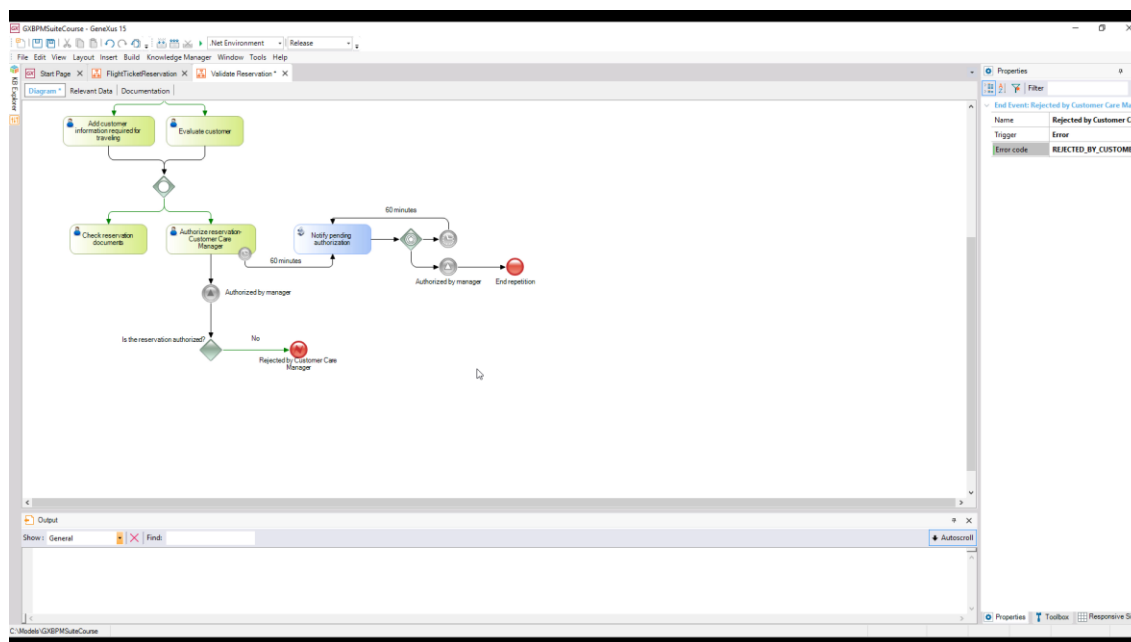
Para refletir isso, adicionamos um exclusive gateway com a descrição "Is the reservation authorized?" e conectamos partindo do signal intermediate event.



Se a autorização for recusada, devemos enviar um sinal de erro ao processo FlightTicketReservation para que ele notifique o cliente da recusa. Para isso, adicionamos um error end event que conectamos desde o

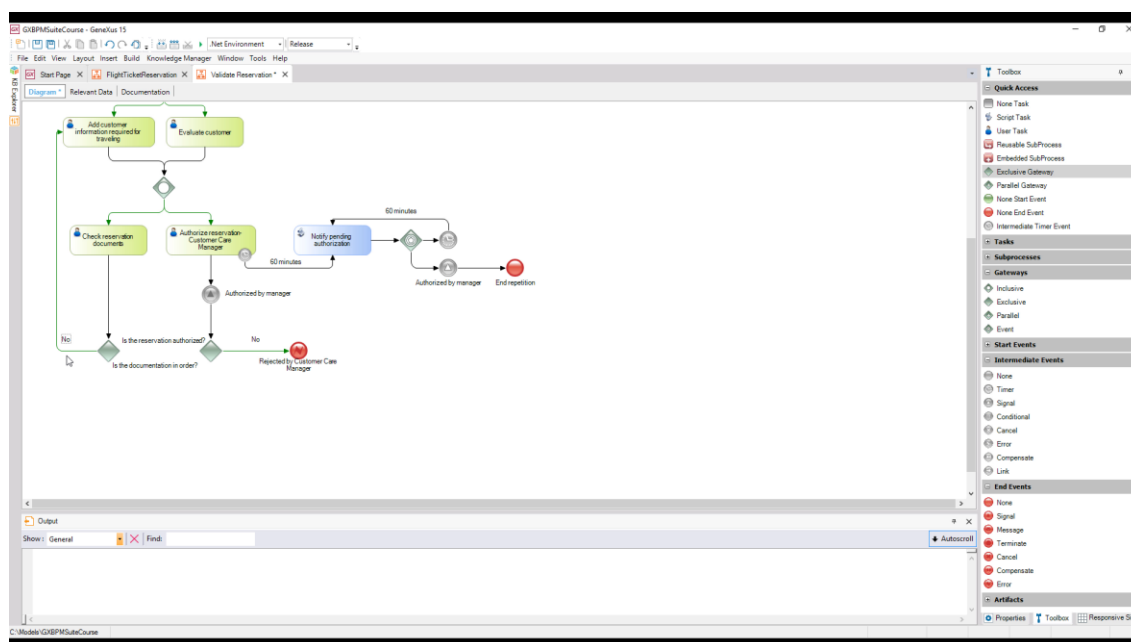
exclusive Gateway. Colocamos a etiqueta “No” na conexão e no end event colocamos a descrição “Rejected by Customer Care Manager”.

Em sua propriedade Error Code, adicionamos **REJECTED_BY_CUSTOMER_CARE_MANAGER**.



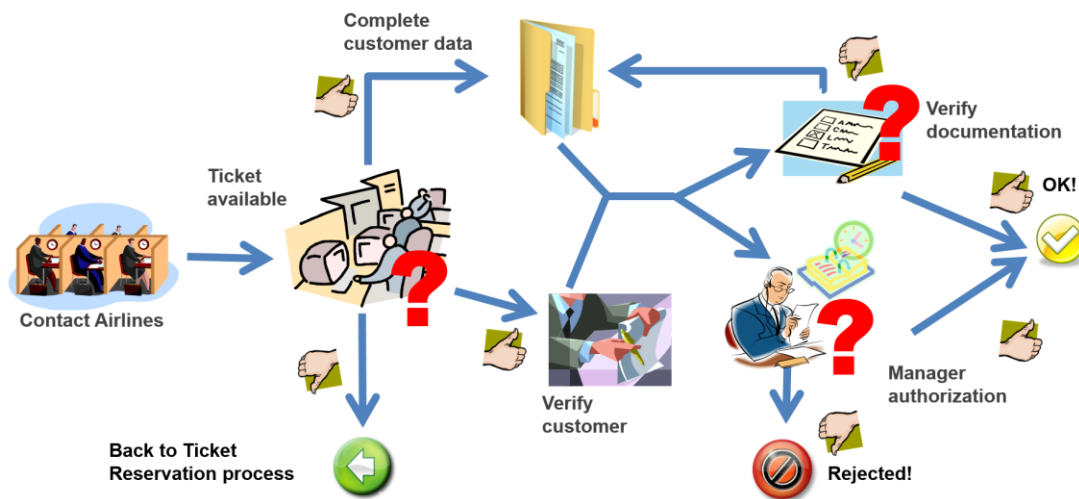
Após cumprir a tarefa de verificação da documentação, pode acontecer de a documentação não estar correta, situação em que se faz necessário voltar à tarefa de coleta de informação. Para checar isso, adicionamos um exclusive gateway com a descrição “Is the documentation in order?” e o conectamos partindo da tarefa “Check reservation documents”.

Por último, conectamos o gateway à tarefa “Add customer information required for traveling” e ao conector colocamos a etiqueta “No”.



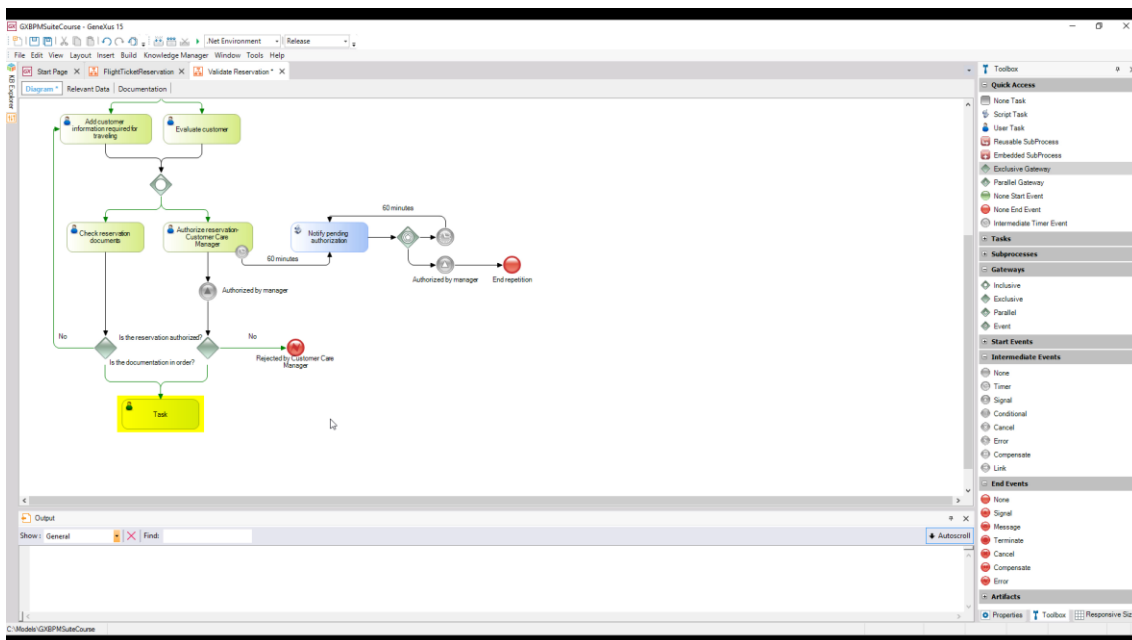
pois que os controles da documentação forem aprovados e autorização do gerente de atendimento ao cliente confirmada, deve-se considerar a reserva válida.

Validate Reservation process



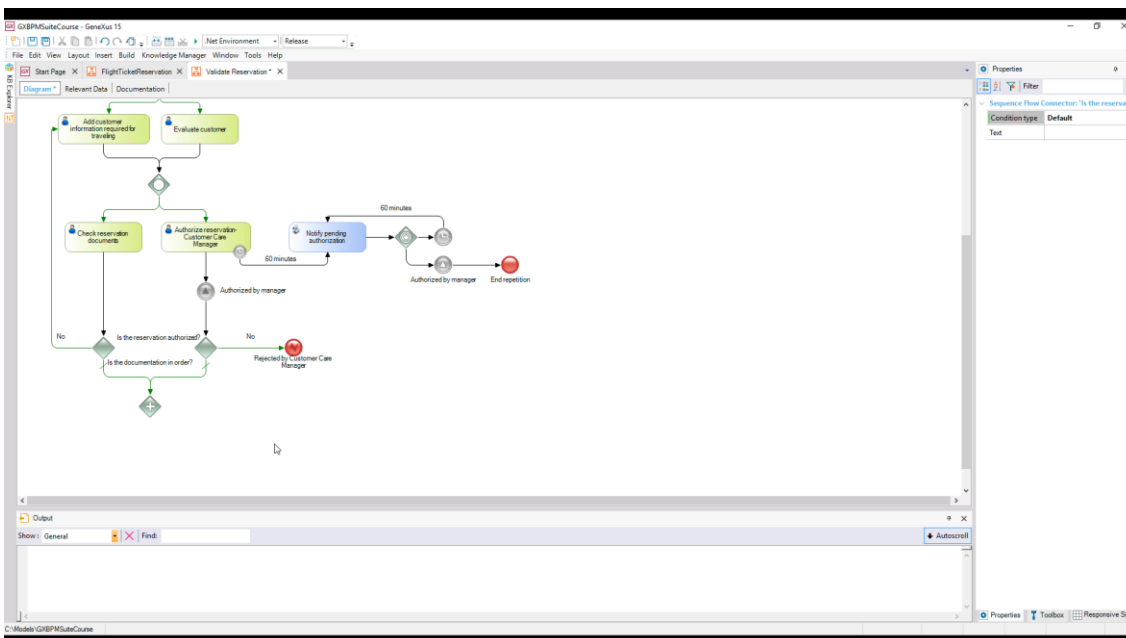
Para poder continuar, ambas as tarefas devem estar finalizadas e, se for necessário, deve-se esperar por uma delas até que isso aconteça.

A sincronização é necessária porque se simplesmente juntássemos os dois caminhos...



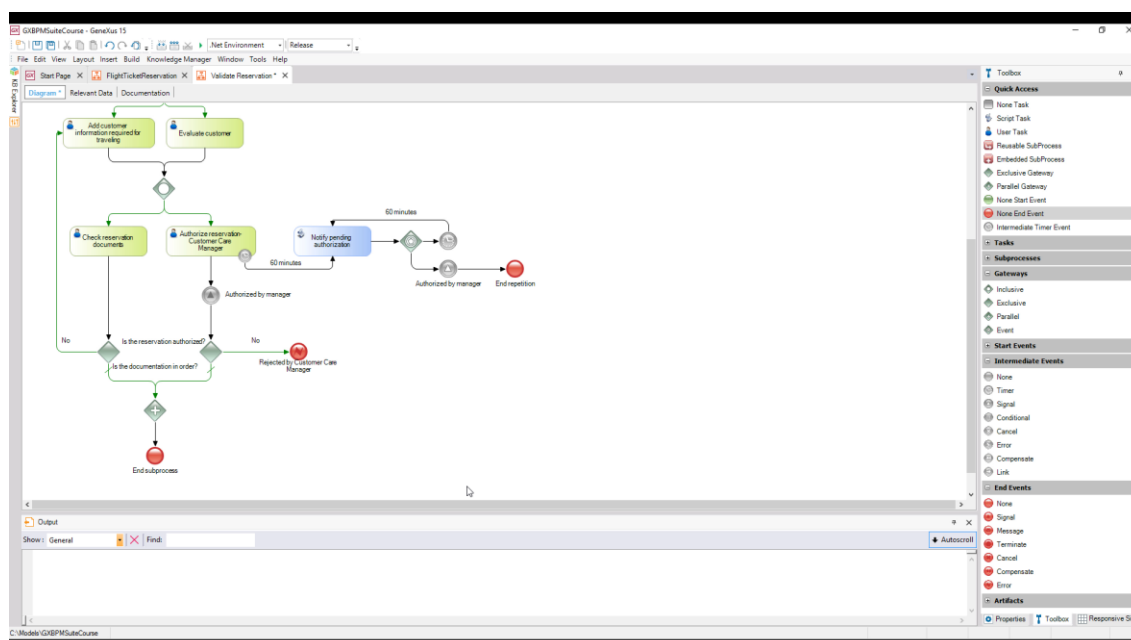
... uma das etapas poderia ser executada antes, motivo pelo qual a tarefa seguinte seria executada e, depois, ao executar a etapa que estava faltando, a tarefa seguinte à união seria executada pela segunda vez!

Para fazer essa sincronização, na qual se deve esperar por ambos os caminhos, adicionamos um **parallel exclusive gateway** e o unimos partindo dos exclusive gateways.



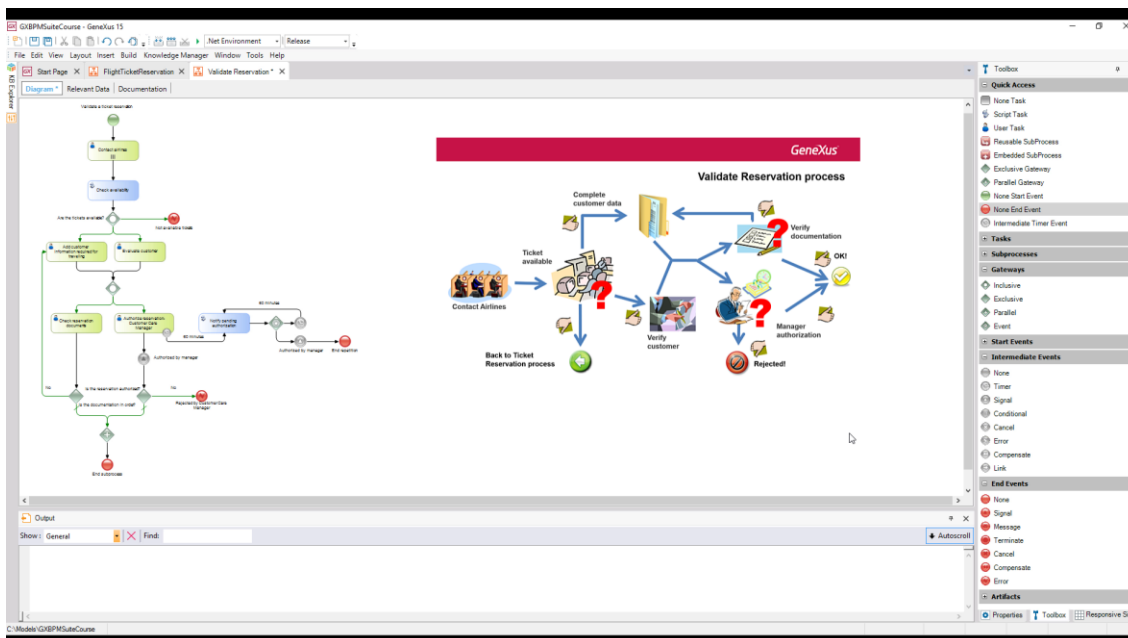
Para assinalar que ambos os caminhos que partem dos exclusive gateways são os caminhos pré-determinados, colocamos o valor Default em suas respectivas propriedades Condition Type.

Finalizados os dois caminhos, o processo de validação de reservas de passagens deve terminar e, para isso, inserimos um none end event e o conectamos partindo do parallel Gateway.

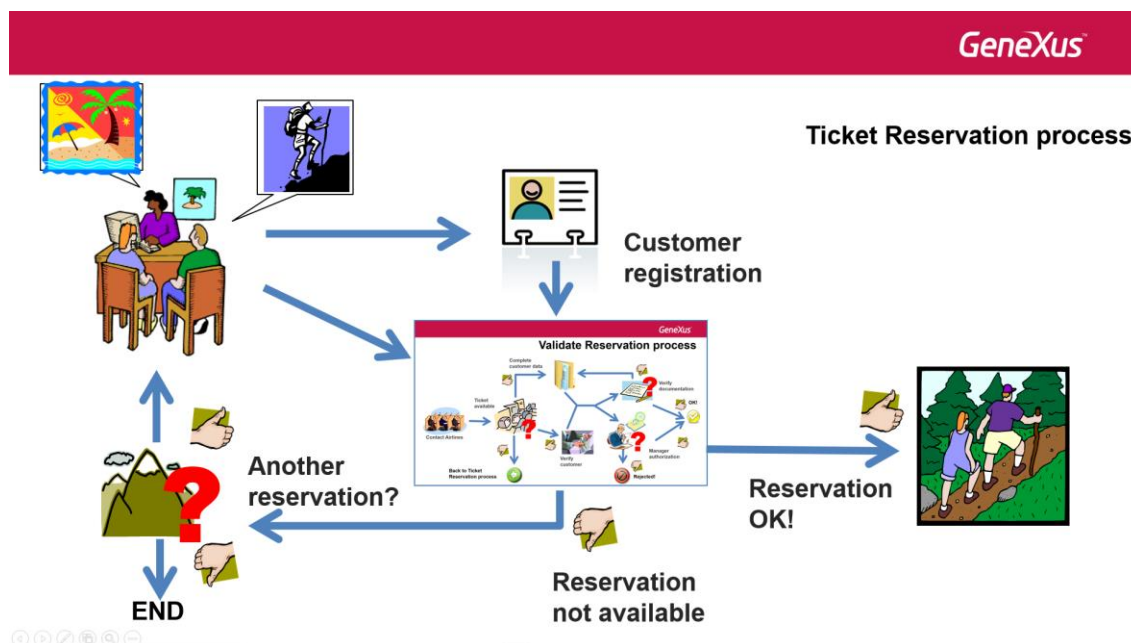


Pressionando o botão direito sobre o diagrama, ajustamos o zoom com Zoom Out até que possamos ver o diagrama completo.

Com isso, concluímos o diagrama do subprocesso de validação de reservas de passagens...



que integra o processo principal de reserva de passagens aéreas.



Este vídeo, junto com o de Modelagem – Primeira Parte, permitiu nos aproximarmos da modelagem de processos, nos ofereceu uma breve introdução ao padrão BPMN e às facilidades que GeneXus nos proporciona para construir diagramas que comprem com esse padrão.

Nos próximos vídeos, veremos outros aspectos da modelagem de processos, como sua automação, monitoramento, otimização, colocação em produção e manutenção.