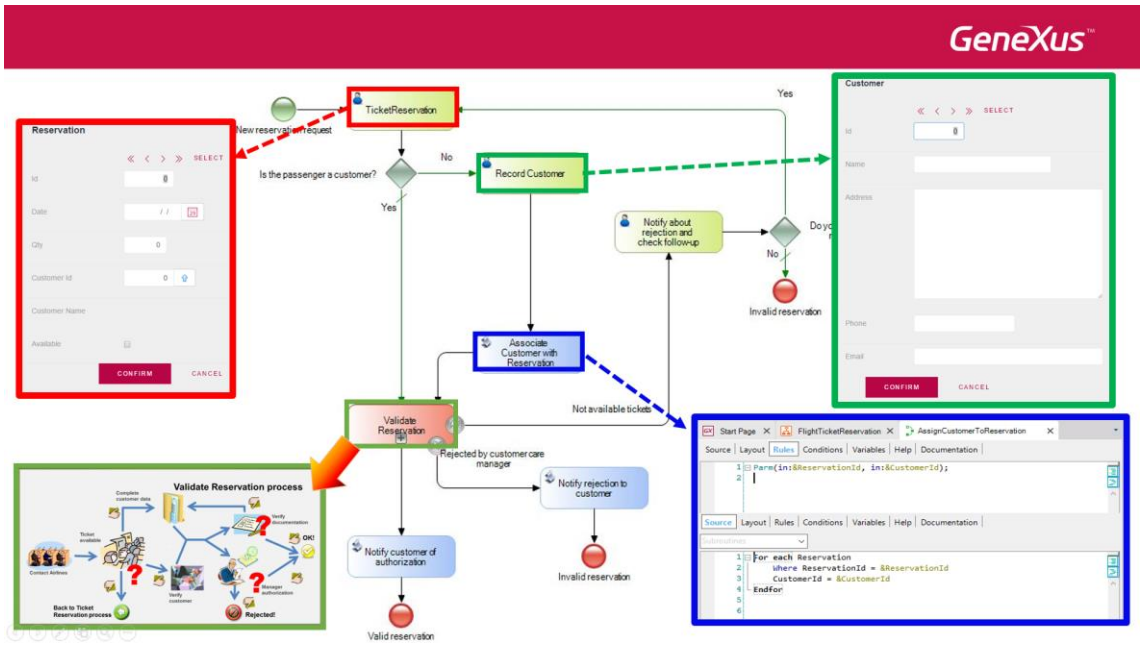
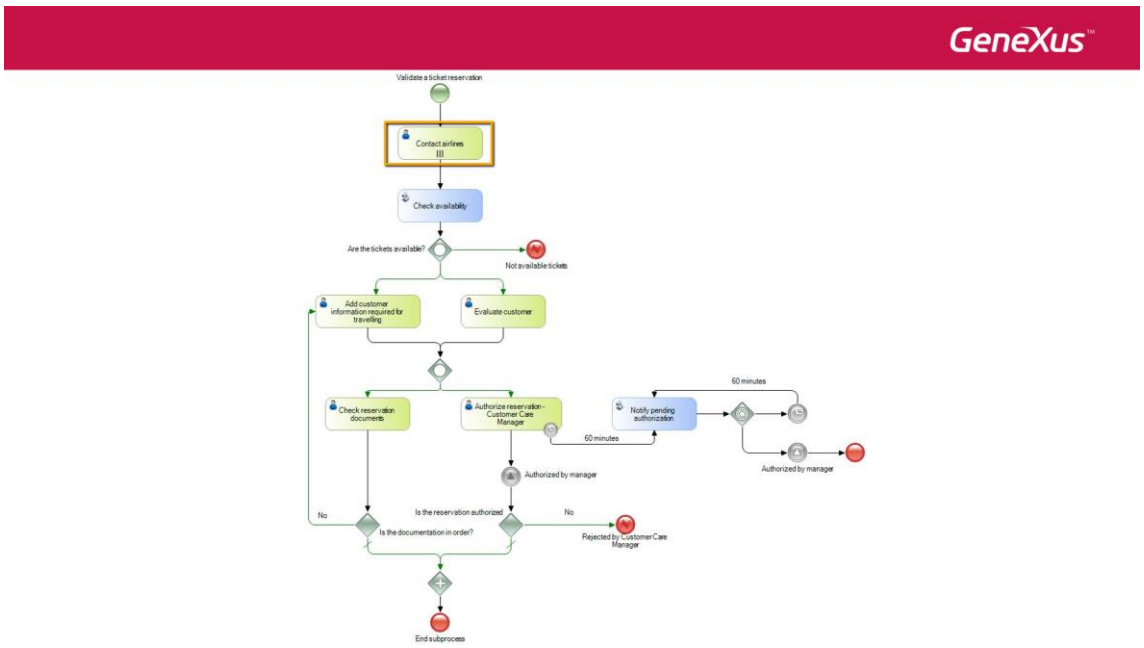
Tarefas multi-instânciadas e mapeamento de dados relevantes

En vídeos anteriores, associamos as tarefas do diagrama de reserva de passagens da Agência de Viagens a objetos GeneXus, convertendo o modelo do processo en uma aplicação funcional.



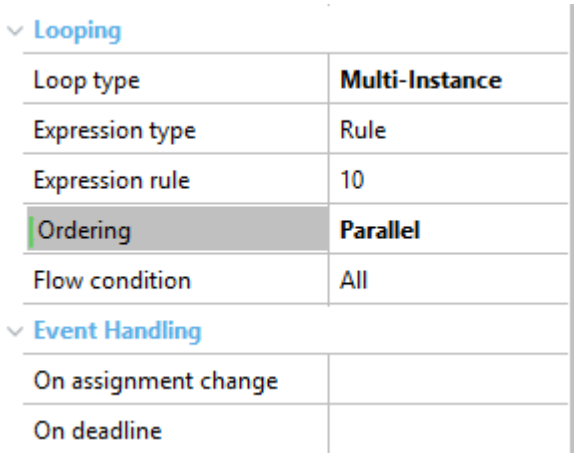
Continuaremos fazendo o mesmo com o diagrama de validação da reserva, subprocesso do processo de reserva de passagens.

Se analizamos o diagrama ValidateReservation, vemos que a primeira tarefa que se executará é a de contactar companhias aéreas.



Esta tarefa tem a particularidade de que será executada um certo número de vezes, pois é necessário contactar várias companhias aéreas, inclusive sendo realizado simultaneamente, por vários usuários diferentes.

Se vamos a suas propriedades, vemos que a propriedade **Loop type** tem o valor Multi-Instance, a propriedade **Ordering** tem o valor Parallel, a propriedade **Expression type** tem o valor Rule e a propriedade **Expression rule** tem o valor: 10.

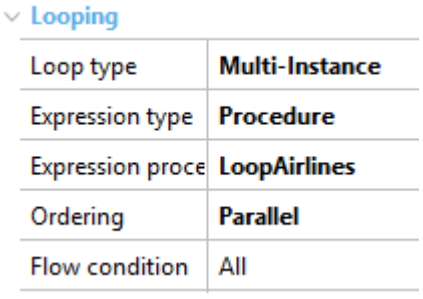


Isto significa que a tarefa **se repetirá exatamente 10 vezes em paralelo**, que foi a ideia definida na etapa de modelagem. Além disto, como a propriedade **Flow Condition** tem o valor All, a tarefa ContactAirlines será terminada somente quando termine de executar as 10 instâncias.

Porém, analizando a tarefa mais profundamente com o pessoal da Agência de Viagens, concluimos que a quantidade de vezes que se tem que executar a tarefa depende da quantidade de companhias aéreas com as que trabalha a agência neste momento e esta quantidade pode variar com o tempo.

Para saber quantas companhias aéreas a Agência tem registradas, podemos utilizar um procedimento que percorra a tabela das companhias aéreas da Agência de Viagens e devolva a quantidade de companhias aéreas registradas.

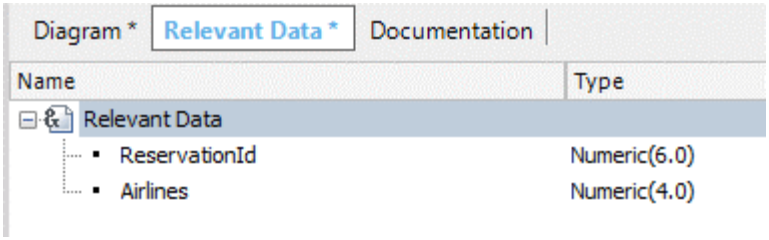
Para implementar isto, mudamos a propriedade **Expression type** a **Procedure** e na propriedade **Expression procedure** selecionamos o procedimento **LoopAirlines**.



Se abrimos o source do procedimento, vemos que tem un For Each que percorre a tabela Airlines e conta a quantidade de companhias aéreas registradas.

```
1  &ArrayOfAirlines = &WorkflowProcessInstance.GetApplicationDataByName("Airlines")
2  &i=0
3  For each // Airlines
4  Defined by AirlineName
5      &i =&i + 1
6      &ArrayOfAirlines.SetValue(&i, AirlineId.ToString())
7  endfor
8
9  &numberofinstances = &i
10
```

Além disto, carrega os identificadores das companhias aéreas em um array, que foi definido como dado relevante do diagrama ValidateReservation.



A forma que se acessa este dado relevante no procedimento é utilizando métodos da API do motor de Workflow. Veremos isto en detalhes mais adiante, em outro vídeo.

```
1  &ArrayOfAirlines = &WorkflowProcessInstance.GetApplicationDataByName("Airlines")
2  &i=0
3  For each // Airlines
4  Defined by AirlineName
5      &i =&i + 1
6      &ArrayOfAirlines.SetValue(&i, AirlineId.ToString())
7  endfor
8
9  &numberofinstances = &i
10
```

Como mencionamos antes, a quantidade de companhias aéreas determina a quantidade de instâncias que serão criadas da tarefa ContactAirlines, sendo que este valor é devolvido pelo procedimento à tarefa ContactAirlines, no último parâmetro da regra Parm.

```
1  parm(in:&WorkflowProcessDefinition,in:&WorkflowProcessInstance,in:&WorkflowWorkitem,out:&numberofinstances);
```

Resumendo: para definir uma tarefa com múltiplas instâncias, atribuimos à propriedade LoopType o valor Multi-Instance e para definir a quantidade de vezes que esta tarefa se instancía, usamos a propriedade Expression type em Rule e atribuimos a quantidade na propriedade Expression Rule ou atribuimos a propriedade Expression Type em Procedure e utilizamos um procedimento que devolve a quantidade de vezes que se instânciará a tarefa, como vimos neste último caso.

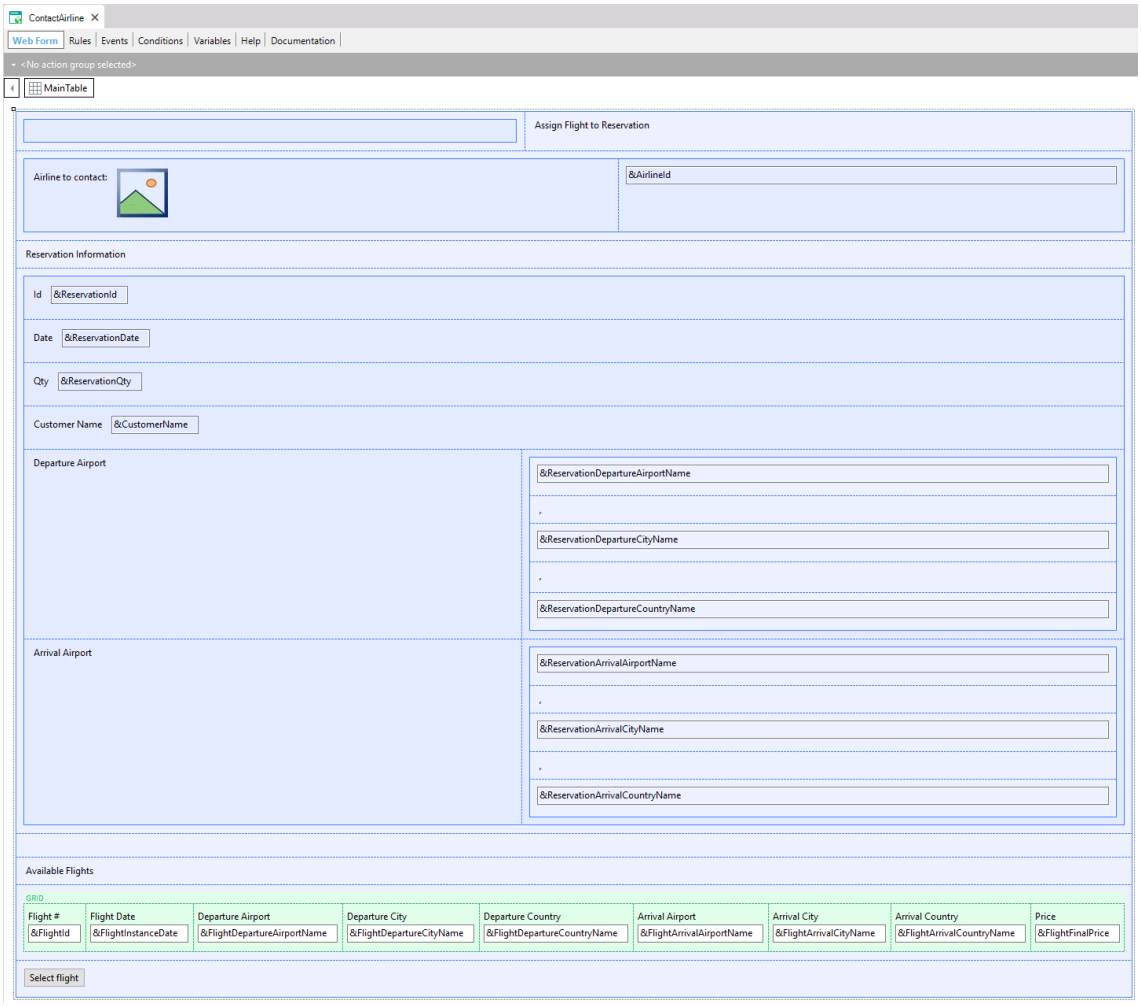
Voltando ao diagrama, a tarefa ContactAirlines terá associado um objeto GeneXus do tipo webpanel, que será executado a cada vez que se executa a tarefa. Seu nome é ContactAirline. Esta webpanel permitirá escolher para cada companhia, o voo adequado para a reserva.

Task: Contact airlines

Name	**Contact airlines**
Type	**User**
Application	ContactAirline
SD Application	(none)
Visible in history	True

Ao iniciar a execução da webpanel, a instância da tarefa em execução será associada internamente a uma das companhias aéreas, de forma que cada vez que inicie uma nova instância da tarefa, esta se conectará com uma companhia diferente, entre as que estão registradas pela agência.

A webpanel mostra os dados da reserva e os voos que a companhia selecionada tem disponíveis para a data da reserva.

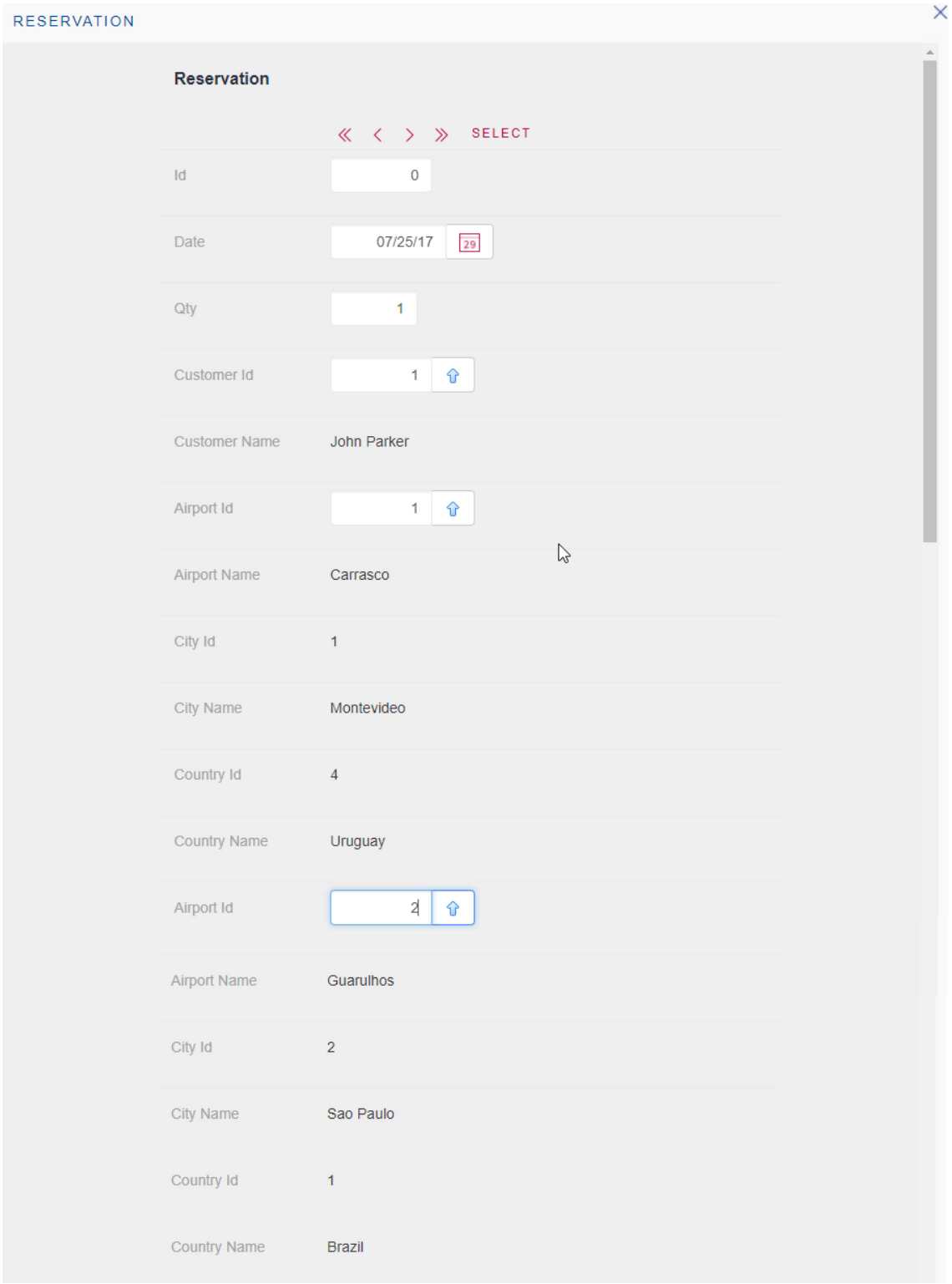


O operador da agência poderá selecionar o voo que deseja associar à reserva.

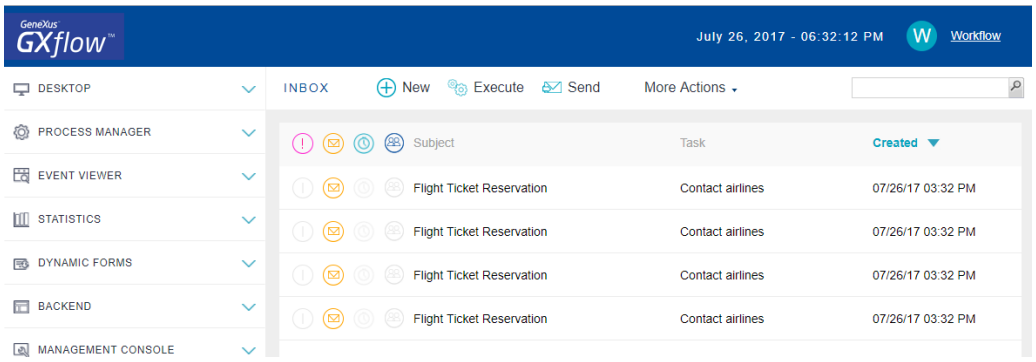
Vamos ver isto em execução.

Na aba do diagrama FlightTicketReservation, clicamos com o botão direito e marcamos Run.

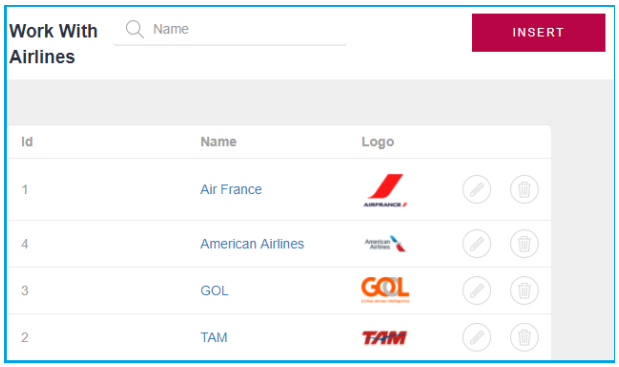
Executamos a tarefa TicketReservation e ingresamos uma reserva para hoje, para o cliente 1, John Parker, que deseja ir do aeroporto de Carrasco, em Montevideu, até o aeroporto de Guarulhos, em São Paulo. Pressionamos Confirmar e fechamos a tela.



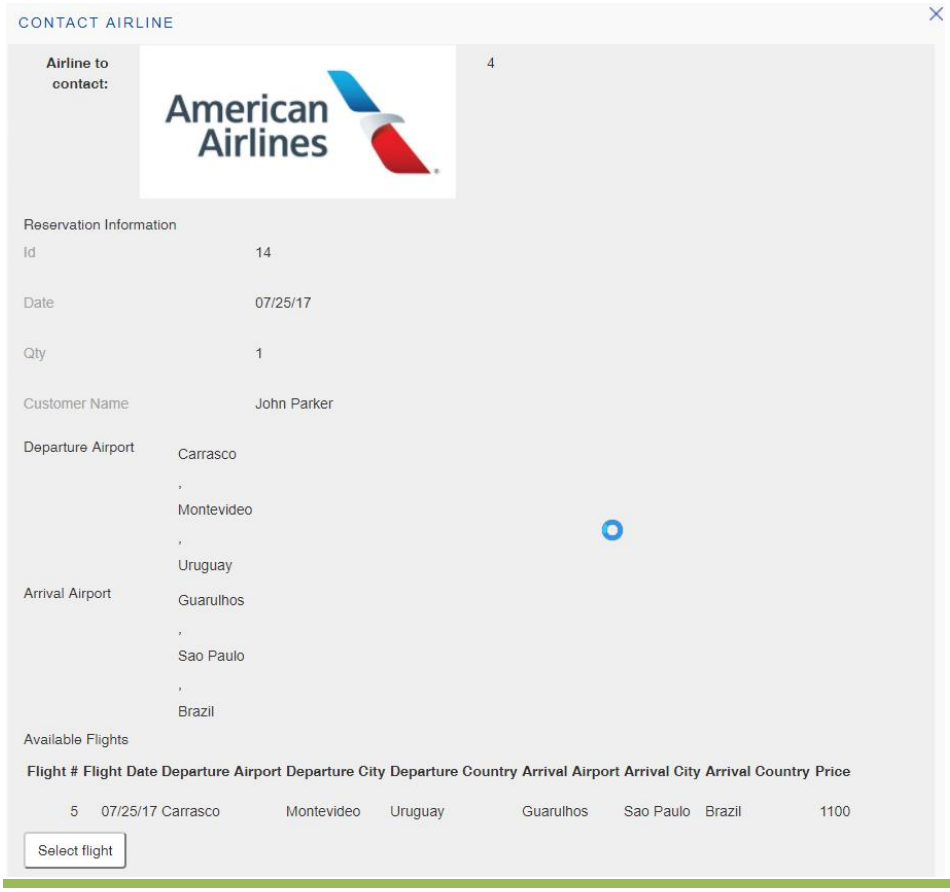
Pressionamos Send para enviar a tarefa e vemos que aparecem quatro tarefas ContactAirlines pendentes.



Isto é porque temos 4 companhias aéreas para contactar e foi criada uma instância da tarefa ContactArlines para cada uma das companhias registradas na Agência.

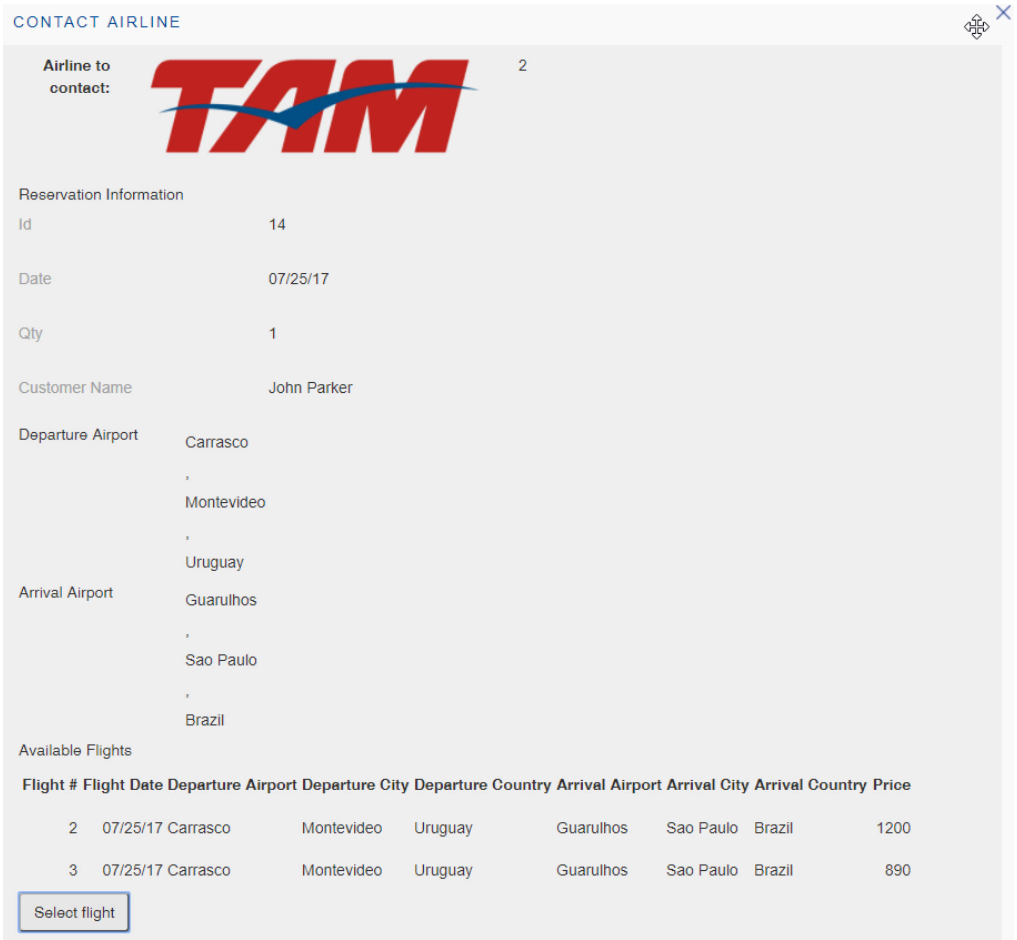


Se fizermos um duplo clique na primeira tarefa pendente, é aberta uma tela para contactar a primeira companhia. Vemos que existe um voo disponível para o dia, origem e destino requeridos pela reserva, assim selecionamos o voo e pressionamos Select Flight.



Desta forma atribuimos um possível voo, que cumpre com a reserva solicitada.

Fechamos a janela e completamos a tarefa, a qual desaparece do inbox como tarefa pendente. Se executamos a próxima tarefa, vemos que é atribuida uma companhia diferente e será assim para cada instância da tarefa ContactAirlines.

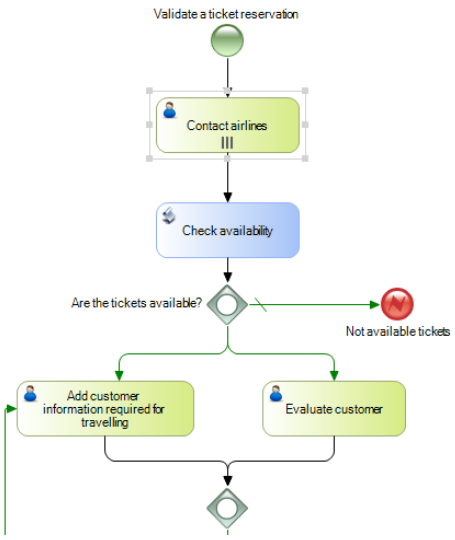


Isto é resolvido no objeto webpanel, já que a partir do dado relevante Airlines (do tipo array) que guarda los identificadores de companhias aéreas, cada vez que inicia a webpanel, é obtido um elemento array diferente para cada instância da tarefa ContactAirlines.

```
Event Start
    &AirlinesWorkflowApplicationData = &WorkflowContext.ProcessInstance.GetApplicationDataByName("Airlines")

    // Get the corresponding Airline from the workitem index
    &i = &WorkflowContext.Workitem.Index
    &AirlineId = &AirlinesWorkflowApplicationData.GetValue(&i).ToNumeric()
    For each
        Where AirlineId=&AirlineId
        &AirlineName = AirlineName.Trim()
        &AirlineLogo = AirlineLogo
    Endfor
Endevent
```

Continuando com o processo de validação da reserva, após contactar as companhias aéreas, é verificado mediante a tarefa **CheckAvailability** que seja possível encontrar ao menos um voo que satisfaça a reserva.



Se abrimos o procedimento **CheckReservationFlights** e formos ao source, vemos que existe um For Each que percorre a tabela de detalhes da reserva e verifica que haja pelo menos um voo selecionado para a reserva. Em caso afirmativo, atribui o valor True à variável &ReservationAvailable.

```
1  //Check if the reservation has at list one assigned flight
2  &ReservationAvailable = False
3  For each // RESERVATIONDETAIL
4       Defined by ReservationDetailSelected
5       &ReservationAvailable = True
6       Exit
7  Endfor
8
```

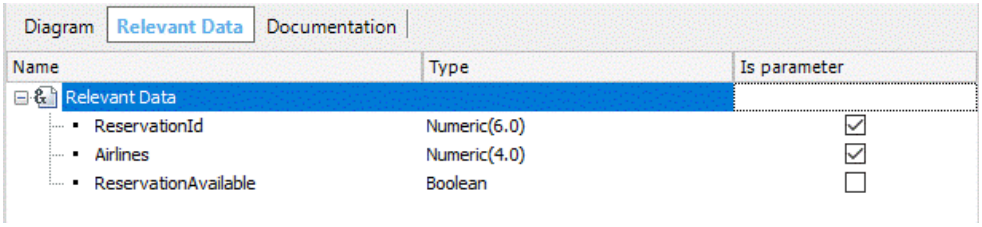
Esta variável retorna como último parâmetro da regra Parm do procedimento.

```
1  Parm(in:&ReservationId, out:&ReservationAvailable);
2
```

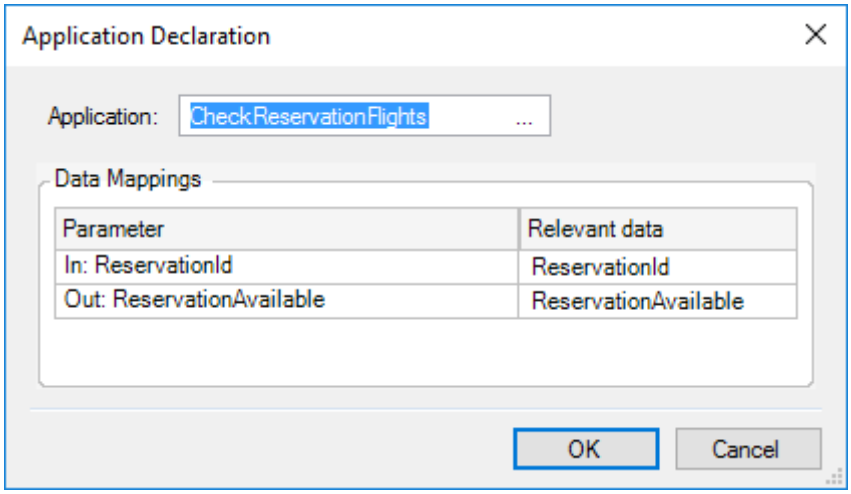
Se a esta variável chamamos exatamente igual que um dado relevante, o motor de workflow carregará automaticamente o dado relevante com o valor da variável.

Nos objetos procedure, o mapeamento de valores entre os dados relevantes e as variáveis presentes na regra Parm, é válido tanto para variáveis de entrada como de saída, enquanto no caso de objetos webpanels, o mapeamento dos valores somente é válido para variáveis de entrada.

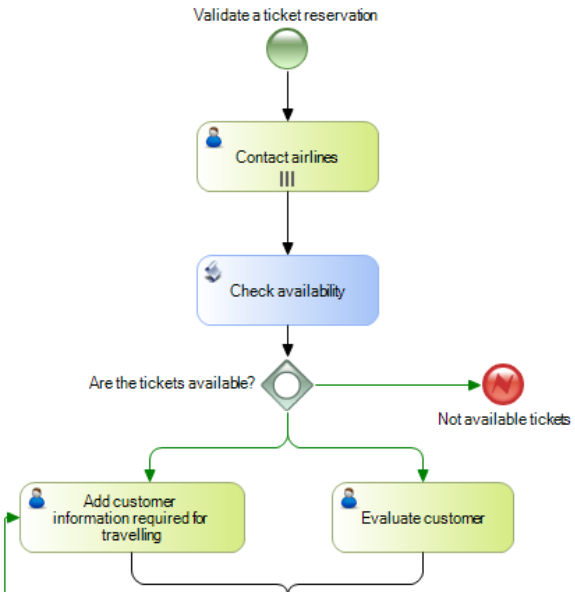
Assim que voltamos ao diagrama ValidateReservation, selecionamos a aba RelevantData e criamos o dado relevante **&ReservationAvailable** do tipo boolean e desmarcamos o check box "IsParameter" porque este dado não é um parâmetro do objeto diagrama.



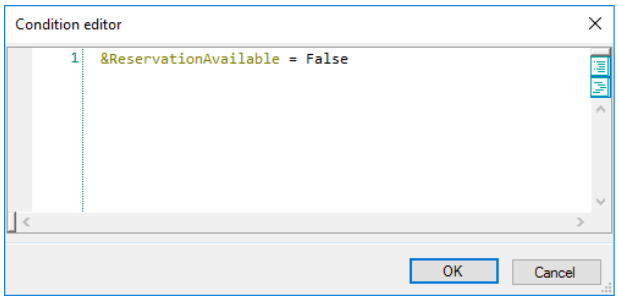
Por último, associamos a procedure **CheckReservationFlights** à tarefa batch **CheckAvailability** e mapeamos os dados relevantes ReservationId e ReservationAvailable.



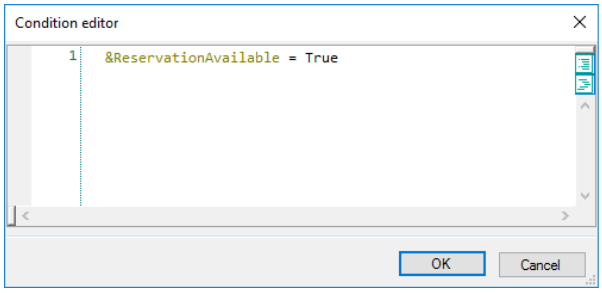
Voltando ao diagrama, uma vez que o procedimento estabelece se a reserva está disponível ou não, o inclusive gateway "Are the tickets available?" devería checar o valor do dado relevante que carregamos.



Para fazê-lo, fazemos duplo clique no conector que sai a direita do inclusive Gateway e escrevemos &ReservationAvailable=False. Na propriedade **Text** escrevemos "Tickets not available".

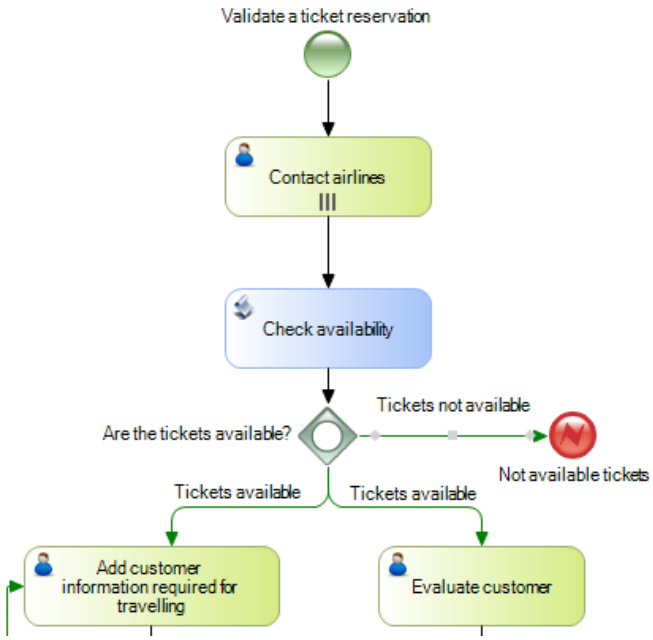


Fazemos o mesmo com os dois conectores que saem para baixo do inclusive Gateway, atribuindo-lhe a condição &ReservationAvailable=True e na propriedade **Text**: "Tickets available".



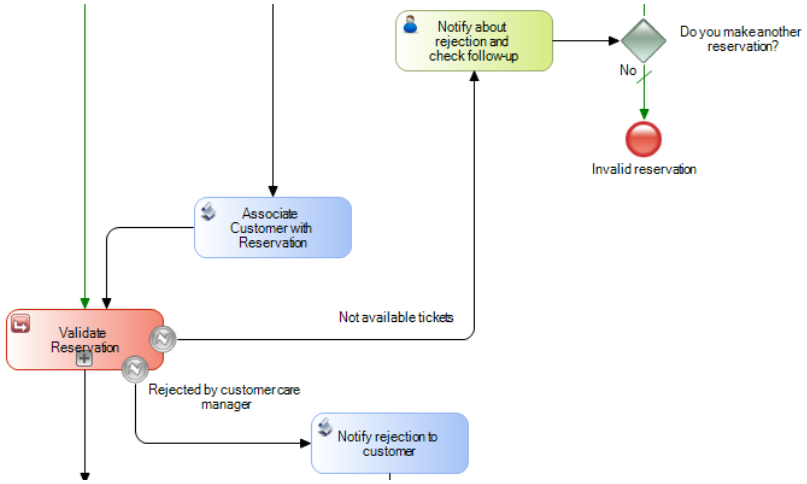
Nas expressões de condição de um Gateway podemos incluir dados relevantes, constantes (como o valor True deste caso), valores de domínios enumerados e atributos da tabela extendida das transações associadas ao diagrama.

Com as definições que fizemos, se houver voos disponíveis para a reserva, o fluxo seguirá para baixo do Gateway e se não houver voos para a reserva, seguirá para a direita, terminando no **Error End Event** chamado "Not available tickets".



Este tipo de evento de finalização com error, nos permite finalizar o subprocesso de validação da reserva e enviar a comunicação do erro ao processo principal de reserva de passagens.

Se observamos o processo principal, vemos que também há um símbolo de evento intermediário de error, com a mesma etiqueta "Not available tickets", que está conectado a uma tarefa interativa onde notifica o cliente da situação.



O evento intermedário de error é do tipo "catch", enquanto o evento de fim de error do subprocesso, é do tipo "throw".

Esta é a forma en que podemos saber no processo principal, qual foi exatamente a causa da finalização do subprocesso e atuar em consequência.

No próximo vídeo continuaremos com o subprocesso de validação da reserva, com as tarefas interativas "Add customer information required for traveling" e "Evaluate Customer" que são executadas simultaneamente.