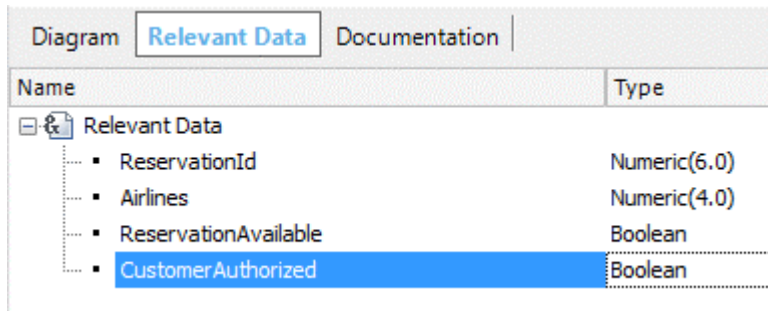


## Changing relevant data, timer event and calendars

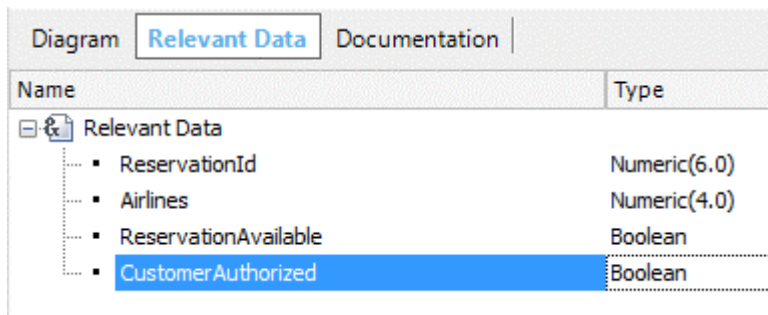
In the diagram, the **Evaluate Customer** task will have to evaluate the customer's financial situation and store the corresponding decision in a relevant data item that will be queried later on in the diagram, when the reservation is authorized.

So, we open the Relevant Data tab and create the CustomerAuthorized relevant data item of Boolean type.



Name	Type
Relevant Data	
▪ ReservationId	Numeric(6.0)
▪ Airlines	Numeric(4.0)
▪ ReservationAvailable	Boolean
▪ CustomerAuthorized	Boolean

Now we will associate the **Evaluate Customer** task with the **EvaluateCustomerStatus** web panel and map the ReservationId relevant data item.



Name	Type
Relevant Data	
▪ ReservationId	Numeric(6.0)
▪ Airlines	Numeric(4.0)
▪ ReservationAvailable	Boolean
▪ CustomerAuthorized	Boolean

When we open the web panel object, we can see the reservation identifier, the customer's details and two buttons: Authorize and Reject.

We open the Events tab and see that it loads the value of the relevant data item we've just entered with True or False value, depending on whether we click on the Authorize or Reject buttons, respectively.

In this web panel we're using methods of the workflow API to retrieve and change the "CustomerAuthorized" relevant data item.

```

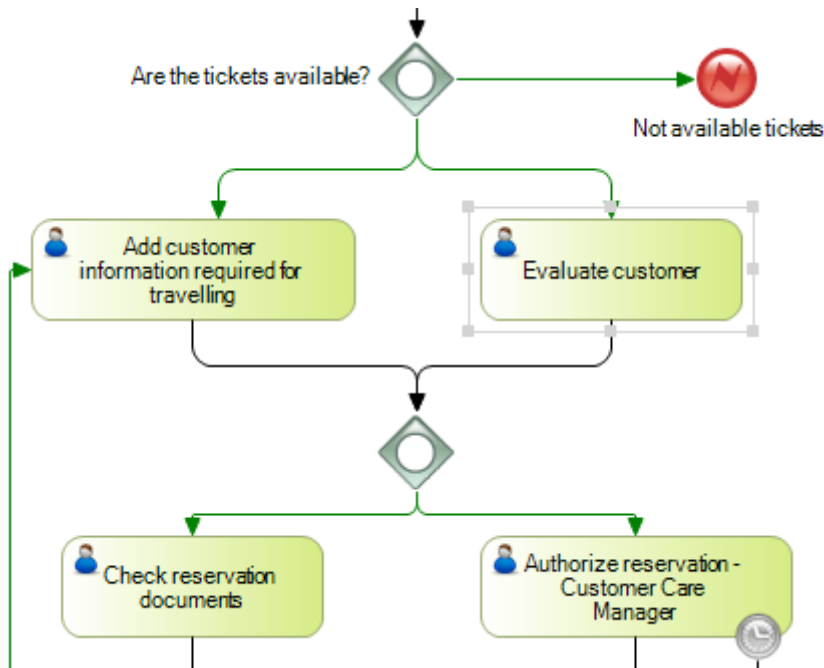
1 Event 'Authorize'
2   &CustomerAuthorizedWorkflowApplicationData = &WorkflowContext.ProcessInstance.GetApplicationDataByName("CustomerAuthorized")
3   &CustomerAuthorizedWorkflowApplicationData.BooleanValue = True
4   return
5 -Endevent
6
7 Event 'Refuse'
8   &CustomerAuthorizedWorkflowApplicationData = &WorkflowContext.ProcessInstance.GetApplicationDataByName("CustomerAuthorized")
9   &CustomerAuthorizedWorkflowApplicationData.BooleanValue = False
10  return
11 -Endevent

```

Remember that in web panels, unlike procedures, mapping values between the relevant data and variables included in the Parm rule is valid only for input variables. On the other hand, in procedures the mapping of values is valid for both input and output variables.

**For this reason, in a procedure we only have to create a variable with the same name as that of a relevant data item and set it as output variable in the Parm rule. In this way, its value will be sent to the diagram's relevant data item with no need to use API methods to access it.**

We go back to the diagram and see that there's an inclusive Gateway after the tasks **Evaluate Customer** and **Add customer information required for traveling**, which closes the paths opened by another inclusive Gateway.



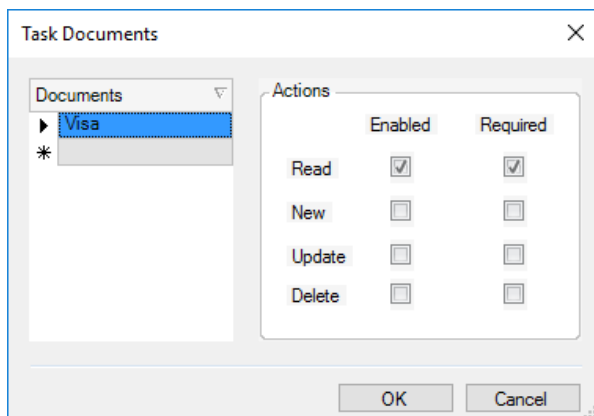
In this case, the second Inclusive Gateway will synchronize the paths that arrive to it, but unlike the Parallel Gateway that waits for **all the diagram paths**, the Inclusive Gateway synchronizes only the paths **that actually reach the Gateway at runtime** and not all those that are included in the diagram.

Since in this case we run both tasks, the flow will continue to the tasks **“Check reservation documents”** and **“Authorize reservation – Customer Care Manager”**.

Let’s continue with the task on the left.

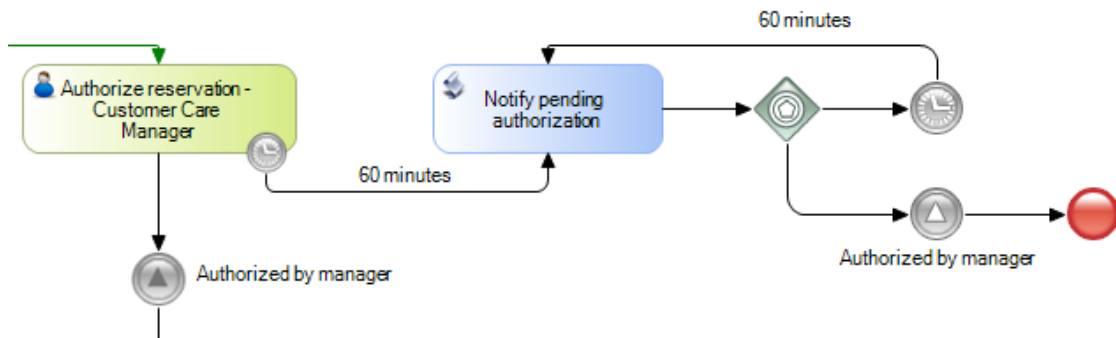
So far, we’ve associated documents with the task **“Add customer information required for traveling”**. However, other tasks such as **“Check reservation documents”** can access these documents.

To do so, we open the Work With Documents property and set it to True. Next, we select the document; for example, **“Visa”**.



In this case we define that the action is Read and it is required.

We click on OK and go back to the diagram to focus on the reservation authorization task, which is performed by the customer care manager.



Remember the agency’s requirements: in this task, the customer care manager must receive a system notification every hour to remind him about the pending authorization. Once the manager authorizes the task, the system will have to cancel these notifications and the process will have to continue.

To model this type of timing and notification, we first use an intermediate event of **timer** type, associated with the task. If we open the properties of this event, we can see that it has been set with a 60-minute deadline.

Intermediate Event: IntermediateEvent	
Name	IntermediateEvent
Trigger	Timer
Interrupts activity	False
Timer definition	Compatible
Timer usage	Deadline
Submit to calendar	False
Time unit	Minutes
Lapse expression type	Rule
Lapse expression rule	60

When the **Interrupts activity** property is set to False, it means that this timer will not interrupt the task when the 60-minute deadline expires.

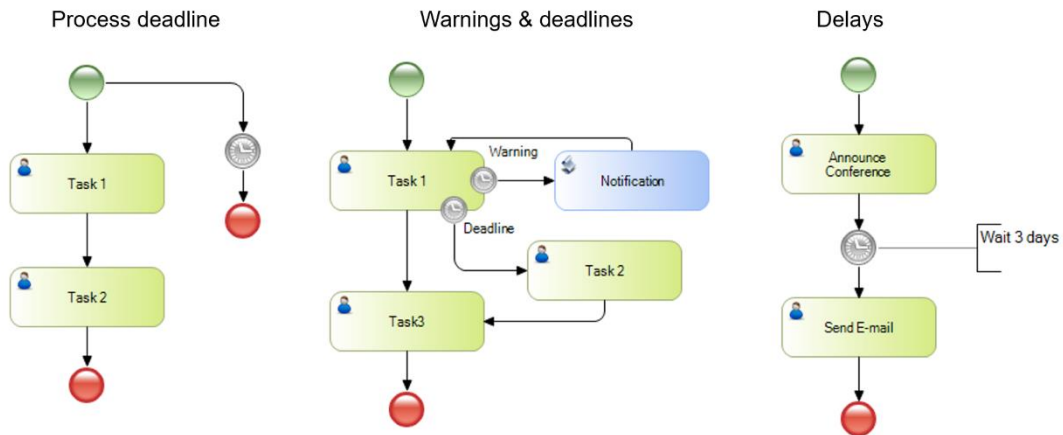
Once this time has elapsed, the batch task “**Notify pending authorization**” will be executed to notify the manager. After sending the notification, a Gateway of event type, associated with another timer, will make sure that the message is repeated every 60 minutes.

Once the manager authorizes the task, the process flow will move down from the task and will reach an intermediate event of signal type, configured as “**throw**” (note the dark color of the arrow) that will send a signal that will be captured by another intermediate event of signal type, which is configured as “**catch**” (with a light color arrow) and will end this notification pattern with an end event.

Timers allow us to model various patterns related to time, such as setting a deadline for the process or tasks (deadlines), sending an alert about a near deadline (warning), or setting a delay between activities (delay).



### Timing patterns

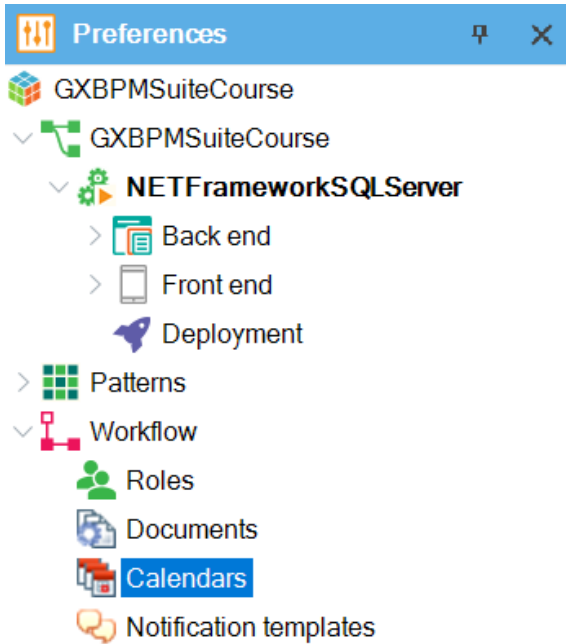


This configuration is made from the timer properties, where the **Timer usage** property can be set to implement a deadline or a warning, and the **Interrupts activity** property is used to set whether or not to interrupt the task once the indicated period has elapsed.

Intermediate Event: IntermediateEvent	
Name	IntermediateEvent
Trigger	Timer
Interrupts activity	False
Timer definition	Compatible
Timer usage	Deadline
Submit to calendar	None
Time unit	Deadline
Lapse expression type	Warning
Lapse expression rule	60

Note that we have a **Submit to calendar** property. This property allows us to set the deadline or warning to be associated with a specific calendar –that is to say, including the working days and times and leaving out holidays.

To create a calendar, we open the Preferences window and double-click on Calendars below Workflow.

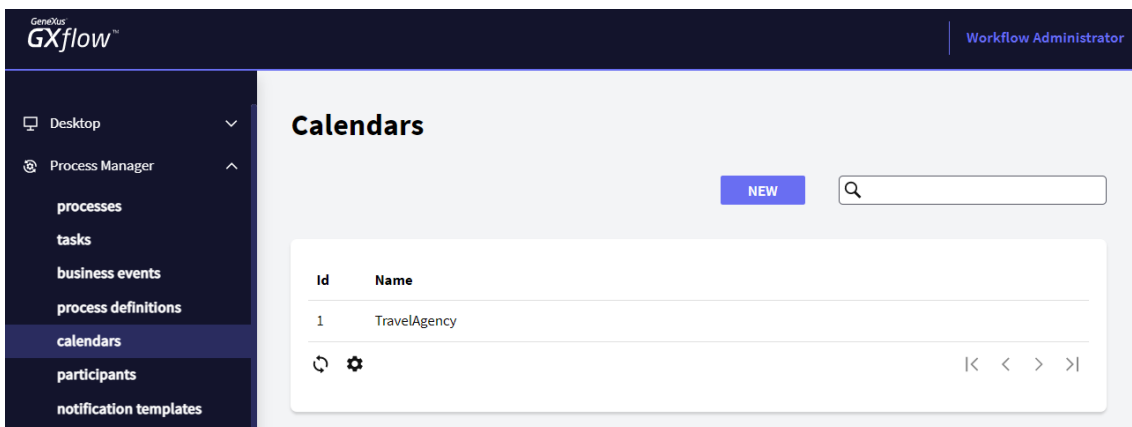


We enter a name for the calendar, such as Travel Agency. Now we select the **Validate Reservation** diagram, in the Calendar property we select **Travel Agency** and save.

Name	Description
Calendars	
TravelAgency	Travel Agency

Working days and times, as well as non-working holidays, are set from the workflow client. We will run the process **FlightTicketReservation** process to do so.

In the browser window, below Process Manager, we select Calendars.



We can see that the Travel Agency calendar we've just created is displayed. We select it and click on Edit.

✕

### Edit Calendar

General Schedules Holidays

Name

Description

We open Schedules and choose the travel agency's working days and times.

✕

### Edit Calendar

General Schedules Holidays

Remove	Day Begin	Hour Begin	Day End	Hour End
<input type="checkbox"/>	None	00:00	None	00:00
<input type="checkbox"/>	None	00:00	None	00:00
<input type="checkbox"/>	None	00:00	None	00:00
<input type="checkbox"/>	None	00:00	None	00:00
<input type="checkbox"/>	None	00:00	None	00:00

We can also open the "Holidays" window and set the company's holidays, such as May 1<sup>st</sup>.

✕

### Edit Calendar

General Schedules Holidays

Remove	Month	Day
<input type="checkbox"/>	May	1
<input type="checkbox"/>	Month	Day
<input type="checkbox"/>	Month	Day
<input type="checkbox"/>	Month	Day
<input type="checkbox"/>	Month	Day
<input type="checkbox"/>	Month	Day