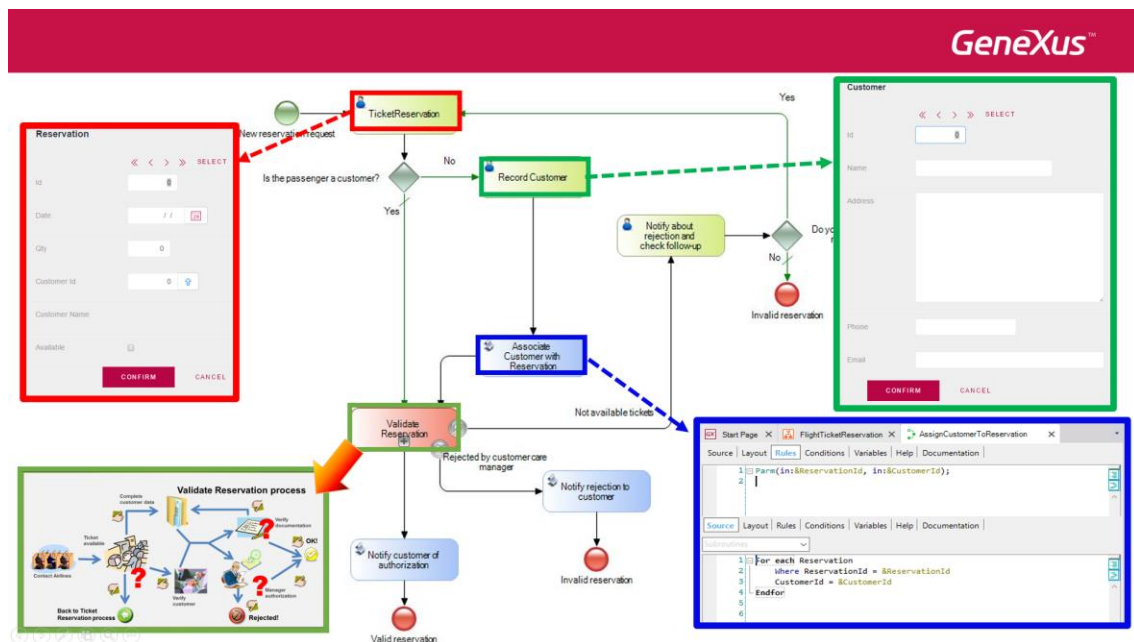


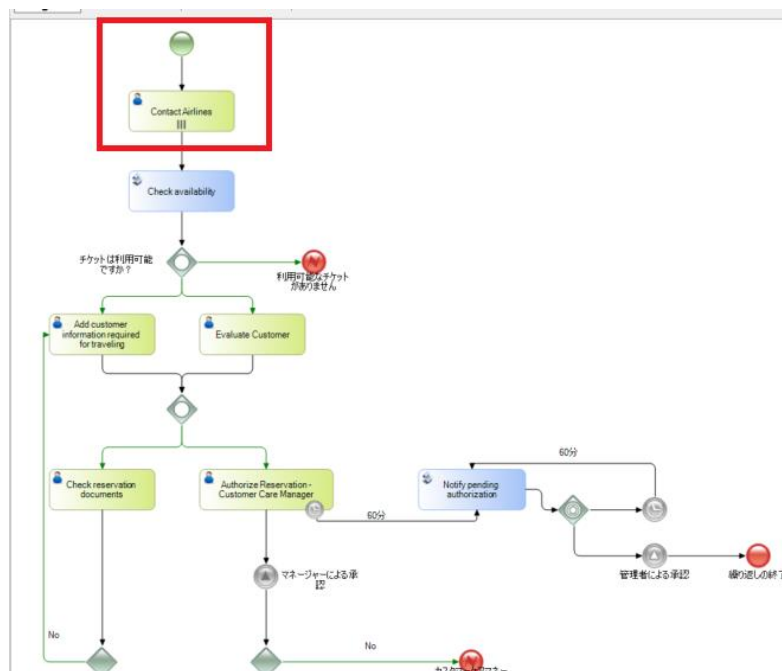
マルチインスタンスタスクと関連データのマッピング

以前の章では、旅行代理店の「TicketReservation」ダイアグラムのタスクを GeneXus オブジェクトに関連付け、プロセスモデルを機能的なアプリケーションに変換しました。



「TicketReservation」プロセスのサブプロセスである「ValidateReservation」ダイアグラムでも同様の手順を続けます。

「ValidateReservation」ダイアグラムを確認すると、最初のタスクは航空会社に連絡することであるとわかります。



このタスクの特徴は、複数の航空会社に連絡する必要があるため、一定回数実行されることです。これは、異なるユーザーから同時に実行される場合もあります。

「Contact Airlines」では「Loop type」プロパティが「Multi-Instance」、「Expression type」プロパティが「Rule」に設定されています。「Ordering」プロパティを「Parallel」に設定します。

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

「Expression rule」プロパティの値を「10」に設定します。これは、モデリング段階で示されたとおり、タスクが正確に 10 回並行して繰り返されることを意味します。

さらに、「Flow condition」プロパティが「All」に設定されているため、「Contact Airlines」タスクは 10 回のインスタンスが実行された後にのみ終了します。

しかし、旅行代理店のスタッフと一緒にこのタスクを詳しく調べると、このタスクを実行する回数は、旅行代理店が現在取引している航空会社の数によって決まることがわかりました。さらに、この回数は時間とともに変化する可能性があります。

代理店が登録している航空会社の数を調べるには、旅行代理店の航空会社のテーブルを調べて登録されている航空会社の数を返すプロシージャを使用することができます。

これを実装するには、「Expression type」プロパティを「Procedure」に変更し、「Expression procedure」プロパティにて「LoopAirlines」プロシージャを選択します。

▼ Looping	
Loop type	Multi-Instance
Expression type	Procedure
Expression procedure	LoopAirlines
Ordering	Parallel
Flow condition	All

「LoopAirlines」プロシージャでは Source にて「Airlines」テーブルを参照し、登録されている航空会社の数をカウントする For Each コマンドが設定されています。

```
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
```

さらに、「ValidateReservation」ダイアグラムの関連データ項目として設定された航空会社の識別子を配列にロードします。

Diagram * Relevant Data * Documentation	
Name	Type
Relevant Data	
▪ ReservationId	Numeric(6.0)
▪ Airlines	Numeric(4.0)

この Relevant Data 項目には、Workflow エンジンが提供する API のメソッドを使用してプロシージャからアクセスします。これについては、別の章で詳しく説明します。

```
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
```

前述したように、航空会社の数によって「Contact Airlines」タスクで作成されるインスタンスの数が決まるため、この値はプロシージャによって、Parm ルールの最後のパラメータで「Contact Airlines」タスクに返されます。

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

まとめると、複数のインスタンスを持つタスクを定義するには、「LoopType」プロパティに「Multi-Instance」値を設定します。このタスクがインスタンス化される回数を示すには、「Expression type」プロパティを「Rule」に設定し、「Expression Rule」プロパティに数値を割り当てます。または、「Expression Type」プロパティを「Procedure」に設定し、この例で見たように、タスクがインスタンス化される回数を返すプロシージャを使用することもできます。

ダイアグラムに戻り、「Contact Airlines」タスクに WebPanel オブジェクトを関連付けることで、タスクが実行されるたびに WebPanel が実行されます。ここでは「ContactAirline」という WebPanel オブジェクトを指定します。この WebPanel を使用すると、航空会社ごとに予約に最適なフライトを選択できます。

タスク: Contact airlines

Name	Contact airlines
Task Metadata Id	2
Task Metadata GUID	a9d5f688-4fb2-4af1-b2ce-40d7e977
Type	User
Web Object	ContactAirline
Object	(none)
Visible in history	True

WebPanel を実行すると、実行中のタスクインスタンスは航空会社の 1 つと内部的に関連付けられるため、新しいタスクインスタンスが開始されるたびに、代理店によって登録済みの異なる航空会社と連絡が取れます。

「ContactAirline」では、予約の詳細と、選択した航空会社が予約日に利用できるフライトが表示されます。

Web Layout | Rules | Events | Events (without WorkWithPlus code) | Conditions | Variables | Help | Documentation | Patterns

選択されているアクショングループはありません

MainTable | row

Assign Flight to Reservation

Airline to contact:

Reservation Information

Id: &ReservationId

Date: &ReservationDate

Qty: &ReservationQty

Customer Name: &CustomerName

Departure Airport

Arrival Airport

&ReservationDepartureAirportName

,

&ReservationDepartureCityName

,

&ReservationDepartureCountryName

&ReservationArrivalAirportName

,

&ReservationArrivalCityName

,

&ReservationArrivalCountryName

代理店の従業員は、予約に関連付けられるフライトを選択できます。実行して見えます。

「FlightTicketReservation」ダイアグラムのタブを右クリックして実行を選択します。

「TicketReservation」タスクを実行し、今日の予約を入力します。顧客はジョン・パーカーと呼ばれ、モンテビデオのカラスコ空港からサンパウロのグアルーリョス空港まで旅行したいと考えています。「確認」をクリックして画面を閉じます。

RESERVATION

Reservation

<< < > >> 選択

Id

0

Date

24/06/27

29

Qty

1

Customer Id

1

Customer Name

John parker

Airport Id

1

Airport Name

Carrasco

City Id

1

City Name

Montevideo

Country Id

4

Country Name

Uruguay

Airport Id

2

Airport Name

Guarulhos

City Id

2

City Name

Sao Paulo

Country Id

1

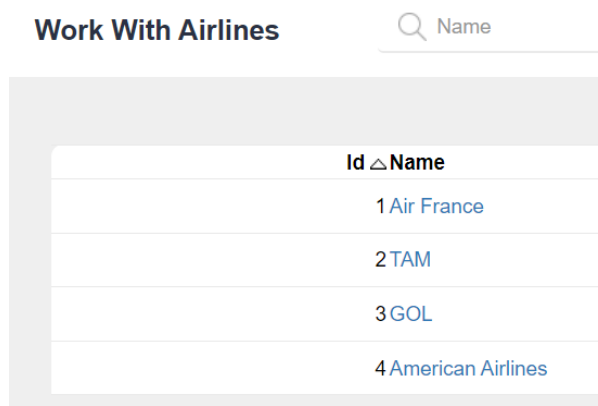
Country Name

Brazil

タスクを送信するために「送信」をクリックすると、「ContactAirlines」の保留中のタスクが 4 つ表示されます。




その理由は、4 つの航空会社に連絡する必要があるため、代理店に登録されている航空会社ごとに「ContactAirlines」タスクのインスタンスが 1 つ作成されているからです。



最初の保留中のタスクをダブルクリックすると、最初の航空会社に連絡するための画面が開きます。予約に必要な日付、出発地、目的地に利用可能なフライトがあるので、フライトを選択して「Select flight」をクリックします。

CONTACT AIRLINE

Airline to contact:



4

Reservation Information

Id

14

Date

07/25/17

Qty

1

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
5	07/25/17	Carrasco	Montevideo	Uruguay	Guarulhos	Sao Paulo	Brazil	1100

Select flight

5


このようにして、リクエストされた予約の要件を満たす可能性のあるフライトを割り当てます。

ウィンドウを閉じてタスクを完了すると、受信トレイに保留中のタスクとして表示されなくなります。

次のタスクを実行すると、別の航空会社が割り当てられていることがわかります。これは、「ContactAirlines」タスクのすべてのインスタンスで発生します。

CONTACT AIRLINE

Airline to contact:



2

Reservation Information

Id

14

Date

07/25/17

Qty

1

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
2	07/25/17	Carrasco	Montevideo	Uruguay	Guarulhos	Sao Paulo	Brazil	1200
3	07/25/17	Carrasco	Montevideo	Uruguay	Guarulhos	Sao Paulo	Brazil	890

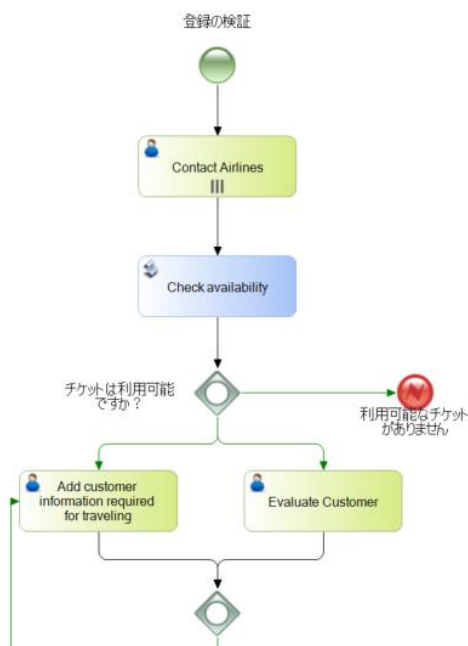
Select flight

別の航空会社の割り当ては、WebPanel オブジェクトで解決されます。航空会社の識別子を格納する航空会社関連データ項目（配列型）に基づいて、WebPanel が起動されるたびに、「ContactAirlines」タスクのインスタンスごとに異なる配列要素が取得されるためです。

```
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
```

予約検証プロセスの次のステップでは、航空会社に連絡した後、「CheckAvailability」タスクを使用して、予約の要件を満たすフライトが少なくとも 1 つあるかどうかを確認します。



「CheckReservationFlights」プロシージャを開くと、ソースに For Each コマンドがあり、予約詳細のテーブルを実行して、予約に少なくとも 1 つのフライトが選択されているかどうかを確認します。フライトが選択されている場合は、変数「&ReservationAvailable」に「True」値が割り当てられます。

```
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
```

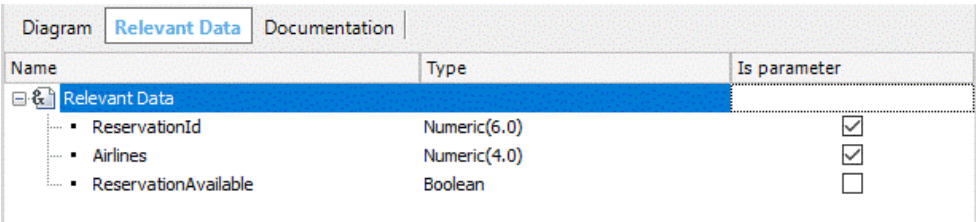
この変数は、プロシージャの「Parm」ルール最後のパラメータとして返されます。

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

この変数を関連データ項目と同じ名前にすると、ワークフローエンジンは自動的に関連データ項目を変数値とともにロードします。

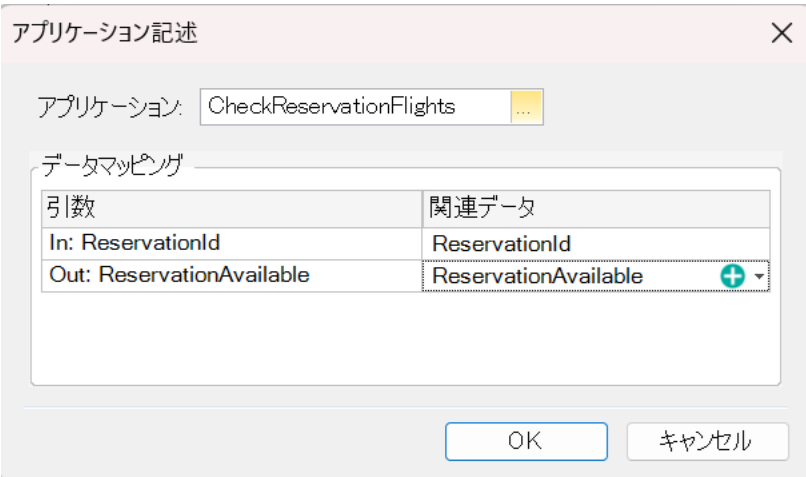
プロシージャオブジェクトでは、関連するデータと「Parm」ルールに含まれる変数間の値のマッピングは、入力変数と出力変数の両方に有効です。一方、WebPanel オブジェクトでは、値のマッピングは入力変数に対してのみ有効です。

そこで、「ValidateReservation」ダイアグラムを開き、「RelevantData」エレメントを選択して、Boolean 型の「ReservationAvailable」という関連データ項目を作成し、このデータはダイアグラムオブジェクトのパラメータではないため、「IsParameter」チェックボックスをオフにします。



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"/>

最後に、「CheckReservationFlights」プロシージャを「CheckAvailability」バッチタスクに関連付け、関連するデータ「ReservationId」と「ReservationAvailable」をマッピングします。



アプリケーション記述

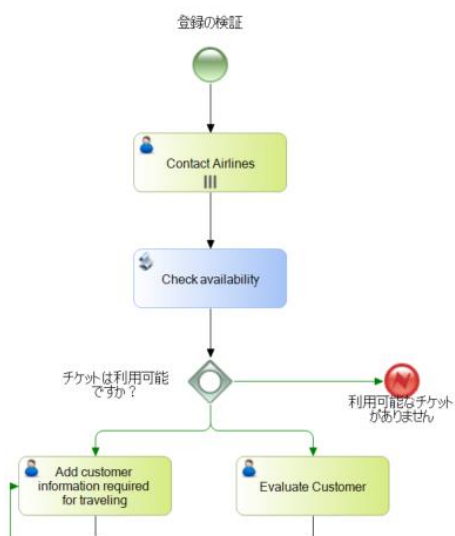
アプリケーション: CheckReservationFlights

データマッピング

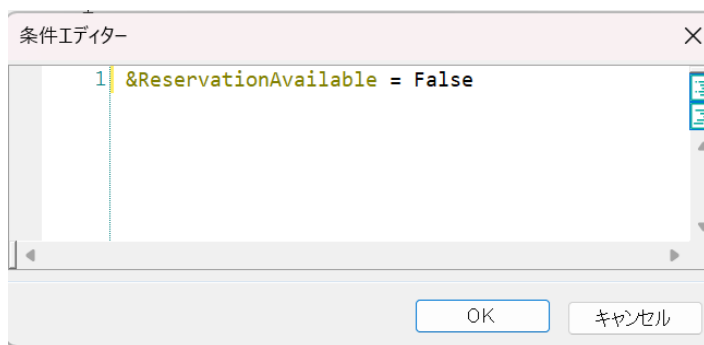
引数	関連データ
In: ReservationId	ReservationId
Out: ReservationAvailable	ReservationAvailable

OK キャンセル

以下の場合、プロシージャが予約可能かどうかを設定すると、包括的ゲートウェイ「チケットは利用可能ですか？」は、ロードした関連データ項目の値をチェックする必要があります。

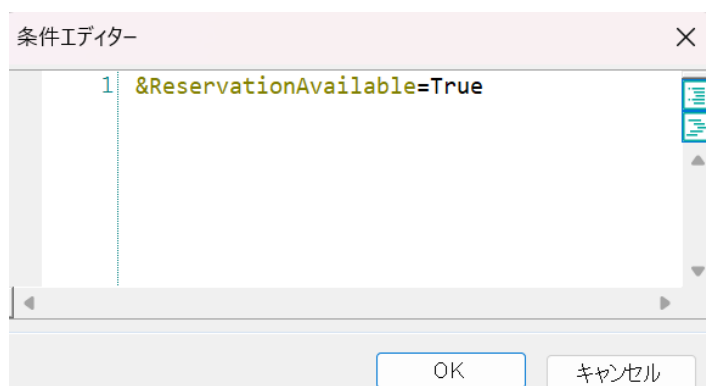


これを行うには、包括的ゲートウェイの右側にある送信コネクタをダブルクリックし、「&ReservationAvailable=False」と入力します。「Text」プロパティには「利用可能なチケットがありません」と入力します。



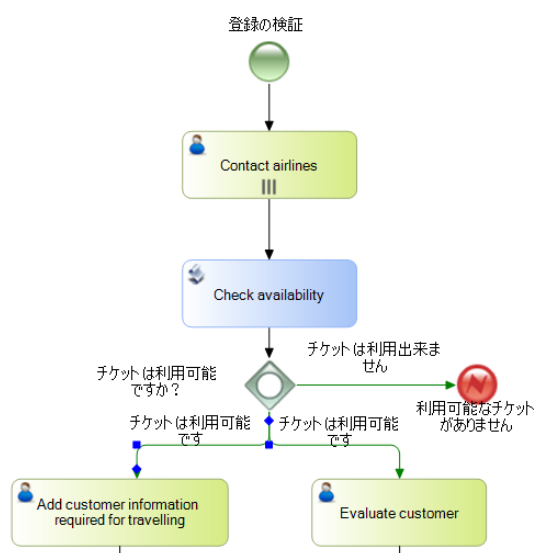
包括的ゲートウェイから進む 2 つのコネクタにも同じことを行い、条件に

「&ReservationAvailable=True」を割り当て、「Text」プロパティに「チケットは利用可能です」と入力します。



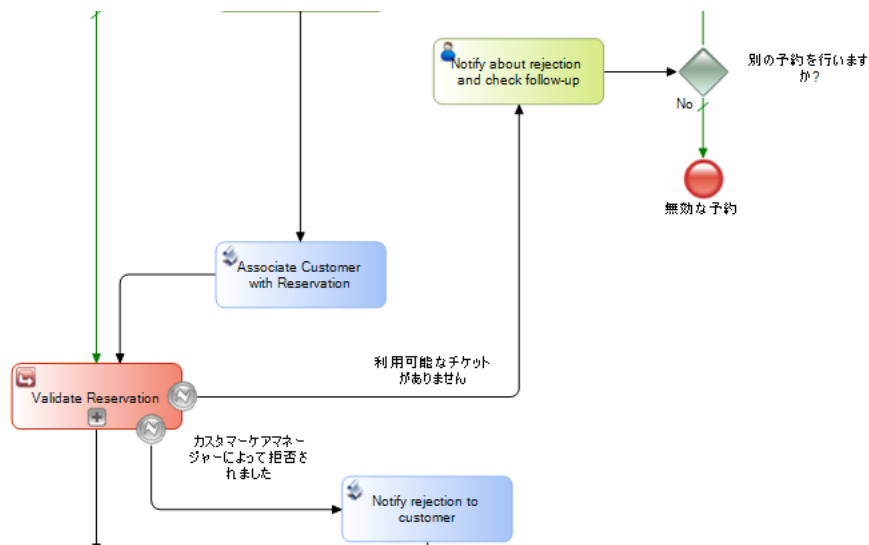
ゲートウェイの条件式には、関連データ、定数（この場合の True 値など）、列挙されたドメイン、およびダイアグラムに関連付けられたトランザクションの拡張テーブルの項目属性を含めることができます。

この定義によって、予約に利用可能なフライトがある場合、フローはゲートウェイから下に移動します。予約に利用可能なフライトがない場合は右に移動し、「利用可能なチケットがありません」というエラーの終了イベントで終了します。



エラーで終了するイベントにより、予約検証サブプロセスを終了し、メインのチケット予約プロセスにエラー通知を送信できます。

メインプロセスを見ると、同じタグ「チケットがありません」を持つ中間エラーイベントのシンボルも存在し、顧客に問題を通知する対話型タスクに接続されていることがわかります。



エラー中間イベントは「キャッチ」型ですが、サブプロセスエラー終了イベントは「スロー」型です。

このようにして、メインプロセスからサブプロセスが終了した理由を見つけ、対応するアクションを実行することができます。

次の章では、予約検証サブプロセスを継続し、同時に実行される対話型タスク「Add customer information required for travelling」と「Evaluate customer」を使用します。