Introduction to business process modeling based on the BPMN standard.

The operation of a company such as a Travel Agency involves carrying out proceedings and running processes which are specific to the business; for instance, booking airline tickets or hiring tourist services for travelers.

Each one of these **business processes** is run in a certain order and with the purpose of creating a service or product. The administration of these business processes is called **BPM** - Business Process Management.

GeneXus provides tools for modeling, managing and running business processes, such as the Business Process Modeler (as a standalone tool) and the object editor Business Process Diagram, which is integrated into the GeneXus development environment.

GeneXus

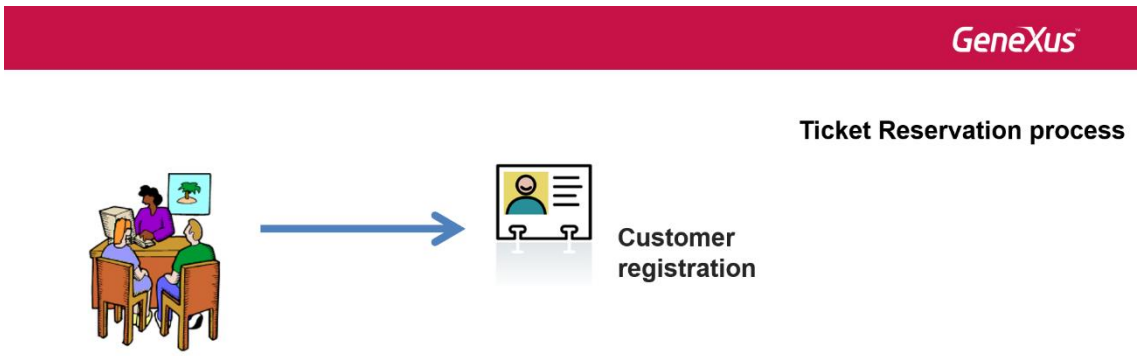
Business
Process
Management

Let's see an example.
The travel agency has asked us to record the process of booking airline tickets.

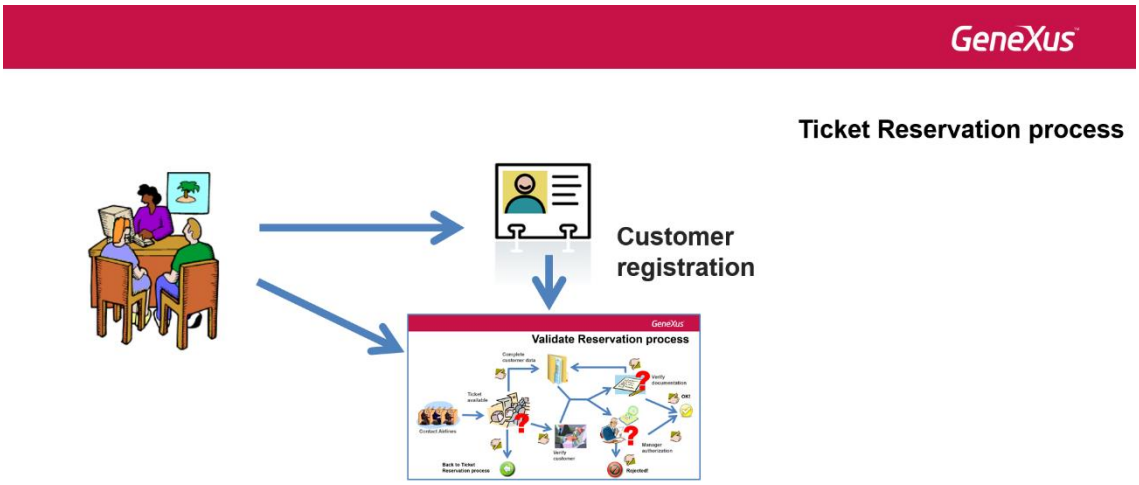
GeneXus

Ticket Reservation process

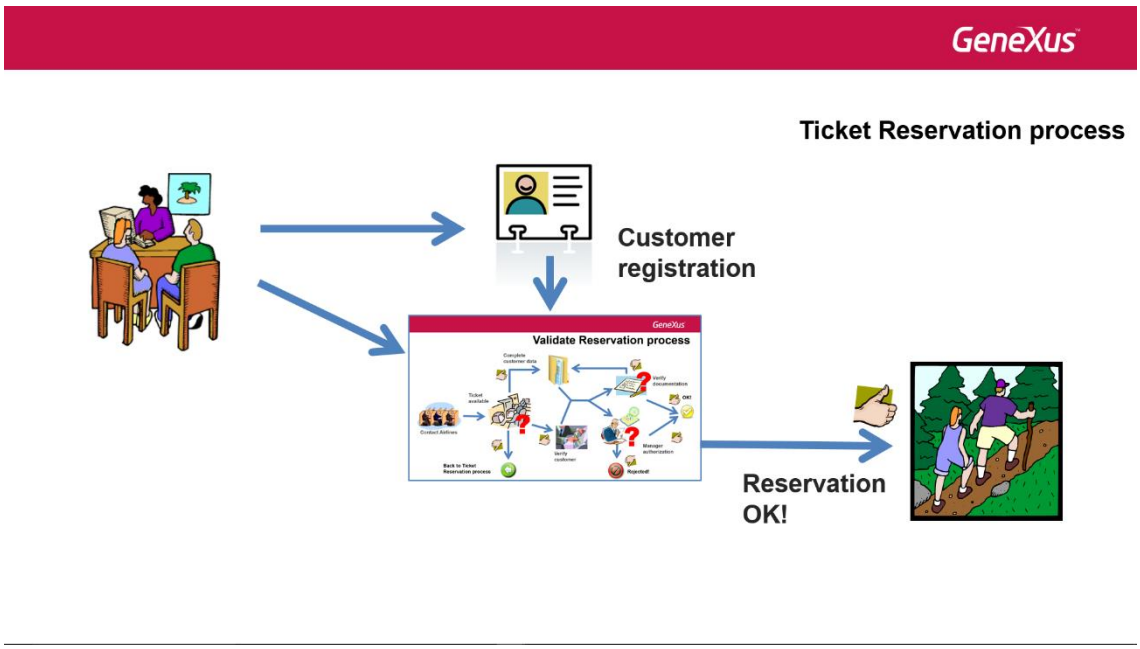
This process begins when we record the reservation details as indicated by the traveler; for instance, where he wants to go, the date and time of departure, departure airport, arrival airport, etc.



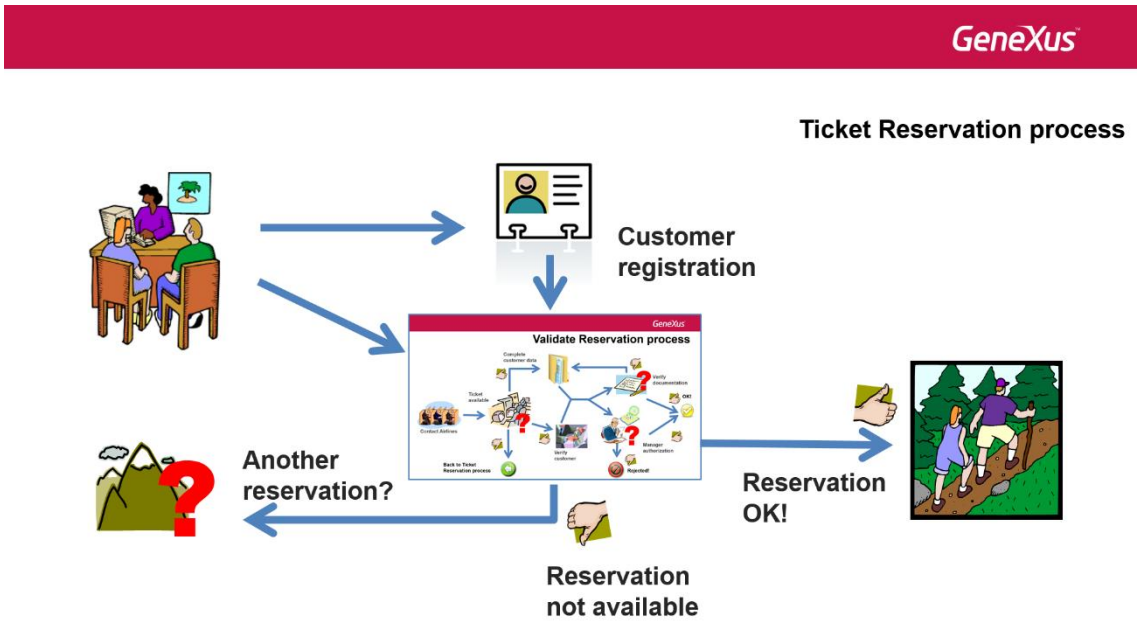
The agency's employee must check if the traveler is a customer of the agency; if he isn't a customer, he must register him.



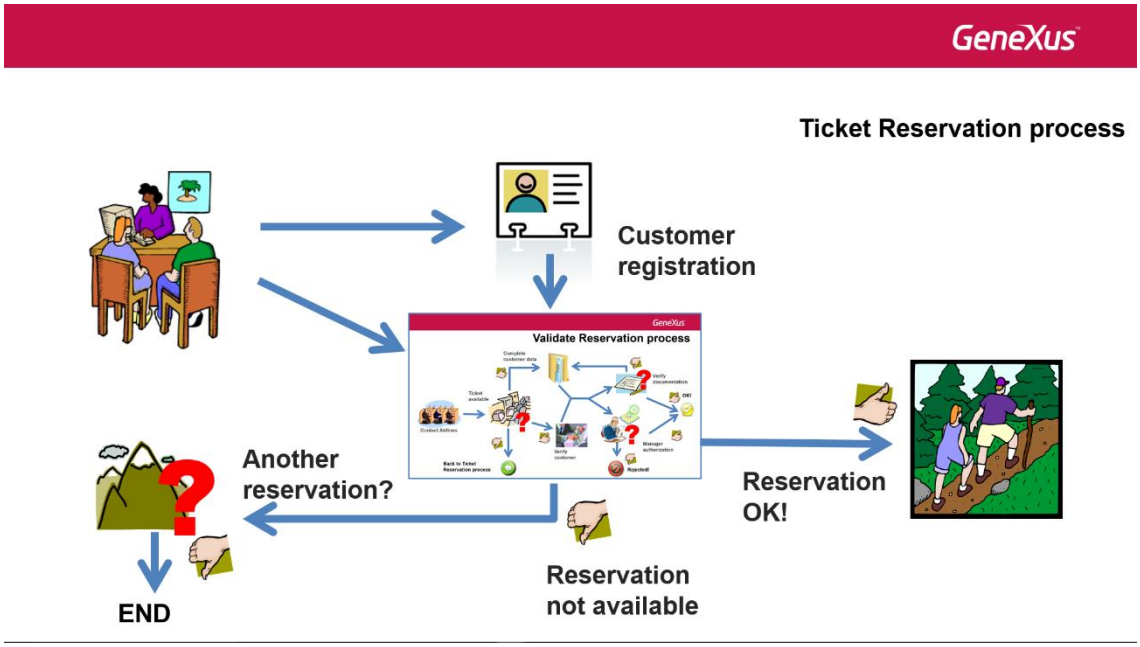
The details of this reservation must be validated in either case. That is to say, if the traveler was an existing customer or if he has been registered as a new customer.
This is done in a sub process known as Validate Reservation, which we will see later.



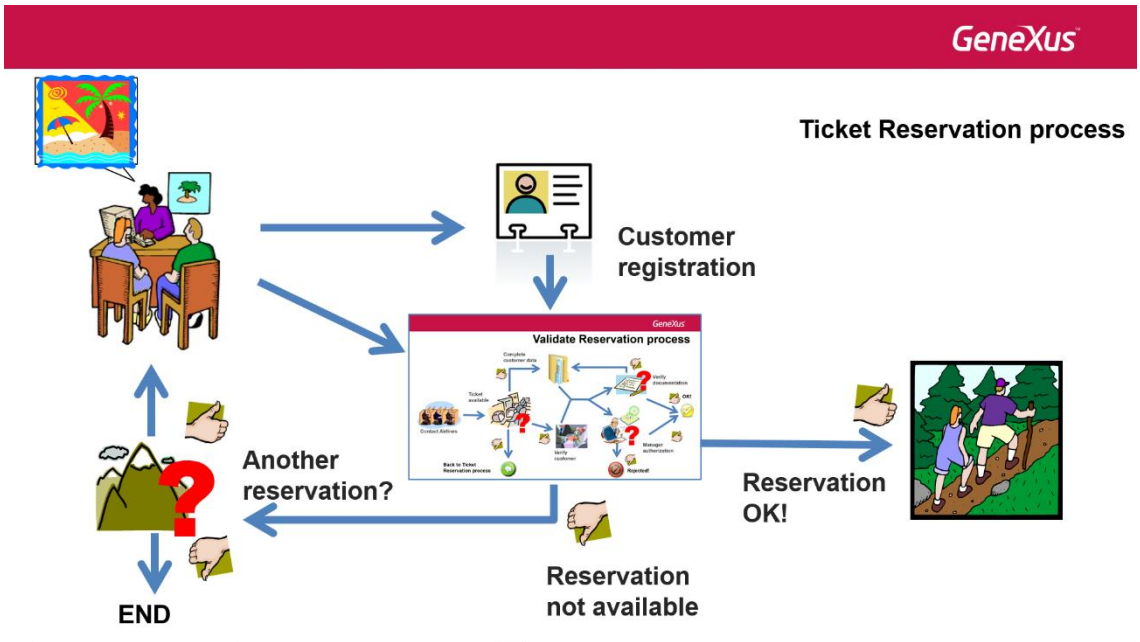
If everything is OK after validating the reservation, the traveler can make the trip.



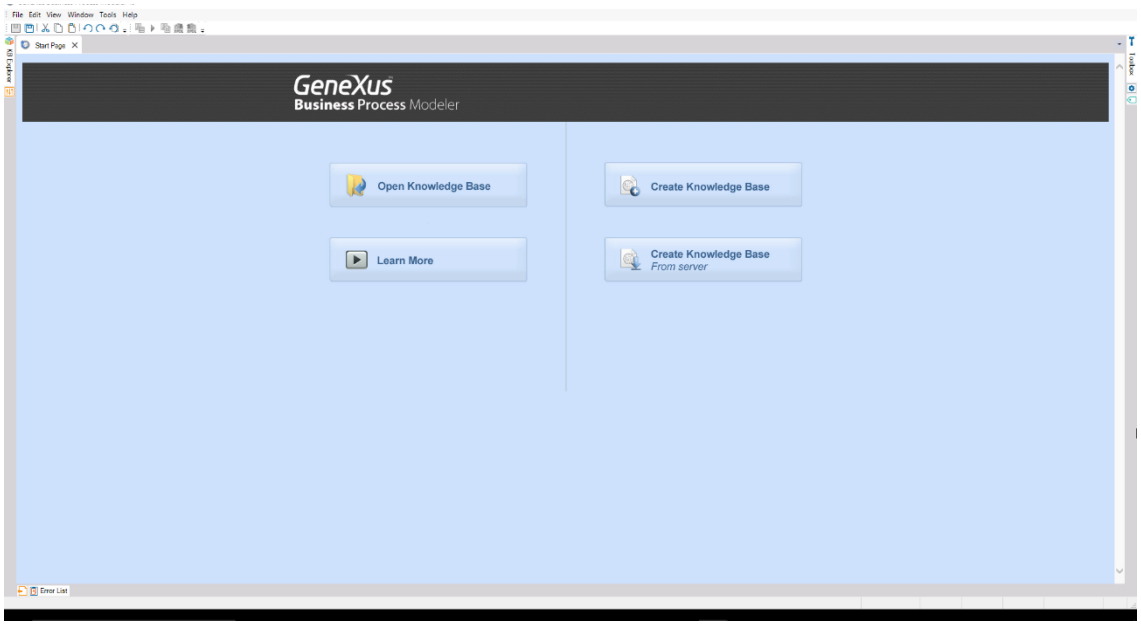
Or, it may also happen that there are no flights available to make the reservation. In this case, the traveler will be asked if he wants to make a different reservation.



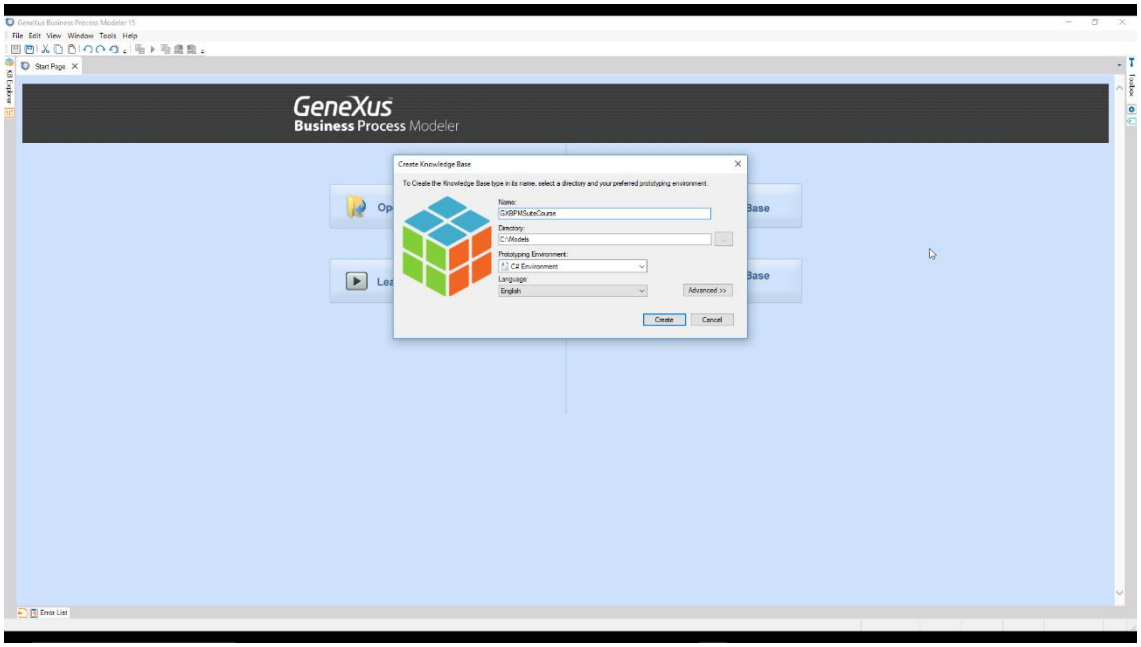
If he declines the offer, the process ends...
Otherwise, a new reservation will be offered and the process starts again.



Let's run the Business Process Modeler to see how to model this task flow.
The first thing we need to do is create a new project, so we press the button labeled Create Knowledge Base.

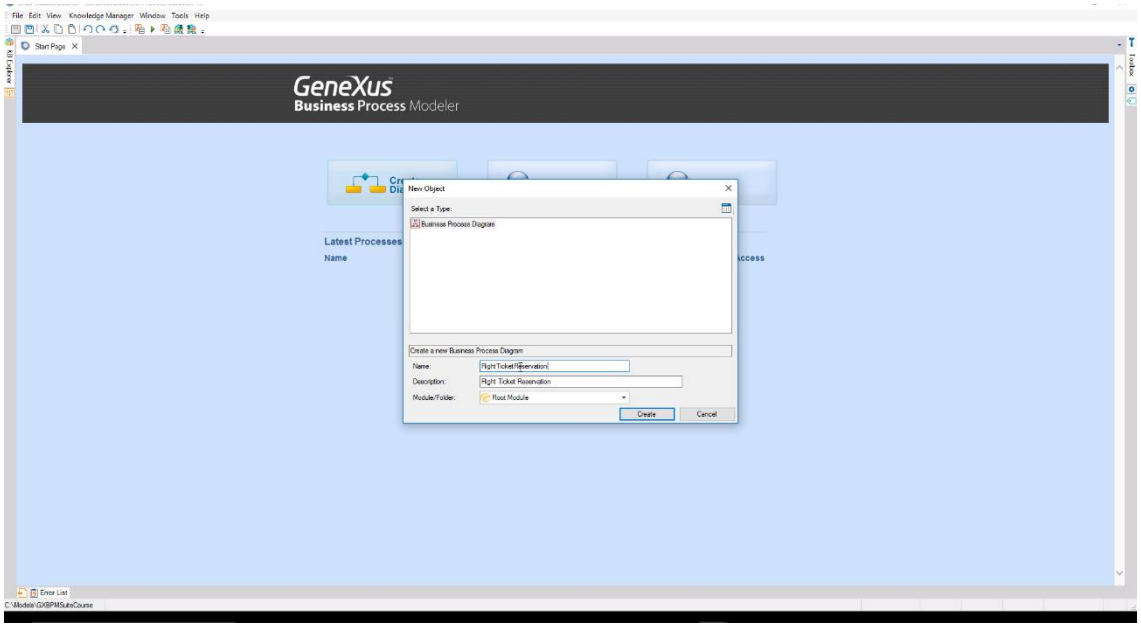


We call it GXBMPSuiteCourse and click on Create.



Note that the development environment is opened, showing a start page.

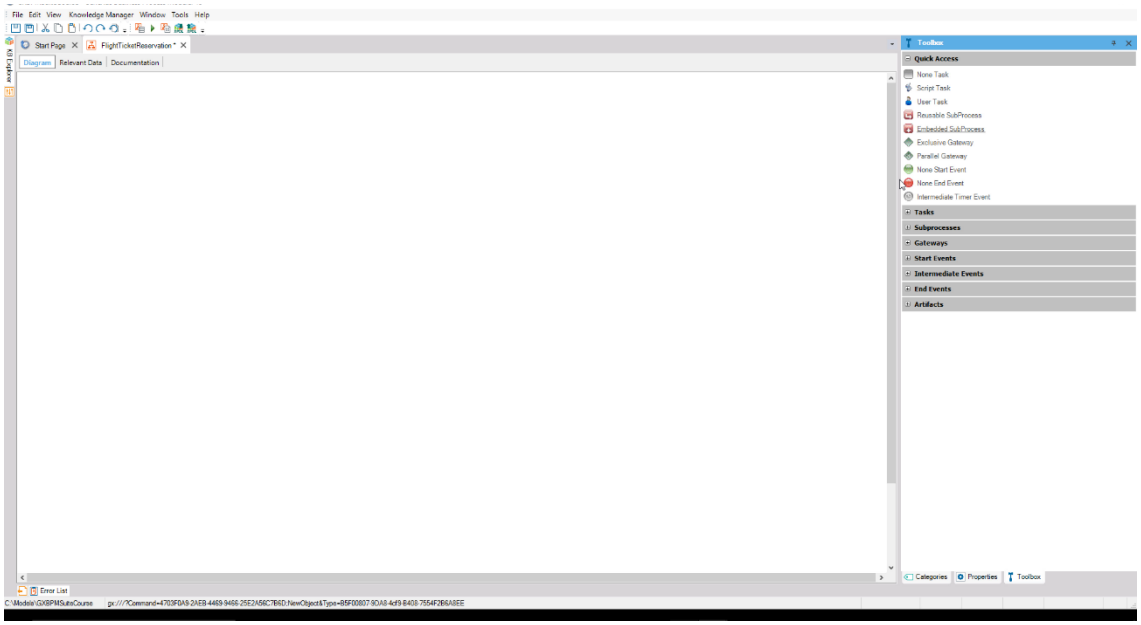
To represent the process, we will create an object of Business Process Diagram type,



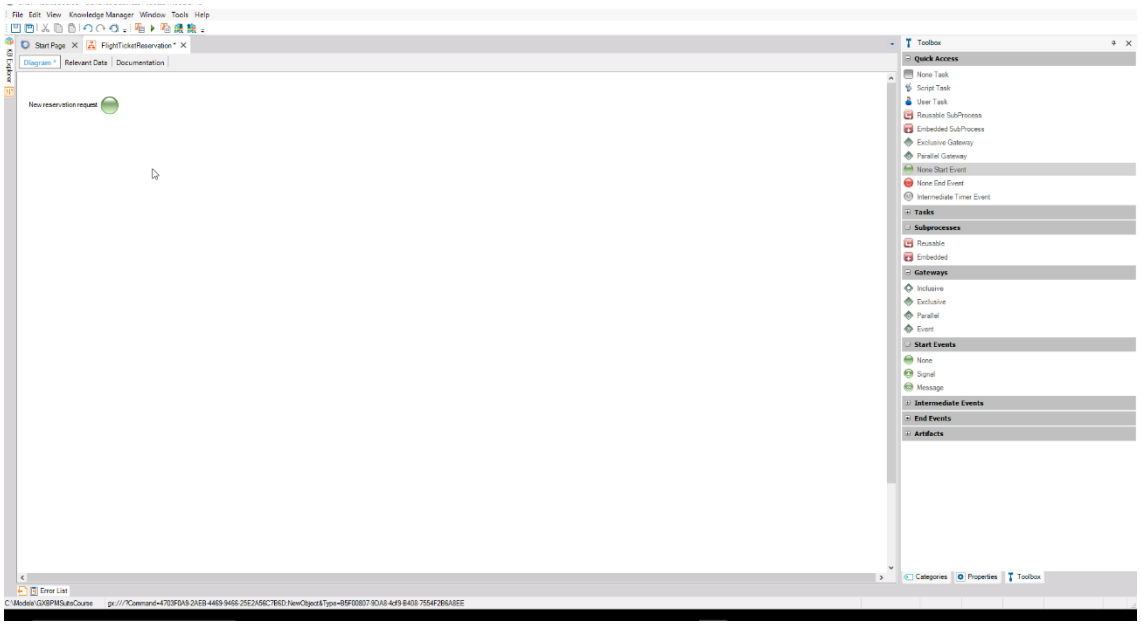
and name it FlightTicketReservation.

Note that we're shown a blank page where we can create our model. We position the mouse cursor on the Toolbok option located in the upper right corner of the screen and click on the pin to fix this window.

In the toolbox we can see a group of symbols that we can use to model our business process, following the international standard known as BPMN or Business Process Model and Notation.

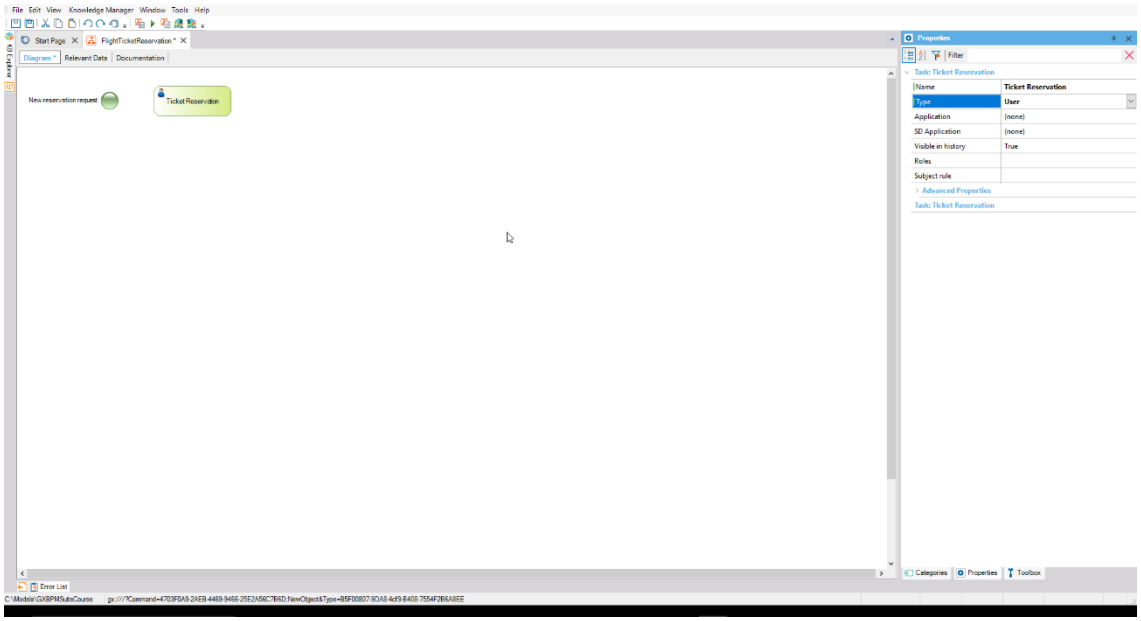


To mark the beginning of the process, we drag a None Start Event symbol. If we click on the + sign on Start, we can see that several symbols are available to start a process; we will talk about them later.

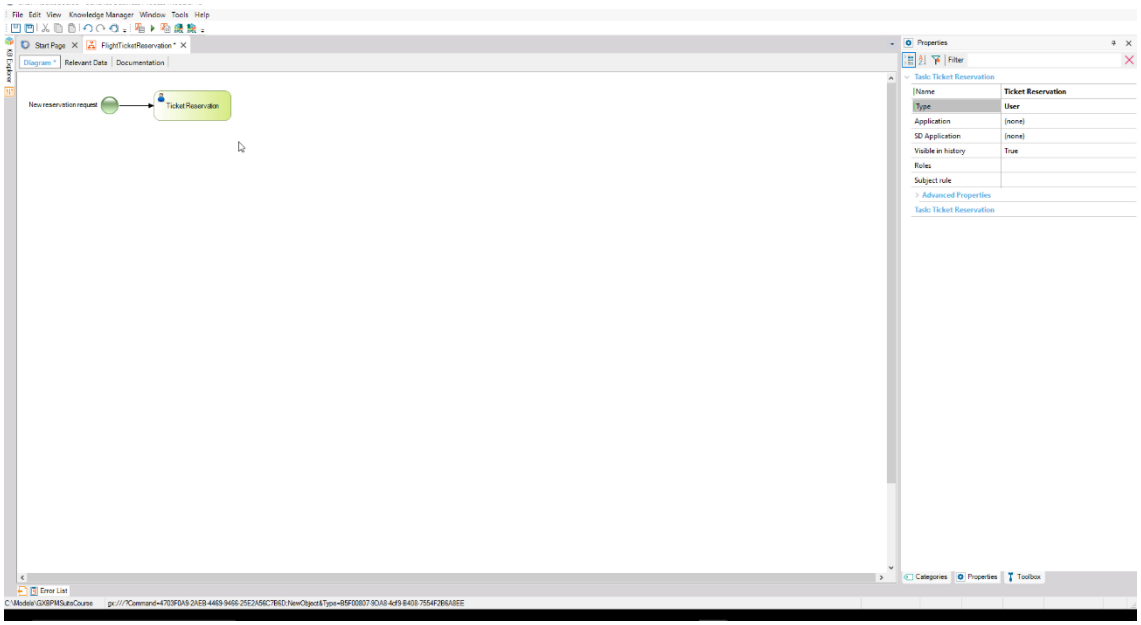


We will add a tag to the start node. We double-click on it, press F2 and type: “New reservation request”.
To book tickets, first we need to perform the task of recording the flight. To model a task, we drag a None Task symbol and name it TicketReservation.

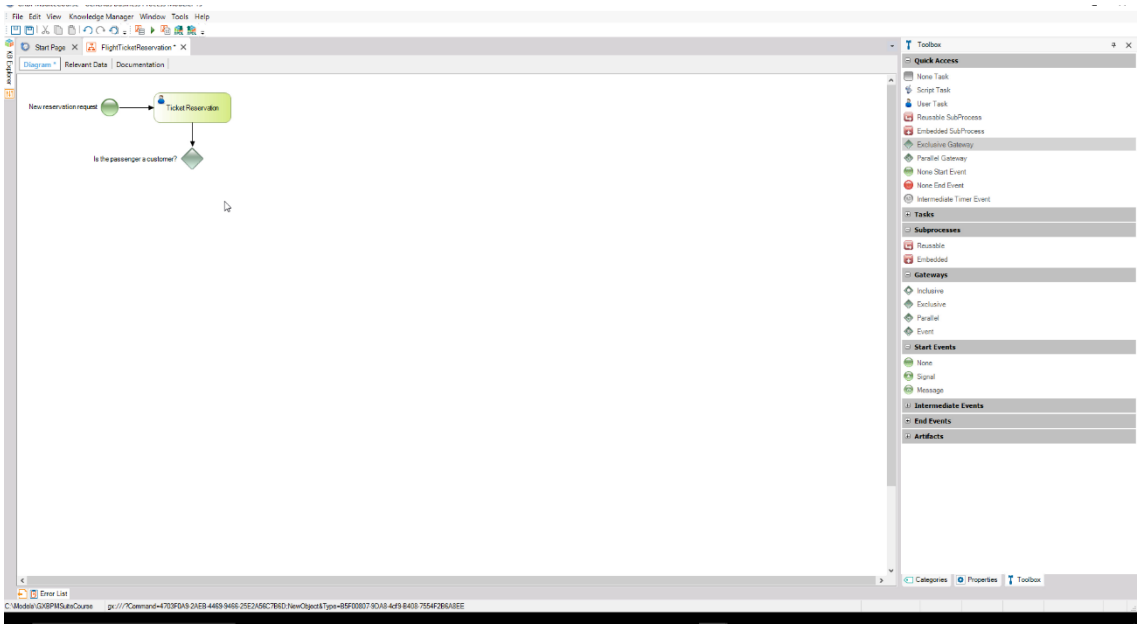
Since we know that the task must be performed by a person, we can assign its type by changing the Type property to the User value. Now this task is an interactive task; that is to say, human intervention will be needed to complete it.



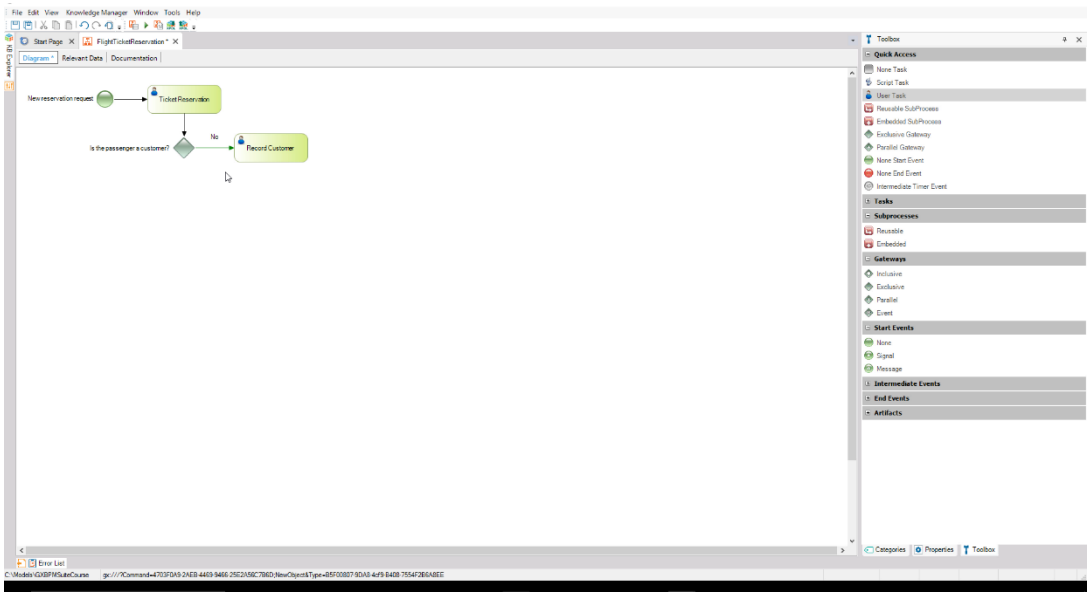
To connect the Start node to the task, we click on the right side of the green circle, hold and drag it until the pointer touches the left side of the task rectangle.



According to the process followed at the travel agency, the system must check if the person making the reservation is an existing customer of the company.

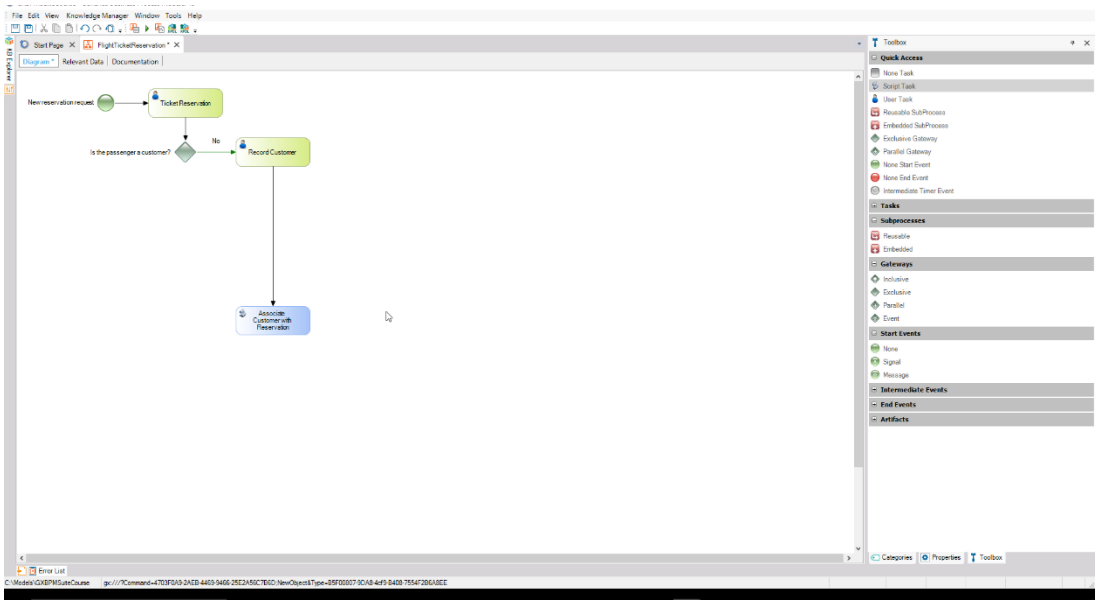


To represent a **decision** in the diagram we open the toolbox, drag an **Exclusive Gateway** node to the diagram and connect it from the TicketReservation task.

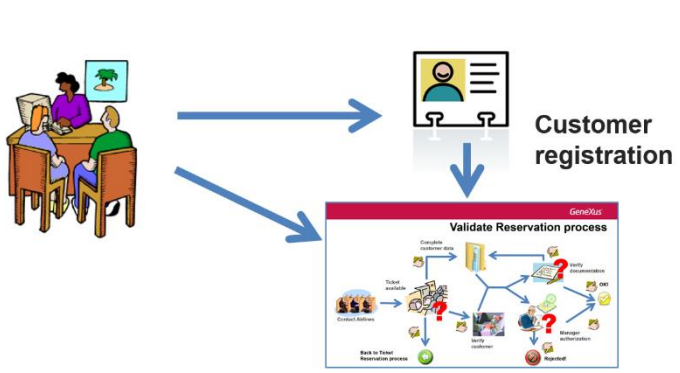


A node of this type evaluates a condition and allows flow branching into several paths. The condition's question may have several answers, but since the node is of "Exclusive" type the flow will only continue through a single path.

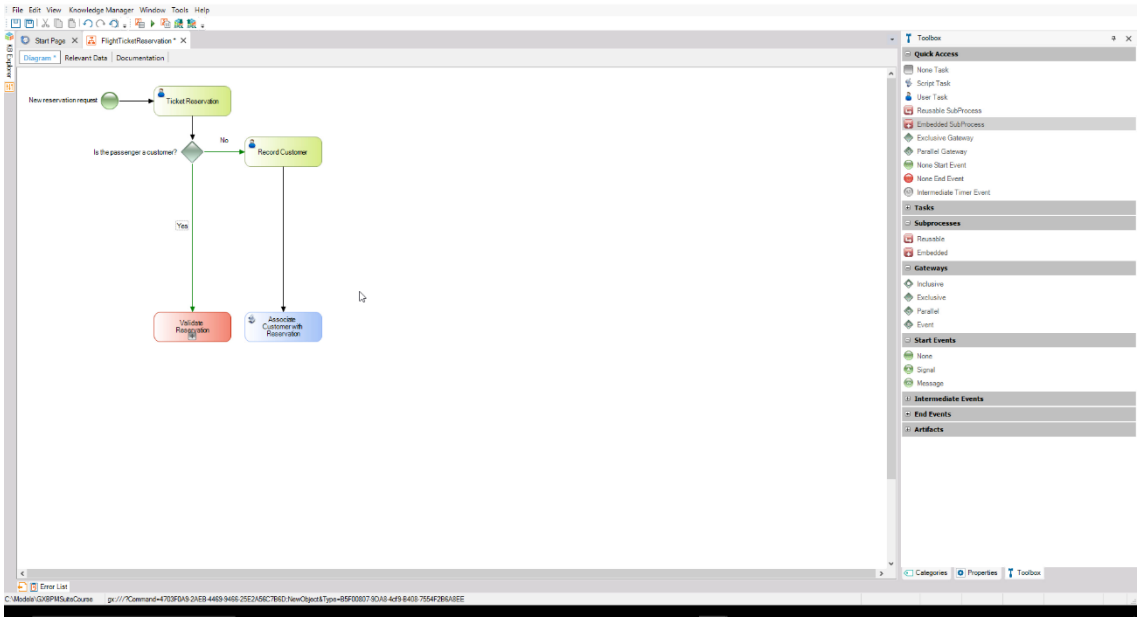
We will add a tag that clarifies the decision we're making. To do so, we select the Gateway, press F2 and type: "Is the traveler an existing customer?"



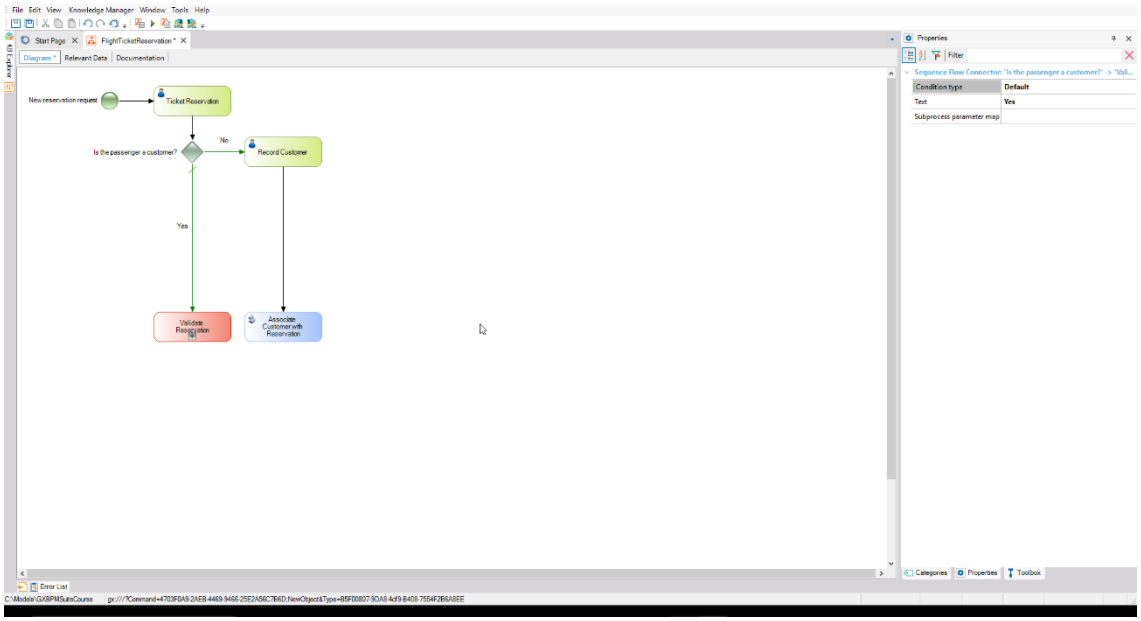
In our example, if the person who wants to book the tourist package **is not** a customer of the company, an agency employee must be directed by the application to add him as a customer.



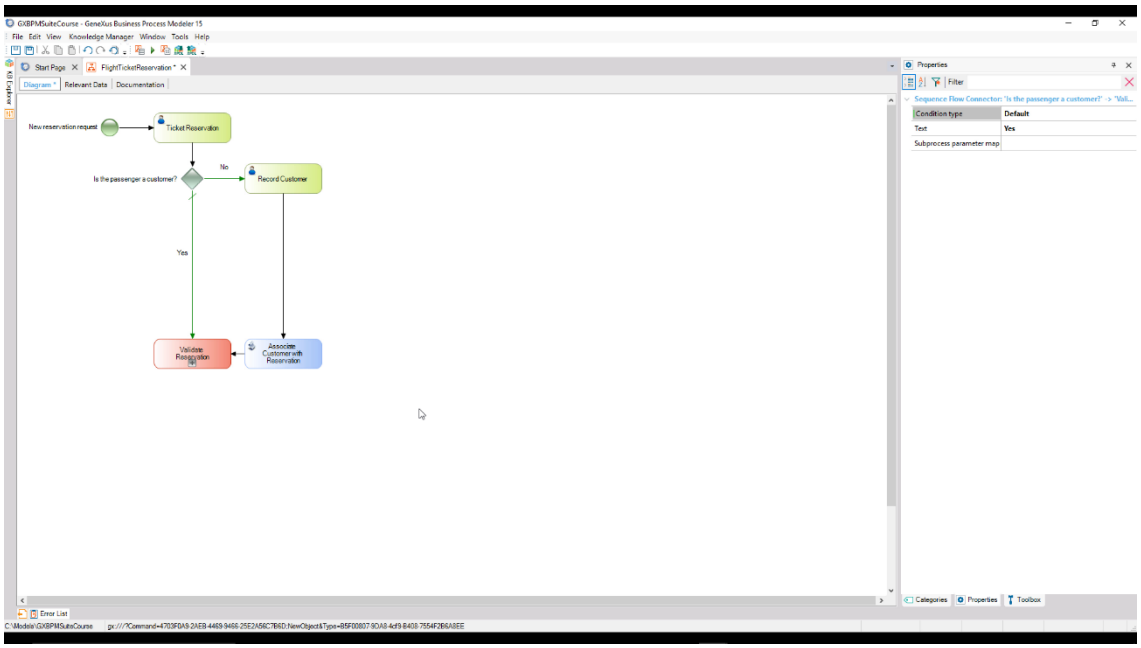
Since it is an interactive task that needs human intervention for its completion, we drag a task of User type from the toolbar, name it Record Customer and connect it from the right side of the Gateway symbol.



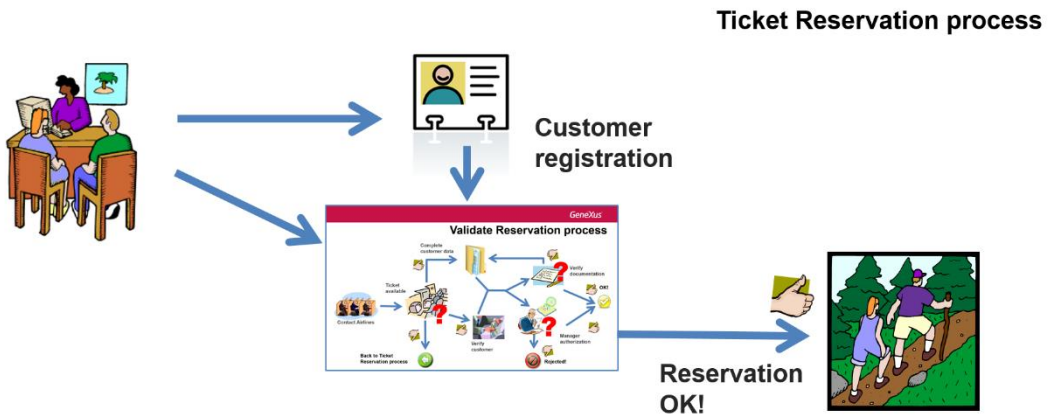
To indicate that this task will be executed if the condition is not met, we add the text “No” to the connector that joins the Gateway to the task.



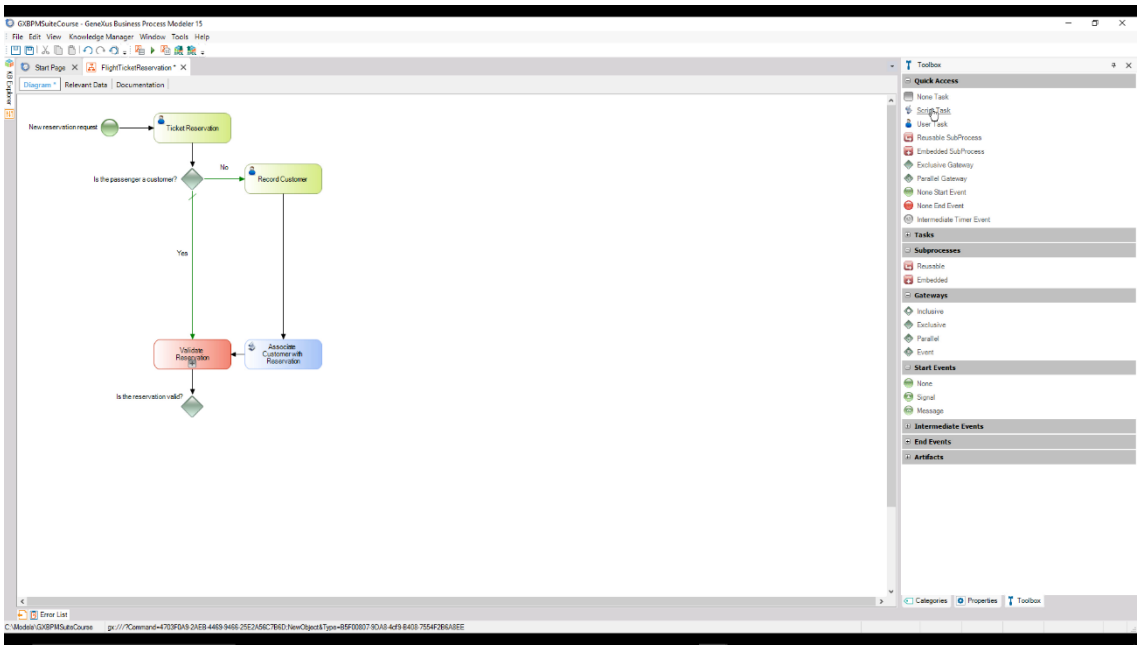
Once a person is added as a customer, the new customer must be associated with the reservation. This task can be automatically performed without human intervention, so we drag a task of Script (or **batch**) type, name it “Associate Customer with Reservation” and connect it from the Record Customer task.



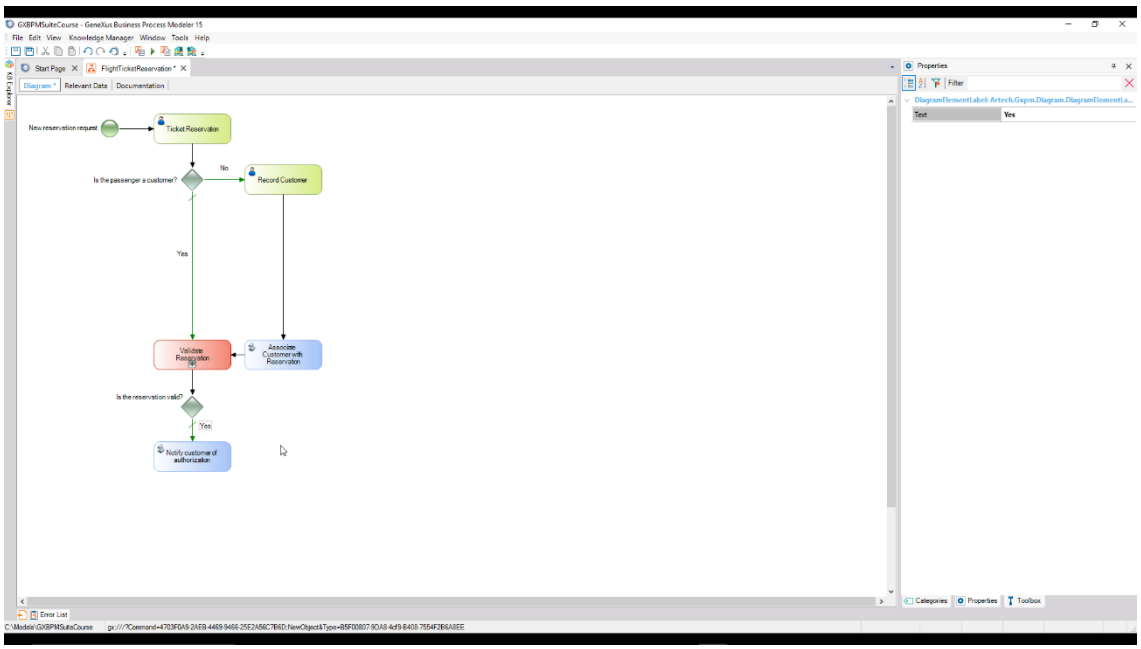
So far, we have described the tasks that will be performed if a new customer has to be added and assigned to the reservation. Now we will add the activity that will be performed if the latter is not necessary.
At this point, the reservation availability needs to be checked (by contacting airlines to confirm if there are flights available for the requested date), confirm that the customer has all the necessary documentation to travel, or check if any financial obstacles exist in relation to this customer. Depending on all of the above, the reservation may be authorized or rejected.
Since this verification involves a series of tasks, we group them in a sub-process called Validate Reservation.



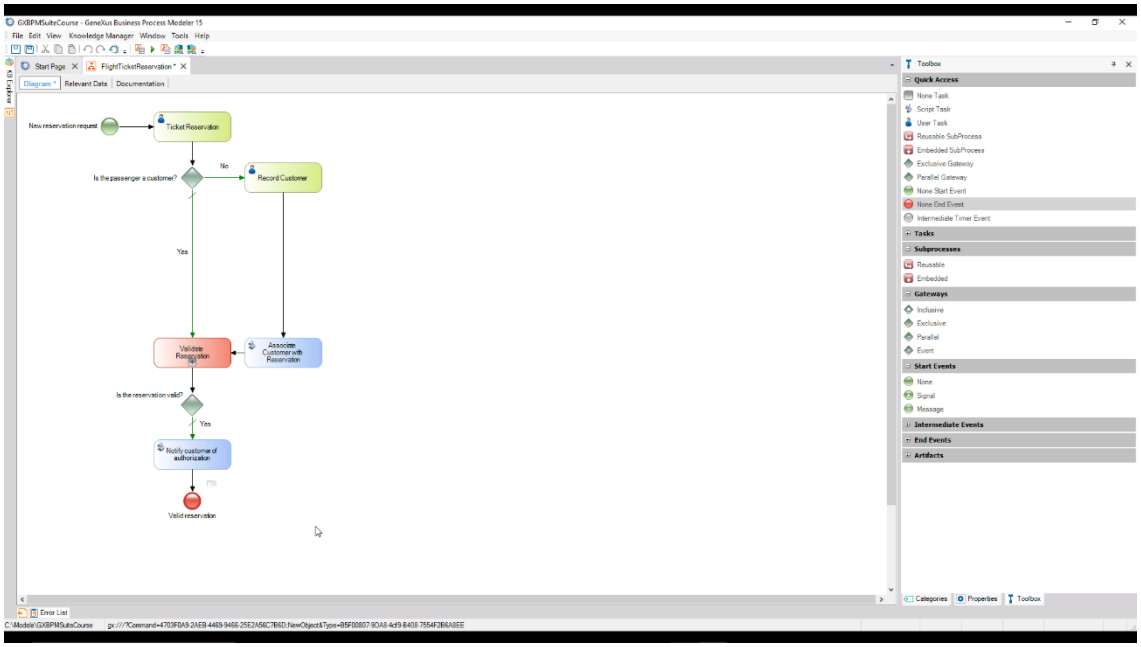
To indicate this in the diagram, we drag an Embedded Sub-process symbol from the toolbar, enter the name we have selected for it and connect it from the lower side of the Exclusive Gateway. We also add the text “Yes” to the place where the diagram will continue if the condition evaluates to true. To do so, we only need to press F2 after clicking on the connector.



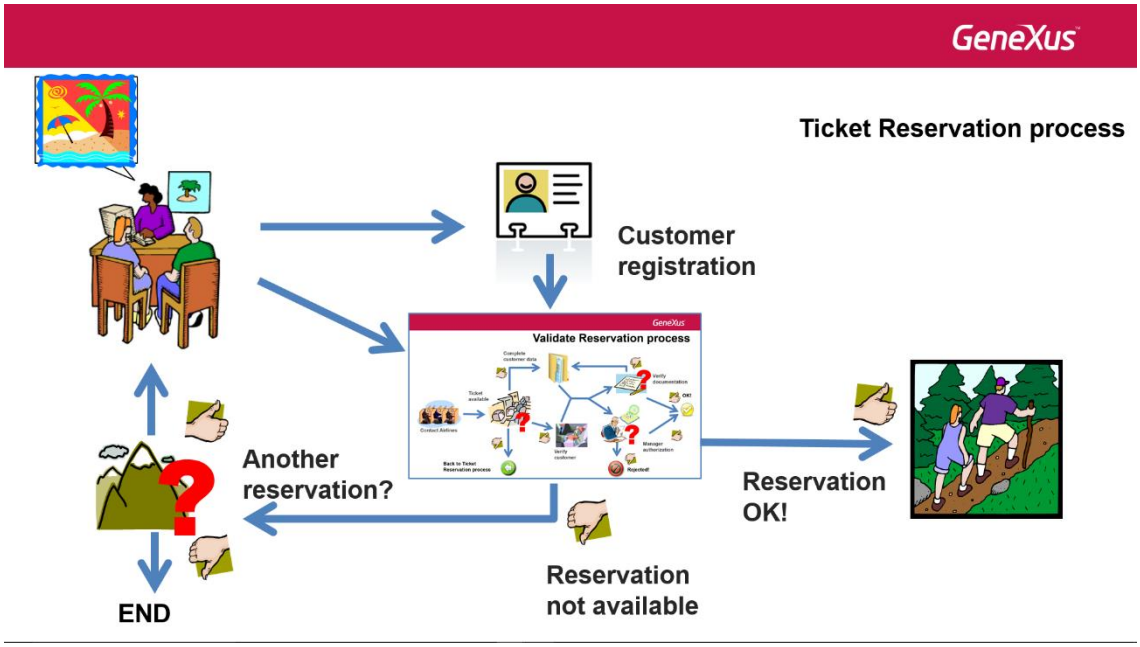
This connection, which was tagged with Yes, is our most common scenario. The reason is that, in general, the person requesting a reservation has already traveled through the agency and is an existing customer of the company. To indicate this, we select the connector, in the properties window we change its **Condition type** property and select the **Default** value.



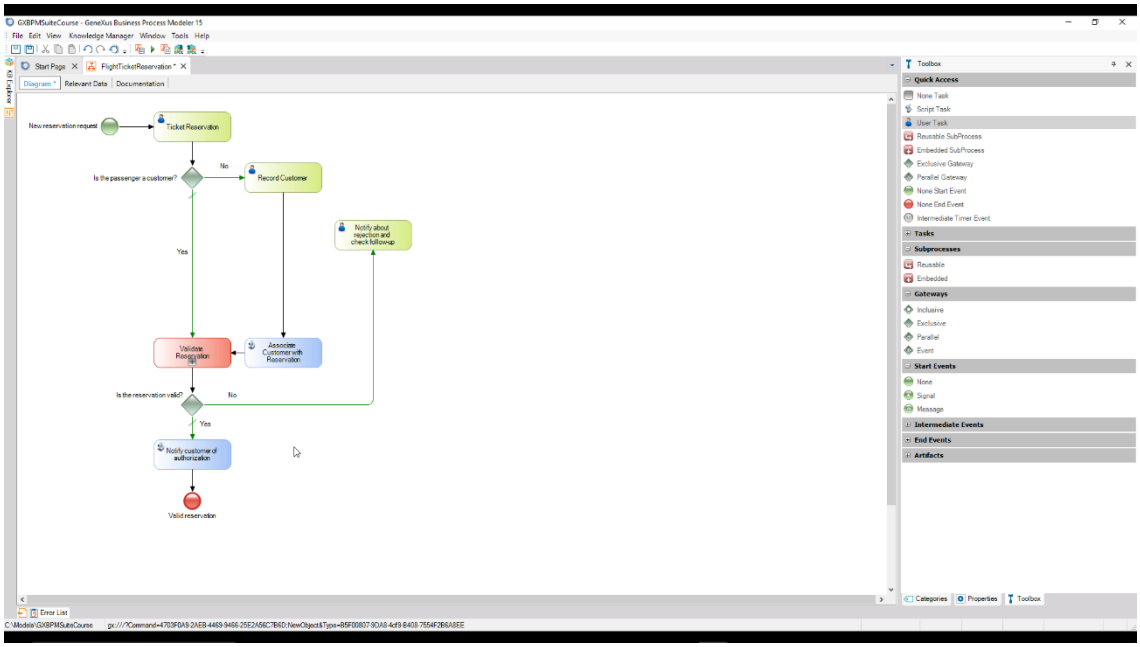
Note that in the diagram the flow has been marked with a green line that crosses it.
We also connect the Validate Reservation sub-process from the Associate Customer with Reservation task, because once the customer has been assigned, the reservation must be validated as well.



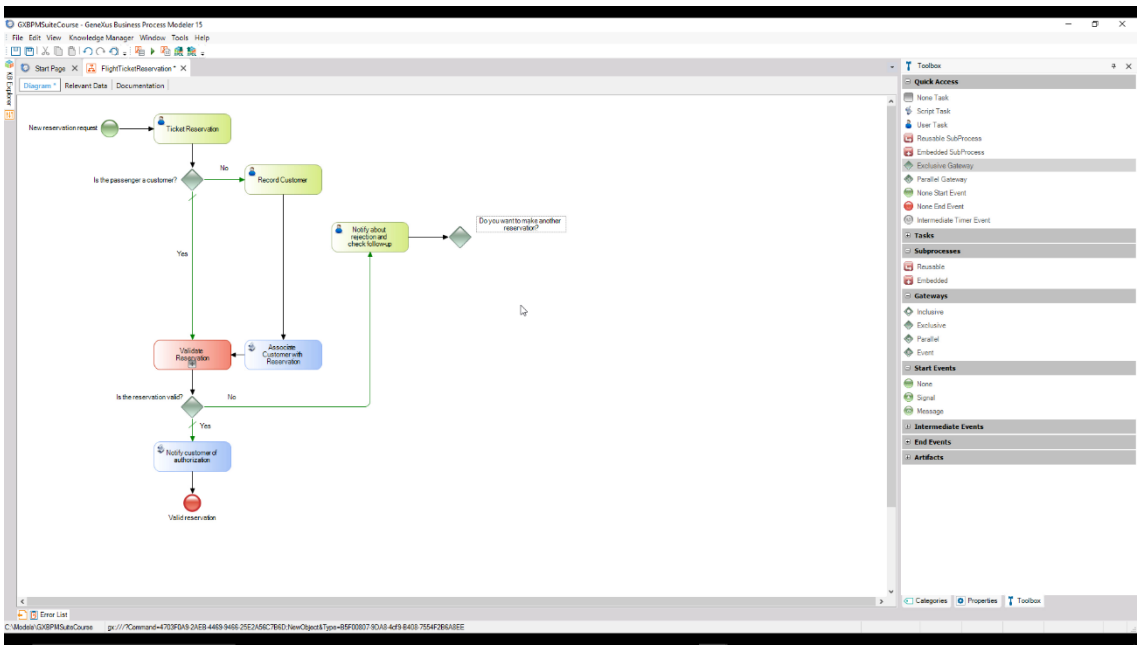
Moving on through the process, if the reservation was correctly validated, the customer will be able to make the trip.



We must evaluate this condition after the validation, so we add an Exclusive Gateway and connect it from the Validate Reservation sub-process.
We also add this description: “Is the reservation valid?” to indicate what we want to evaluate in this Gateway.



If the reservation is valid (which is considered the most probable case), the customer must be notified about the authorization and the process will end.
Since the notification will be emailed by the system, we begin by adding a task of script type with the name “Notify customer of authorization” and connect it from the Gateway. We select this connector, set its Condition type property to Default and add a description with the text ‘Yes’.



To indicate that the process will end after the notification, we insert a None End Event node and connect it from the notification task. Lastly, we add the description “Valid reservation” to the end symbol.

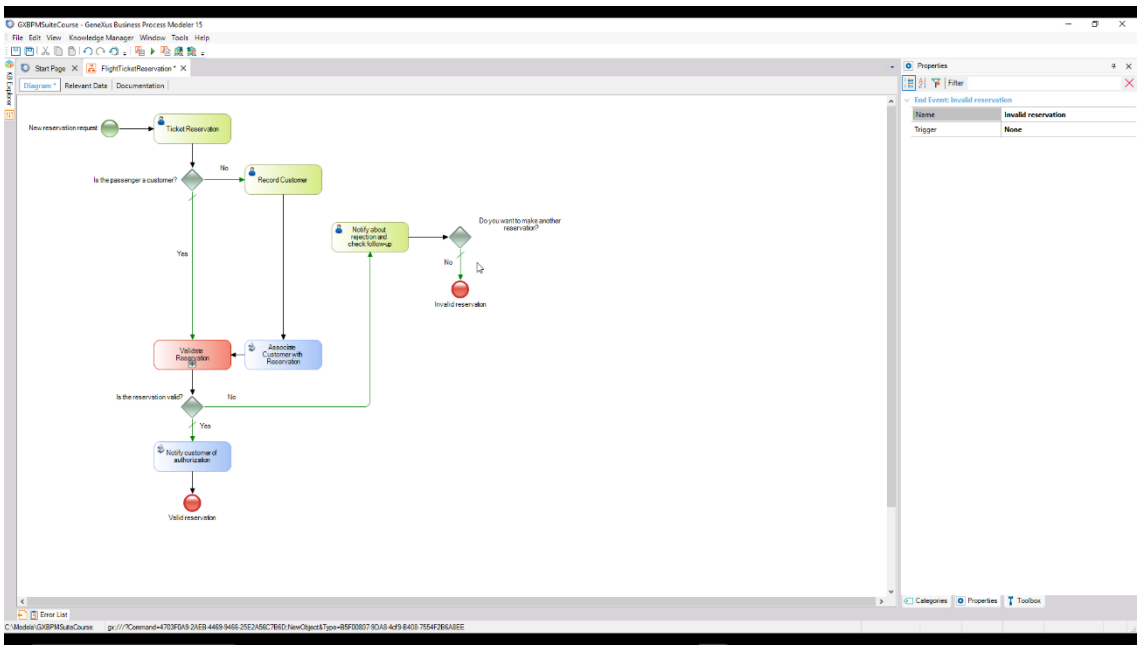
All of the above was added for the case when the reservation is valid. If it isn't valid, we should send a customized notification about the rejection to the customer and ask him if he wants to make another reservation.
If so, the ticket reservation process must start again; otherwise, the process must end.

To model this, we start by adding a task of User type to notify the customer about the rejection. We name it “Notify about rejection and check follow-up” and connect it from the right side of the gateway that evaluates if the reservation is valid.
To this connector we add the description: “No”.

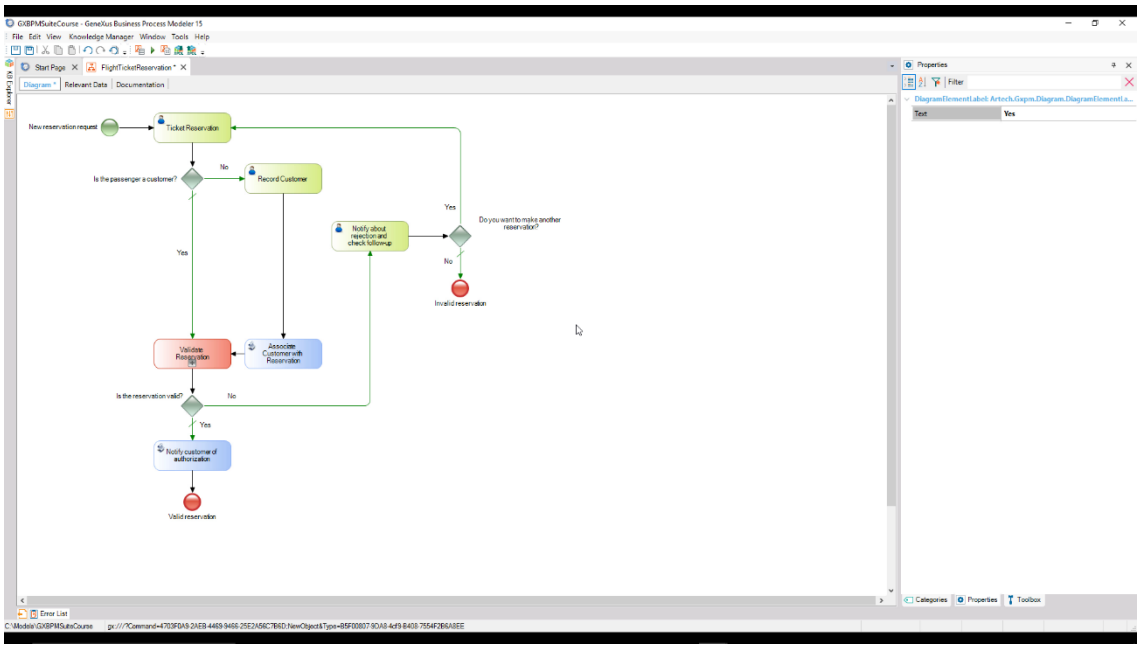
Once we tell the customer that the reservation is not valid, we need to ask him if he wants to make another reservation. To do so, we add an Exclusive Gateway and connect it from the notification task.

To the Gateway we add the description “Do you want to make another reservation?” to clarify its meaning.
If the answer is ‘No’ the process must end, so we insert a None End Event node and connect it from the lower side of the Gateway.
To this connector we add the description “No” and set it as Default. Below the end event, we add the description “Invalid Reservation”.

If the answer is ‘Yes’, we connect the Gateway to the Ticket Reservation task to indicate that a new reservation process will begin. To this connector we add the description ‘Yes’.



To view our finished diagram, we right-click on the empty section of the diagram and choose Zoom Out until the entire diagram is visible on the screen.



In this way we could have completed the diagram for the ticket reservation process.

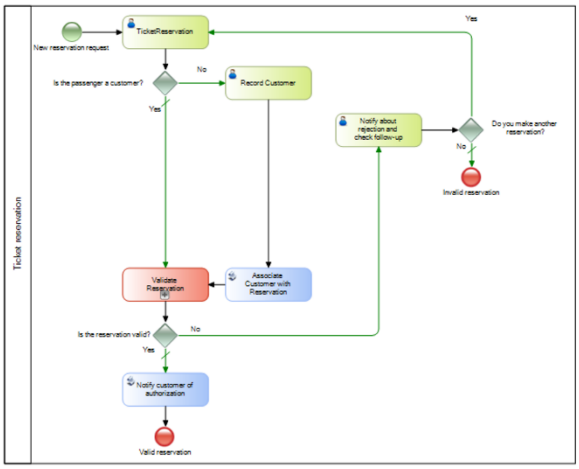
However, the BPMN standard provides a mechanism to document the following information in a diagram: the tasks related to each other, if they belong to the same participant or business entity, or if they are associated with a specific function or role in the company.

In our case, the only business entity involved is the Travel Agency, which has a single process that we called Ticket Reservation.

To indicate this, a **Pool** symbol is used; in our example it covers the entire diagram.

GeneXus

Pool



If the diagram had another business entity with its own process, we would group the tasks of this process in another pool.

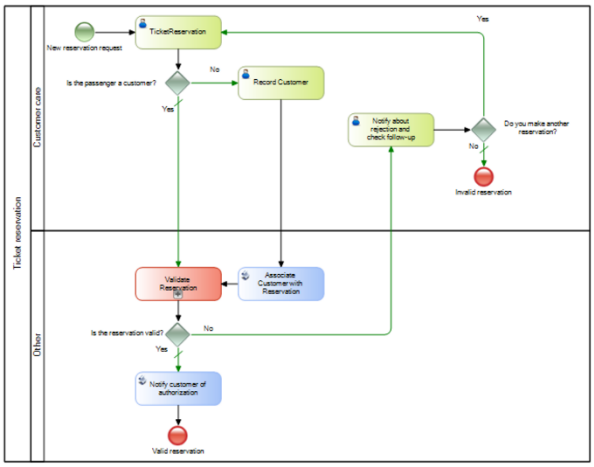
Another thing we could do to add better documentation to our diagram is to group tasks associated with a specific role or function in the company.

In our example, we want to make a group with all the tasks related to customer care and another group with all the other tasks.

To indicate this, the **Lane** symbols are used. A pool can include one or more lanes.

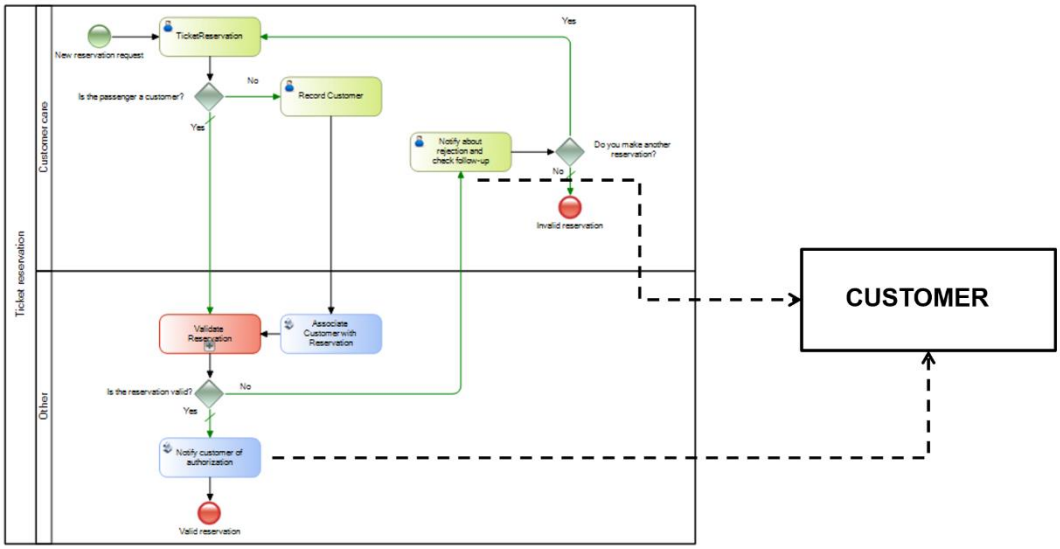
GeneXus

Lanes



It's worth pointing out that even though several process tasks interact with the customer, it isn't part of the process; it is an external entity.

This can be modeled using dotted lines to show interaction between the process and the customer.



Now that the main process is modeled, we will model the Validate Reservation sub-process...