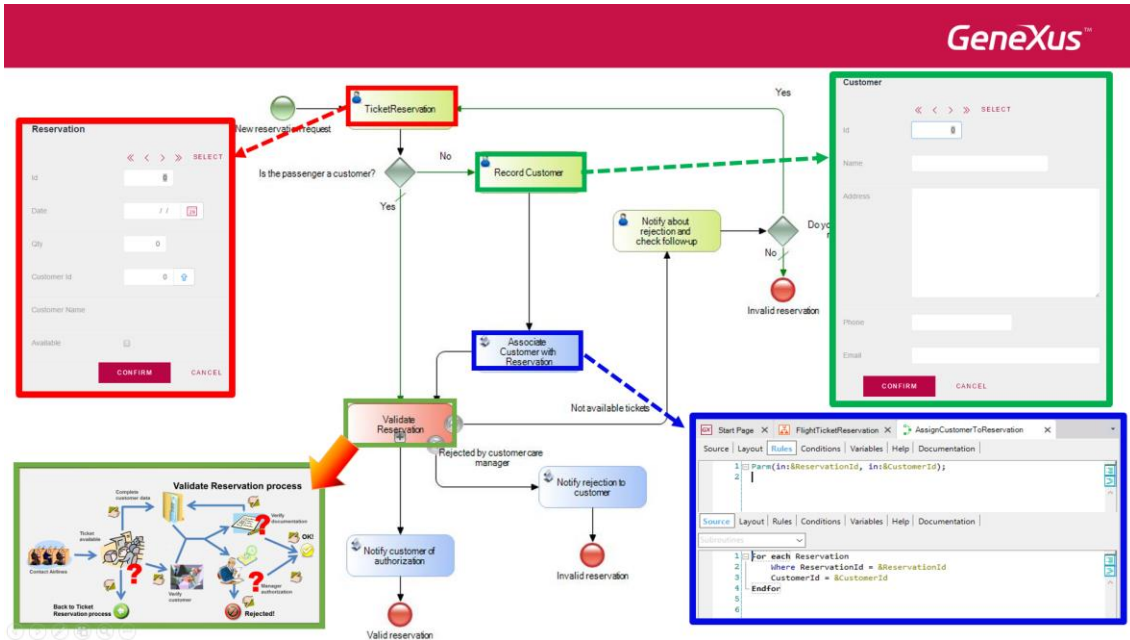


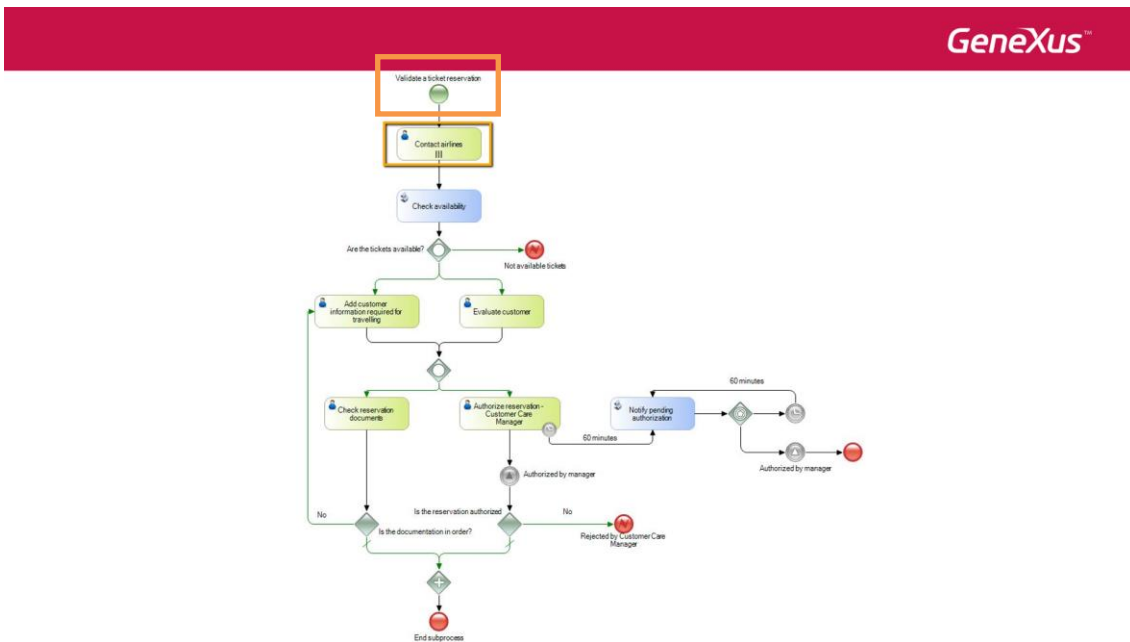
Multi-instanced tasks and mapping of relevant data

In previous videos we have associated the tasks in the Travel Agency's ticket reservation diagram to GeneXus objects, converting the process model in a functional application.



We will continue to do the same with the reservation validation diagram, a subprocess of the ticket reservation process.

If we examine the ValidateReservation diagram, we can see that the first task will be to contact the airlines.



A particular feature of this task is that it will be executed a certain number of times because several airlines have to be contacted. This may also be performed at the same time by different users.

We open its properties and see that the **Loop type** property is set to Multi-Instance, the **Ordering** property is set to Parallel, the **Expression type** property is set to Rule, and the **Expression rule** property has the value 10.

<p>Looping</p>	
Loop type	Multi-Instance
Expression type	Rule
Expression rule	10
Ordering	Parallel
Flow condition	All
<p>Event Handling</p>	
On assignment change	
On deadline	

This means that the task **will be repeated exactly 10 times in parallel**, as it was indicated in the modeling stage. In addition, since the **Flow Condition** property is set to All, the ContactAirlines task will be finished only after the 10 instances have been executed.

However, if we examine the task in more detail with the travel agency’s staff, we realize that the number of times that this task has to be executed depends on the number of airlines that the Agency currently works with. In addition, this number can vary with time.

To find out the number of airlines that the agency has registered, we can use a procedure that runs through the table of the travel agency airlines and returns the number of registered airlines.

To implement this, we change the **Expression type** property to **Procedure** and in the **Expression procedure** property we select the **LoopAirlines** procedure.

<p>Looping</p>	
Loop type	Multi-Instance
Expression type	Procedure
Expression procedure	LoopAirlines
Ordering	Parallel
Flow condition	All

We open the procedure source and see that it has a For Each command that runs through the Airlines table and counts the number of registered airlines.

```

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

```

In addition, it loads the airline identifiers in an array, which was set as a relevant data item of the ValidateReservation diagram.

Diagram *	Relevant Data *	Documentation								
	<table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>Relevant Data</td> <td></td> </tr> <tr> <td>▪ ReservationId</td> <td>Numeric(6.0)</td> </tr> <tr> <td>▪ Airlines</td> <td>Numeric(4.0)</td> </tr> </tbody> </table>	Name	Type	Relevant Data		▪ ReservationId	Numeric(6.0)	▪ Airlines	Numeric(4.0)	
Name	Type									
Relevant Data										
▪ ReservationId	Numeric(6.0)									
▪ Airlines	Numeric(4.0)									

This relevant data item is accessed from the procedure using methods from the Workflow engine API. We will see this in more detail in another video.

```

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

```

As we've mentioned before, the number of airlines determines the number of instances that will be created of the ContactAirlines task, so this value is returned by the procedure to the ContactAirlines task in the last parameter of the Parm rule.

```

1  parm(in:&WorkflowProcessDefinition,in:&WorkflowProcessInstance,in:&WorkflowWorkitem out:&numberofinstances)
2

```

In summary, to define a task with multiple instances, we assign the LoopType property with the Multi-Instance value. To indicate the number of times that this task is instantiated, we use the Expression type property set to Rule and assign the number in the Expression Rule property. Or, we can also set the Expression Type property to Procedure and use a procedure that returns the number of times that the task will be instantiated, as we've seen in this case.

Going back to the diagram, the ContactAirlines task will be associated with a GeneXus object of webpanel type, which will be executed every time that the task is executed. Its name is ContactAirline. This webpanel will allow selecting, for each airline, the most adequate flight for the reservation.

Task: Contact airlines	
Name	Contact airlines
Task Metadata Id	2
Task Metadata GUID	a9d5f688-4fb2-4af1-b2ce-4...
Type	User
Web Object	ContactAirline
Object	(none)

Upon starting the webpanel, the running task instance will be internally associated with one of the airlines, so that every time that a new task instance is started, a different airline from those registered by the agency will be contacted.

The webpanel shows the reservation details and the flights that the selected airline has available for the reservation date.

The screenshot shows a web browser window titled "ContactAirline". The browser's address bar shows "Web Form" and a menu with "Rules", "Events", "Conditions", "Variables", "Help", and "Documentation". Below the browser window, there is a "MainTable" control. The main content area is titled "Assign Flight to Reservation" and contains several sections:

- Airline to contact:** A dropdown menu with a small airplane icon and a text input field labeled "&AirlineId".
- Reservation Information:** A section with several input fields:
 - Id: &ReservationId
 - Date: &ReservationDate
 - City: &ReservationCity
 - Customer Name: &CustomerName
 - Departure Airport: A section with three input fields: &ReservationDepartureAirportName, &ReservationDepartureCityName, and &ReservationDepartureCountryName.
 - Arrival Airport: A section with three input fields: &ReservationArrivalAirportName, &ReservationArrivalCityName, and &ReservationArrivalCountryName.
- Available Flights:** A table with columns for Flight #, Flight Date, Departure Airport, Departure City, Departure Country, Arrival Airport, Arrival City, Arrival Country, and Price. Each column has a corresponding input field with a placeholder like "&FlightInstanceDate". Below the table is a "Select flight" button.

The agency's employee will be able to select the flight that will be associated with the reservation.

Let's see it at runtime.

We right-click on the tab of the FlightTicketReservation diagram and select Run.

We run the TicketReservation task and enter a reservation for today, for customer 1, who is called John Parker and wants to travel from Carrasco airport in Montevideo to Guarulhos airport in Sao Paulo. We click on Confirm and close the screen.

Reservation

Date: 04/14/23

Qty: 2

Customer Id: 1

Customer Name: John Parker

Airport Id: 1

Airport Name: Carrasco

City Id: 1

City Name: Montevideo

Country Id: 1

Country Name: Uruguay

Airport Id: 2

Airport Name: Guarulhos

DELETE CANCEL CONFIRM

We click on Send to send the task and see that four ContactAirlines pending tasks are displayed.

Workflow Administrator

Inbox

HISTORY PREVIEW DELEGATE COLLABORATE MORE ACTIONS - NEW













Subject	Activity	State	Created	EXECUTE	SEND
Flight Ticket Reservation	Contact airlines	ready	04/14/23 07:33 AM	EXECUTE	SEND
Flight Ticket Reservation	Contact airlines	ready	04/14/23 07:33 AM	EXECUTE	SEND
Flight Ticket Reservation	Contact airlines	ready	04/14/23 07:33 AM	EXECUTE	SEND
Flight Ticket Reservation	Contact airlines	ready	04/14/23 07:33 AM	EXECUTE	SEND

The reason is that we have to contact 4 airlines, and one instance of the ContactAirlines task has been created for each one of the airlines registered at the Agency.

Work With Airlines

INSERT

Q Name

Id	Name	Logo		
1	Air France			
2	American Airlines			
3	GOL			
4	TAM			

Click on the first pending task opens a screen to contact the first airline. It has a flight available for the date, origin and destination required for the reservation, so we select the flight and click on Select Flight.

Contact Airline

×

Airline to
contact:

2



Reservation Information

Id 52

Date 04/14/23

Qty 2

Customer Name John Parker

Departure Airport Carrasco
,
Montevideo
,
Uruguay

Arrival Airport Guarulhos
,
Sao Paulo
,
Brazil

Available Flights

Flight #	Flight Date	Departure Airport	Departure City	Departure Country	Arrival Airport	Arrival City	Arrival Country	Price
1	04/14/23	Carrasco	Montevideo	Uruguay	Guarulhos	Sao Paulo	Brazil	1100

Select flight

In this way, we assign a possible flight that meets the requirements of the reservation requested.

We close the window and complete the task, so it is no longer displayed as a pending task in the inbox. We run the next task and see that a different airline is assigned. This will happen for every instance of the ContactAirlines task.

Contact Airline

×

Airline to contact:

1



Reservation Information

Id 52

Date 04/14/23

Qty 2

Customer Name John Parker

Departure Airport Carrasco
,
Montevideo
,
Uruguay

Arrival Airport Guarulhos
,
Sao Paulo
,
Brazil

Available Flights

Flight #	Flight Date	Departure Airport	Departure City	Departure Country	Arrival Airport	Arrival City	Arrival Country	Price
----------	-------------	-------------------	----------------	-------------------	-----------------	--------------	-----------------	-------

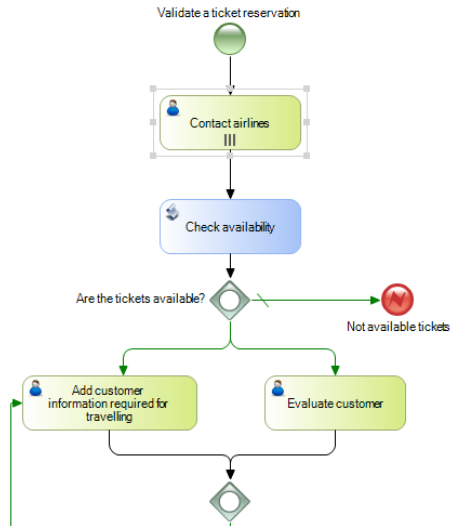
This is solved in the webpanel object, because based on the Airlines relevant data item (of array type) that stores the airline identifiers, a different array element is obtained for each instance of the ContactAirlines task every time that the webpanel is started.

```

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

```

In the following step of the reservation validation process, after contacting the airlines, the **CheckAvailability** task is used to check that at least one flight has been found that meets the reservation's requirements.



We open the **CheckReservationFlights** procedure and see that the source has a For Each command that runs through the reservation details table and checks that at least one flight has been selected for the reservation. If a flight has been selected, it assigns the True value to the &ReservationAvailable variable.

```

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

```

This variable is returned as the last parameter of the procedure's Parm rule.

```

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

```

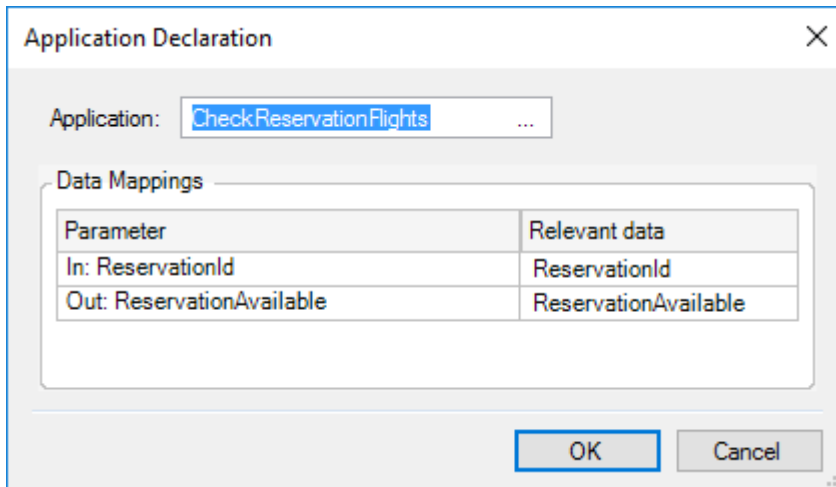
If we give this variable the same name as that of a relevant data item, the workflow engine will automatically load the relevant data item with the variable value.

In procedure objects, mapping values between the relevant data and variables included in the Parm rule is valid for both input and output variables. On the other hand, in webpanel objects the mapping of values is only valid for input variables.

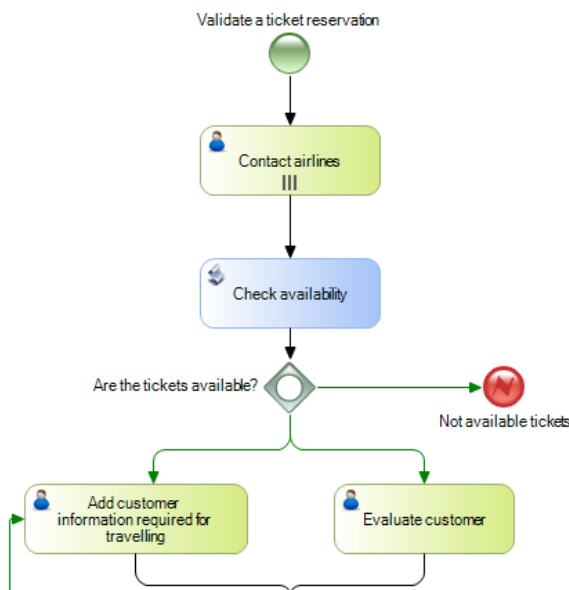
So, we open the ValidateReservation diagram, select the RelevantData tab and create the relevant data item called **&ReservationAvailable** of boolean type, and clear the "IsParameter" check box because this data is not a parameter of the diagram object.

Name	Type	Is parameter
Relevant Data		
▪ ReservationId	Numeric(6.0)	<input checked="" type="checkbox"/>
▪ Airlines	Numeric(4.0)	<input checked="" type="checkbox"/>
▪ ReservationAvailable	Boolean	<input type="checkbox"/>

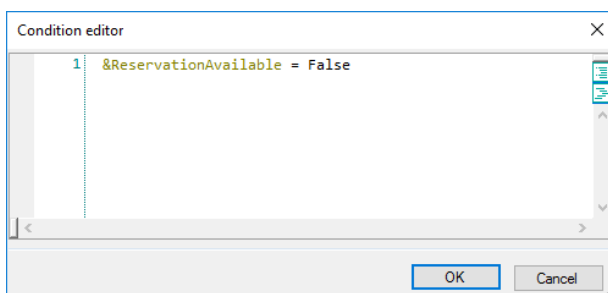
Lastly, we associate the **CheckReservationFlights** procedure with the **CheckAvailability** batch task and map the relevant data ReservationId and ReservationAvailable.



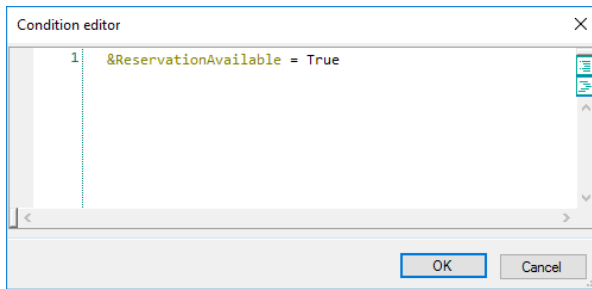
In the diagram, once the procedure sets whether the reservation is available or not, the inclusive gateway "Are the tickets available?" should check the value of the relevant data item we loaded.



To do so, we double-click on the outbound connector on the right side of the inclusive Gateway and type `&ReservationAvailable=False`. In the **Text** property we type "No tickets available".

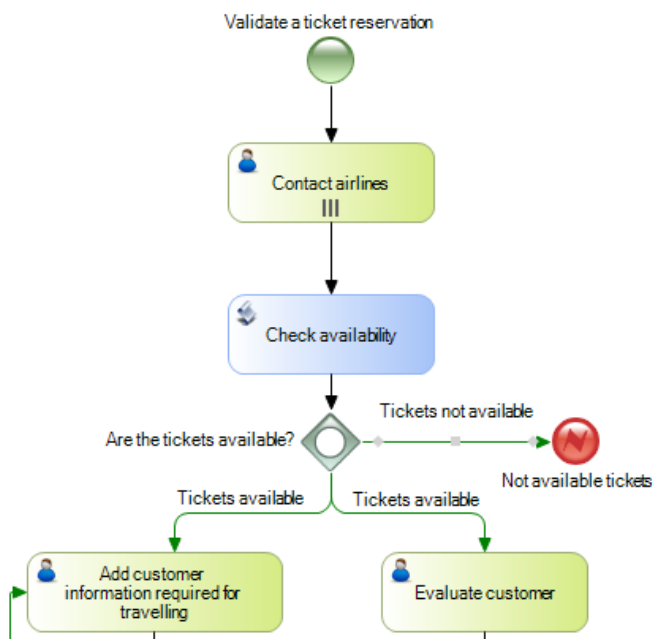


We do the same with the two connectors coming down from the inclusive Gateway, assigning them the condition `&ReservationAvailable=True`, and in the **Text** property: "Tickets available".



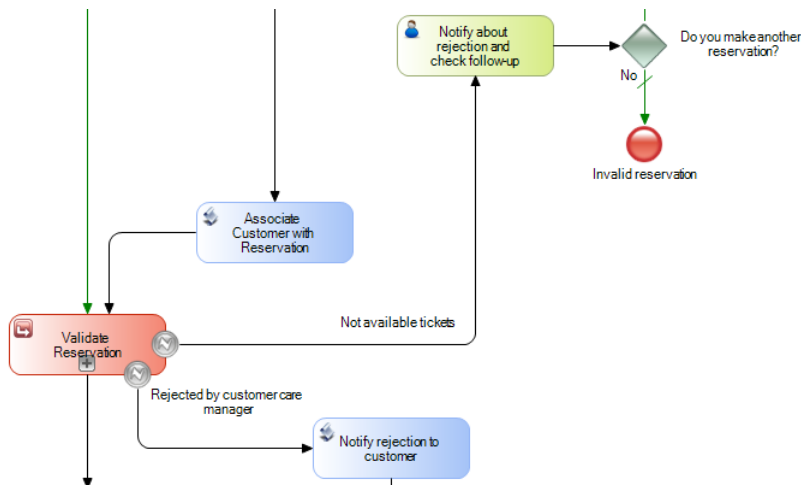
In the condition expressions of a Gateway we can include relevant data, constants (such as the True value in this case), enumerated domain values and attributes of the extended table of transactions associated with the diagram.

Thanks to the definitions we've made, if there are flights available for the reservation, the flow will move down from the Gateway. If there are no flights for the reservation, it will move to the right, ending in the **Error End Event** called "No tickets available".



This type of event that ends with an error allows us to end the reservation validation subprocess and send the error notification to the main ticket reservation process.

If we look at the main process, we can see that it also has the symbol of an intermediate error event with the same tag "No tickets available", which is connected to an interactive task that informs the customer about the problem.



The error intermediate event is of “catch” type, whereas the subprocess error end event is of “throw” type.

In this way, from the main process we can find the reason why the subprocess ended and take the corresponding action.

In the next video we will continue to work with the reservation validation subprocess, using the interactive tasks “Add customer information required for traveling” and “Evaluate Customer” that will be run at the same time.