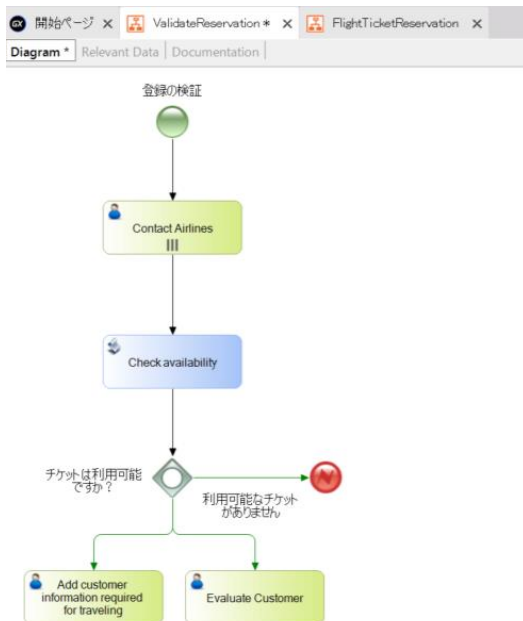


## パスの分岐と結合、定期的な通知の生成、およびシグナル管理

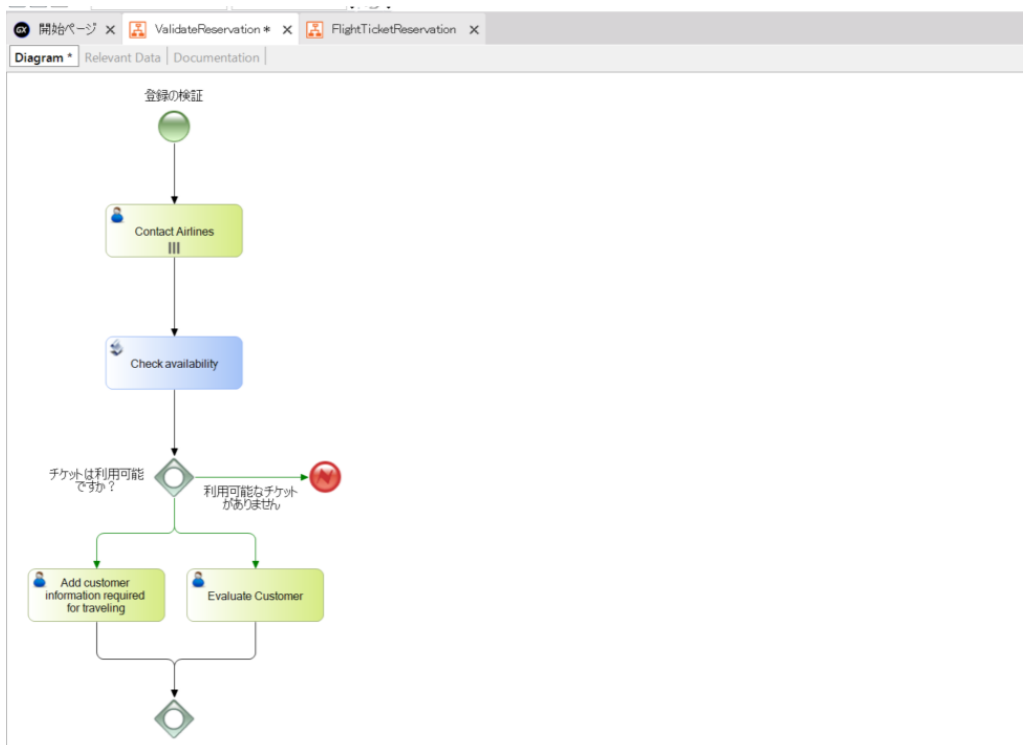
予約検証プロセス「ValidateReservation」のモデルにて、チケットが利用可能な場合は、同時に 2 つのパスを通過する必要があります。 インタラクティブなタスクとなるユーザータスク「Add customer information required for traveling」と「Evaluate Customer」を追加し、両方を包括的なゲートウェイから接続します。



この場合、包括的ゲートウェイを使用して分岐パスを作成しています。各パスには条件が定義されており、プロセスは満たされる条件に応じて 1 つ以上のパスを同時にたどります。

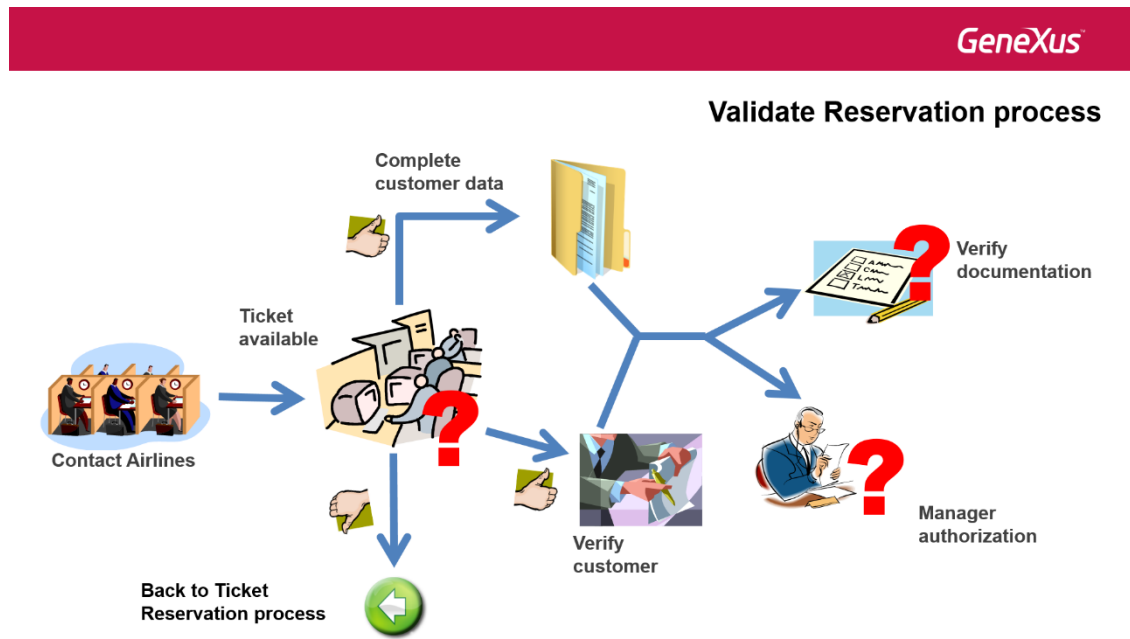
旅行に関する情報の取得と顧客の財務状況の確認というタスクが完了したら、プロセスを続行する必要があります。次に進むには、両方のタスクを完了する必要があります。

つまり、1 つのタスクが他のタスクより先に終了した場合、プロセスを続行するには 2 番目のタスクが完了するまで待つ必要があります。これを実現するために、包括的なゲートウェイも使用します。そのため、ツールバーからゲートウェイをドラッグし、両方のタスクから接続します。

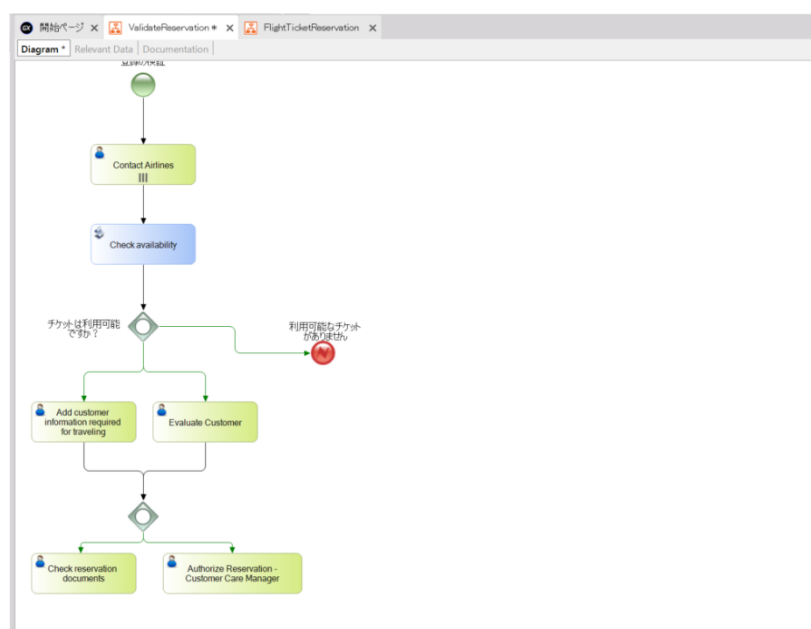


パスが結合されると、包括的なゲートウェイは、プロセスが実際に移動したパスを同期します。これらがゲートウェイへのすべてのパスである必要はありません。

この場合、プロセスは両方のタスクのいずれかから進むことができます。これらのタスクは、包括的ゲートウェイへの唯一の受信パスです。したがって、両方が完了すると、プロセスは続行されます。また、旅行に必要な詳細の確認と管理者の承認の取得という2つのアクティビティの実行も含まれます。



2つのパスへのフローの分岐をモデル化するために、前のケースと同様に包括的なゲートウェイを使用します。したがって、パスを結合するために使用したのと同じ包括的なゲートウェイを利用して、タスク「Check reservation documents」と「Authorize Reservation -Customer Care Manager」を作成しこのノードからそれらを接続します。



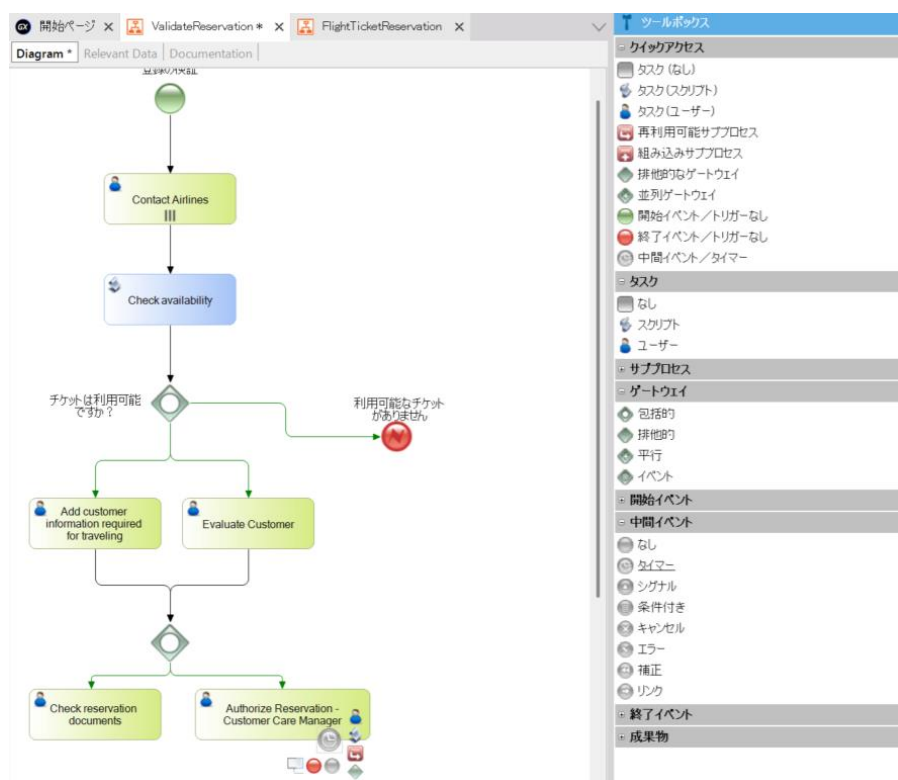
顧客ケアマネージャーが実行するタスクは完了までに一定の期限が必要です。

また、予約が承認待ちであることをシステムから定期的かつ繰り返しの方法で通知する必要があります。

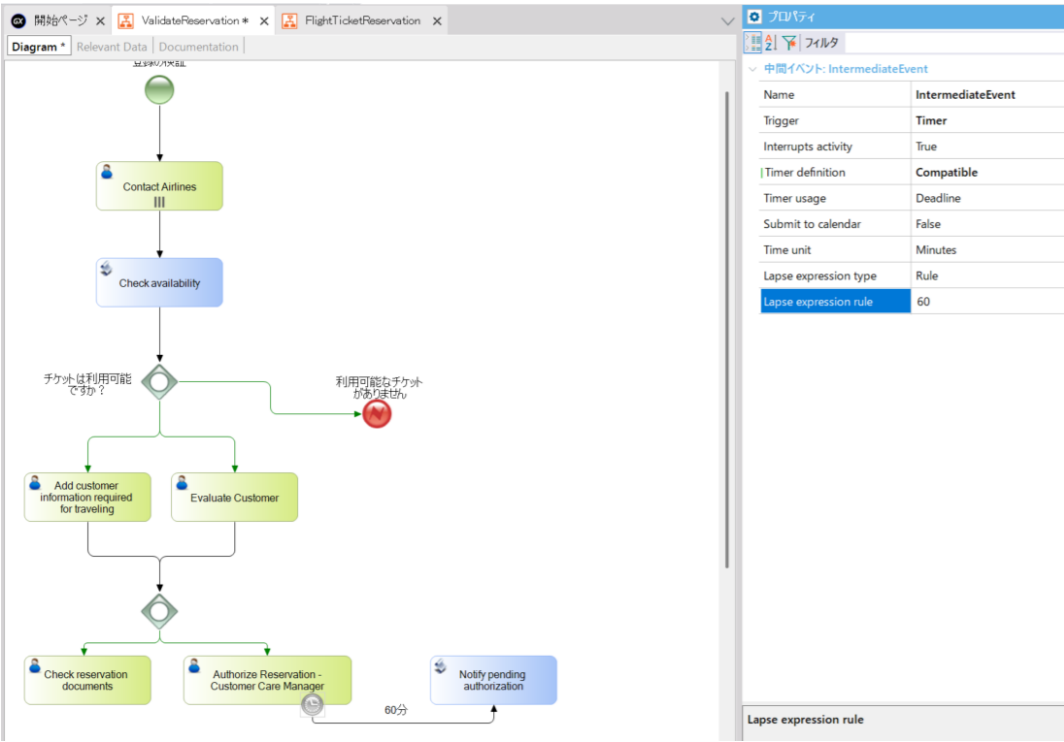
経過時間をマークするには、タイマータイプの中間イベントをタスクに追加します。

イベントとは、プロセスの実行中に発生するものです。プロセスを開始、一時停止、中断、または終了があります。イベントの性質に応じて、開始イベント、中間イベント、または終了イベントに分類できます。すでに、それぞれ「開始イベント/トリガーなし」と「終了イベント/トリガーなし」など、開始イベントと終了イベントの最も簡単な例を使用しました。

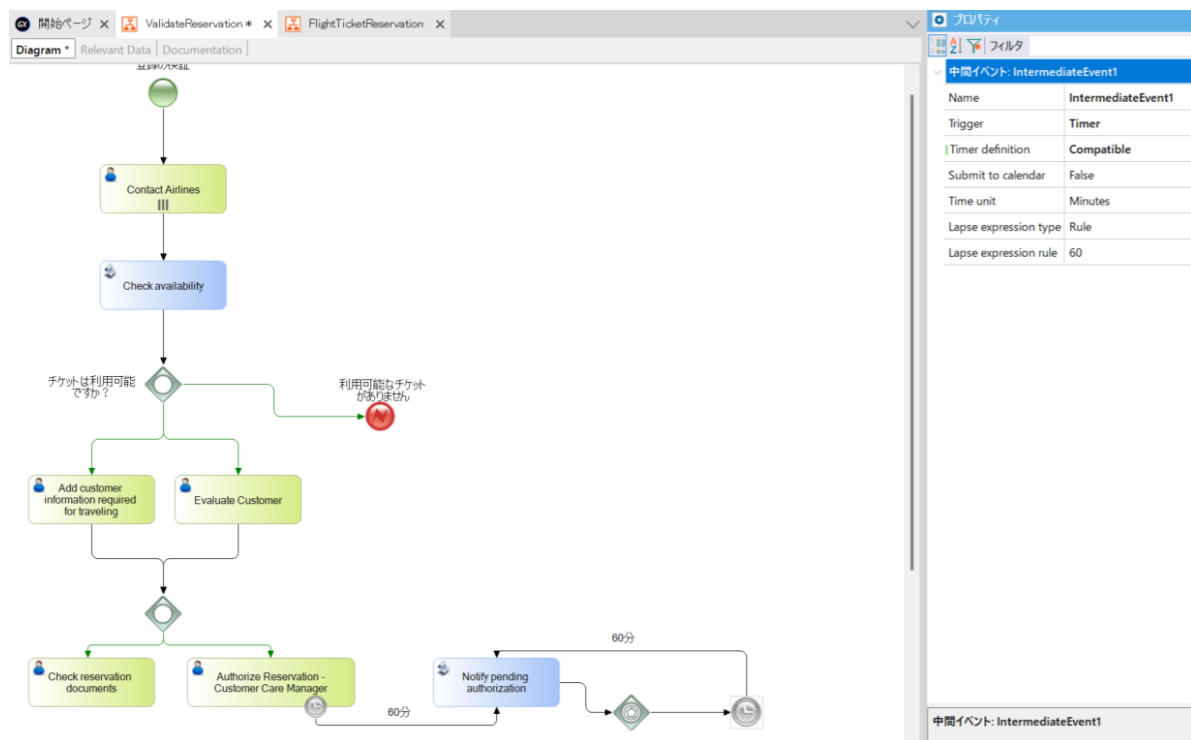
次に、タイマータイプの中間イベントを使用します。タスクに関連付けられたタイマー イベントは、一定時間が経過すると実行を中断するため、タスクの継続時間が制限されます。



通知は自動的に表示されるため、「Notify pending authorization」という名前のスクリプト型タスクを追加し、タイマーイベントから接続します。この接続には、定期的な通知の間隔（この場合は 60 分）がタグ付けされます。また、中間タイマーイベントのプロパティを開き、「Lapse Expression Rule」プロパティに「60」と入力します。



警告が表示されたら、60 分ごとに警告を繰り返すようにします。 これを実現するために、イベントゲートウェイと、出力がバッチタスクに接続される新しいタイマーを使用します。 また、この接続を 60 分に設定し、そのようにタイマープロパティを調整します。



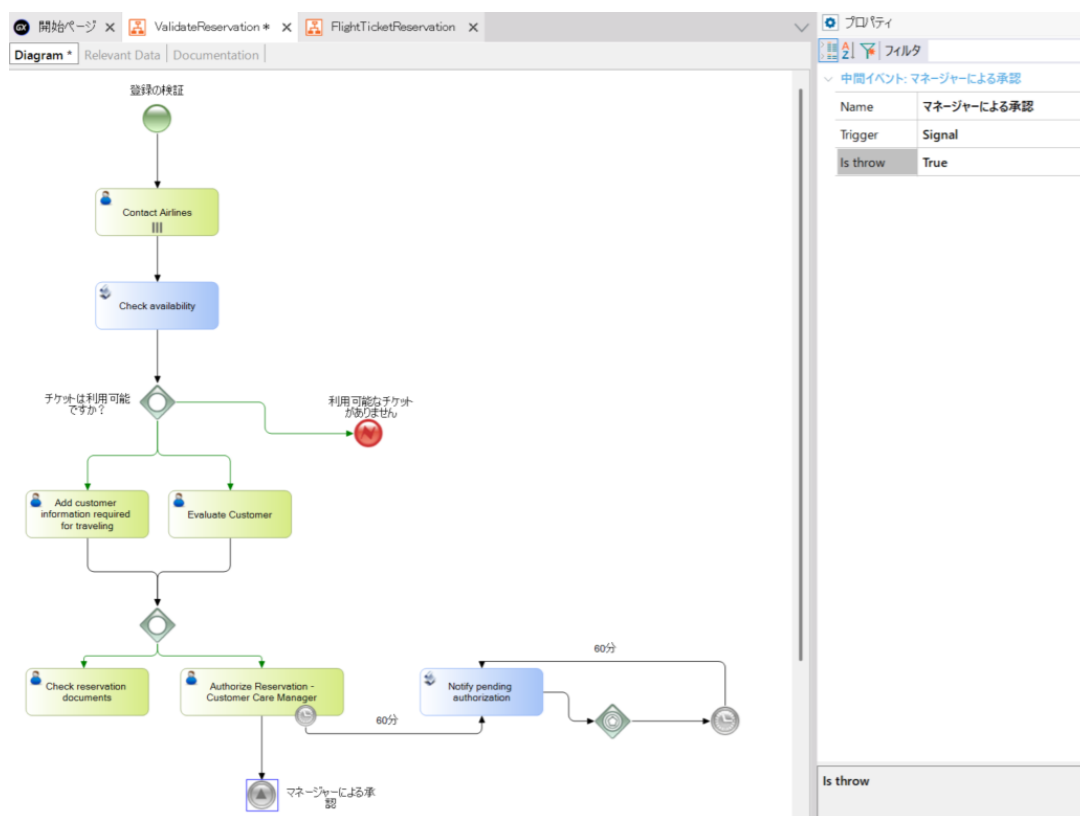
ゲートウェイイベントは、特定のイベントが発生した場合にのみ、プロセスが特定のパスを介して続行されるようにするために使用されます。

たどることができるパスは1つだけであるため、これは排他的ゲートウェイの特殊なケースです。この例では、条件には依存せず、イベントの発生に依存します。

私たちのモデルを続けると、マネージャーが予約を承認した場合、この通知サイクルを中断する必要があります。通知を生成するには、通知を中断するために信号をトリガーおよび検出する必要があります。

シグナルをスローするイベントをモデル化するには、「シグナル中間イベント」のシンボルを追加し、「マネージャーによる承認」という説明を入力し、タスク「Authorize Reservation -Customer Care Manager」から接続します。

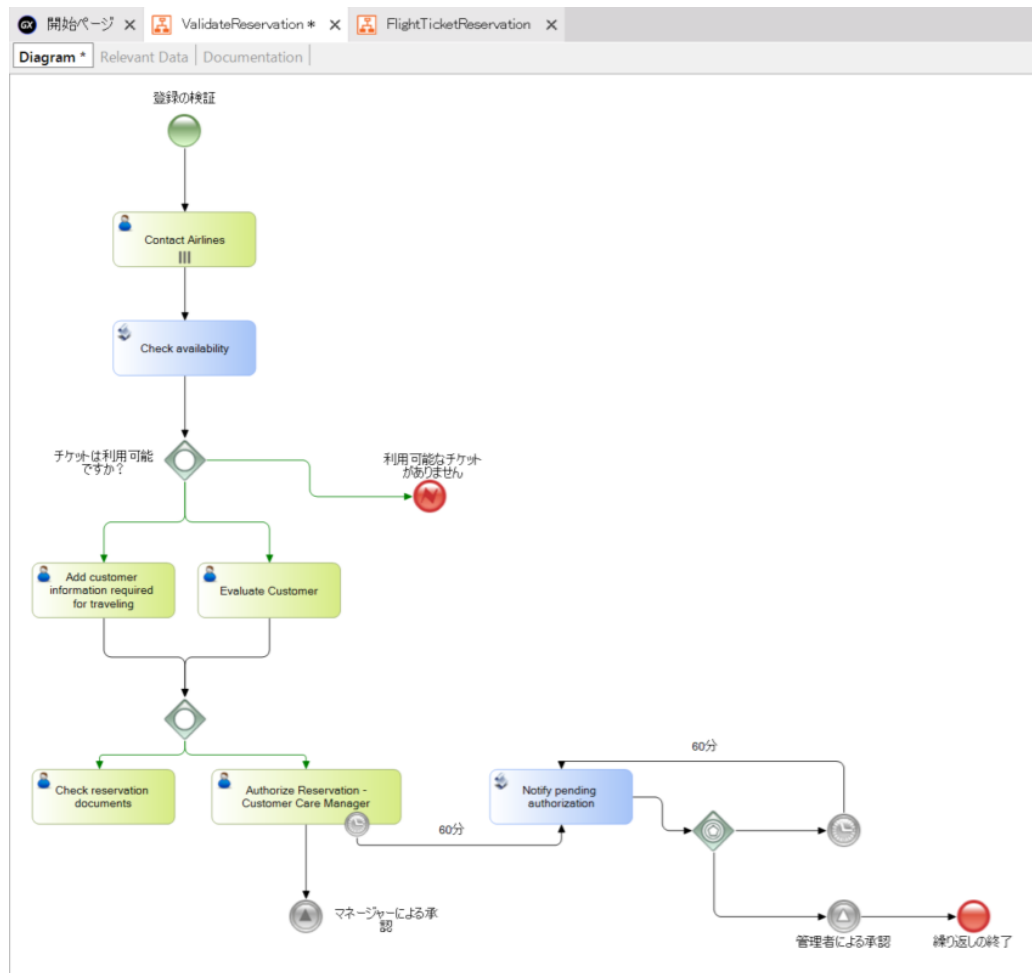
生成される信号を構成したいため、「Is throw」プロパティを変更して「True」に設定します。



次に、通知生成プロセスでこのシグナルをキャッチする必要があります。

「管理者による承認」というシグナルで生成されたものと同じ記述の「シグナル中間イベント」をゲートウェイイベントから接続します。次に、「終了イベント/なし」を追加し、「signal」 イベントから接続します。

このシグナル中間イベントにはシグナルをキャプチャする機能があるため、「Is throw」プロパティは「False」に設定します。



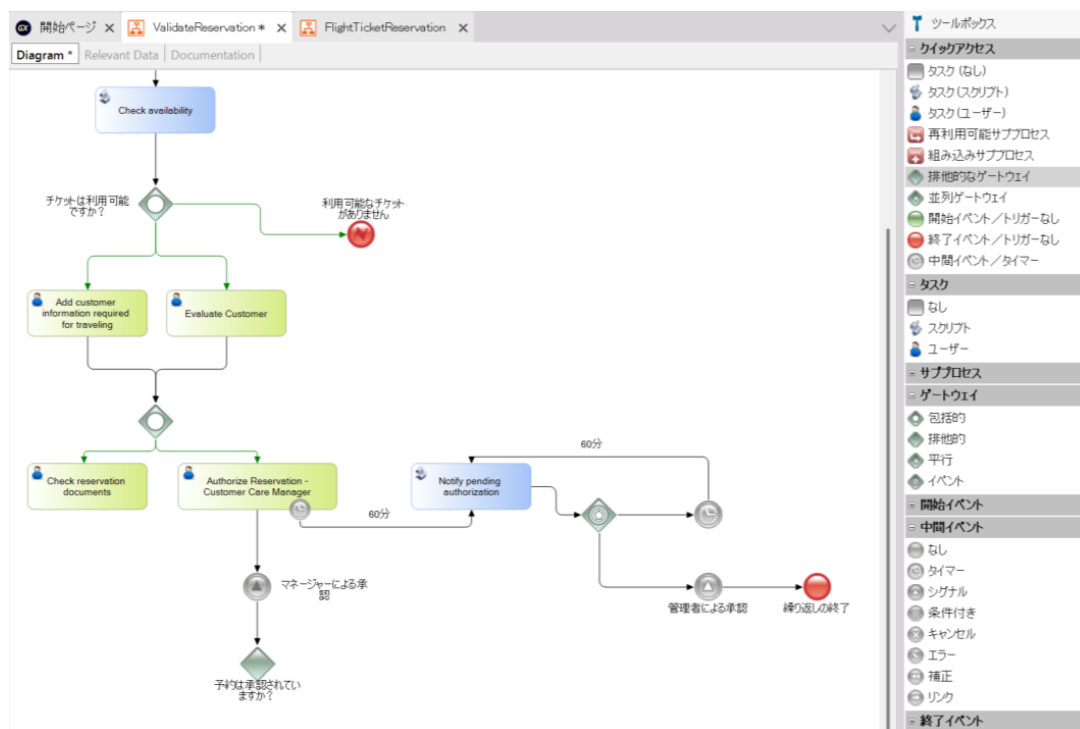


すべての種類の間イベントはイベントをキャッチでき、一部の間イベントはイベントをスローすることもできます。プロセスフローがキャッチタイプのイベントに到達すると、予期されるイベントが発生するまでプロセスが停止します。一方、フローがスロータイプのイベントに到達すると、すぐにスローされて、フローが続行されます。

イベントをキャッチするイベントの種類には枠線のみのアイコンが表示され、イベントをスローするイベントには塗りつぶされたアイコンが表示されます。これは、追加した 2 つのシグナルイベントで確認できます。

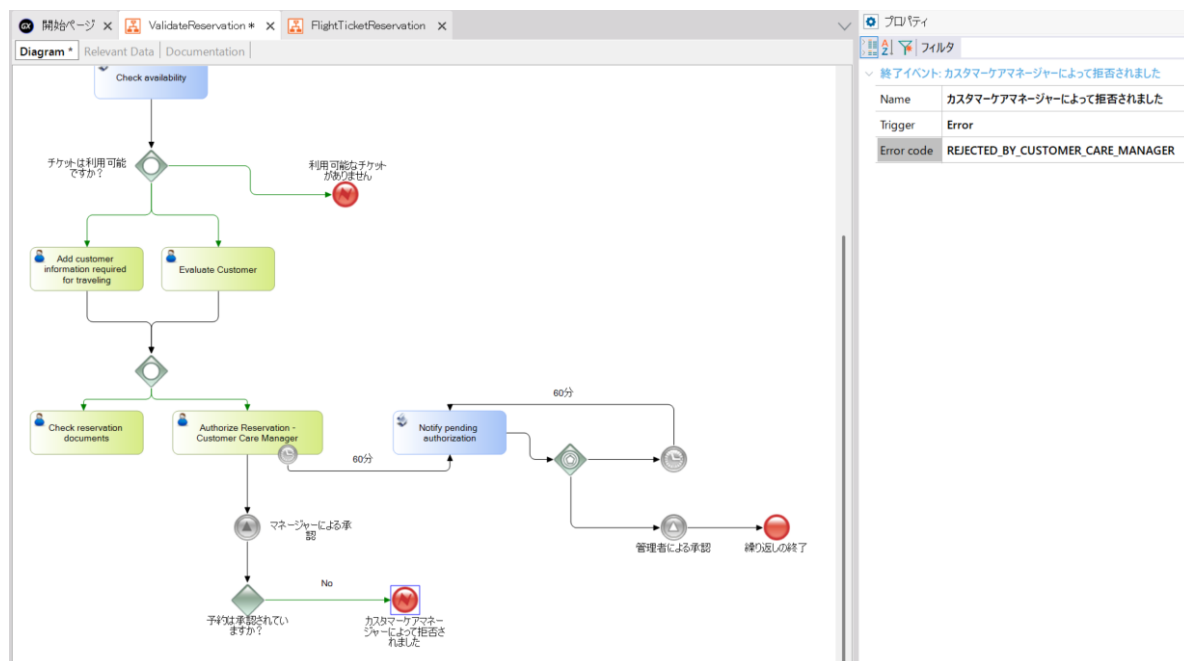
プロセスを続けると、顧客ケアマネージャーが予約を検査した後、承認または拒否される場合があります。

これを表現するために、「予約は承認されていますか?」という説明を持つ排他的なゲートウェイを追加し、シグナル中間イベントから接続します。



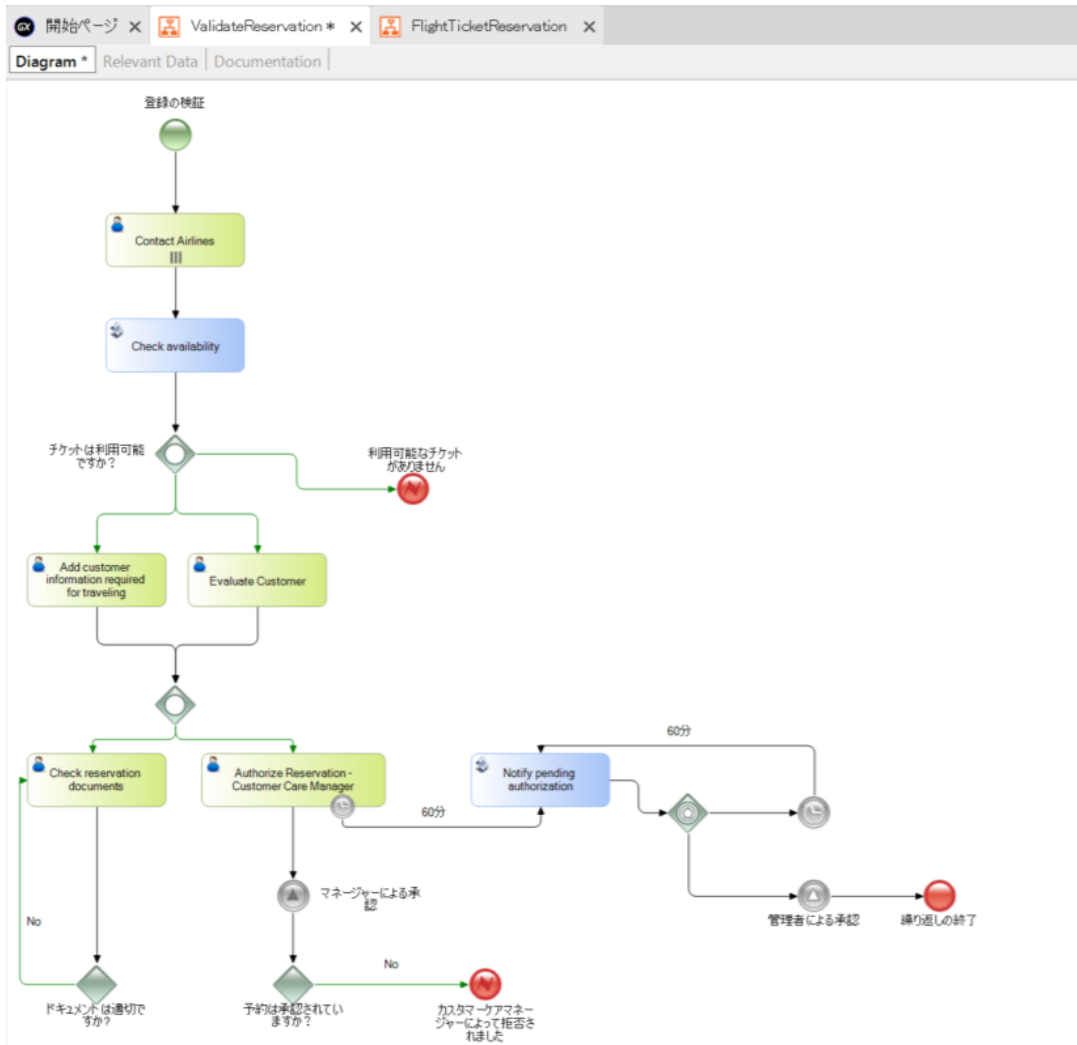
予約が承認されない場合は、「FlightTicketReservation」プロセスにエラー信号を送信して、顧客に拒否することを知する必要があります。 これを実現するには、排他的なゲートウェイから接続されるエラー終了イベントを追加し、接続に「No」、終了イベントに「カスタマーケアマネージャーによって拒否されました」という説明を追加します。

そして「Error code」プロパティに「REJECTED\_BY\_CUSTOMER\_CARE\_MANAGER」と入力します。



提供されたドキュメントを確認するタスクを完了した後、ドキュメントが正しくない場合があります。 この場合、システムはデータ収集タスクに戻る必要があります。 このチェックを実行するには、「ドキュメントは適切ですか?」という説明を持つ排他的なゲートウェイを追加し、タスク「Check reservation documents」から接続します。

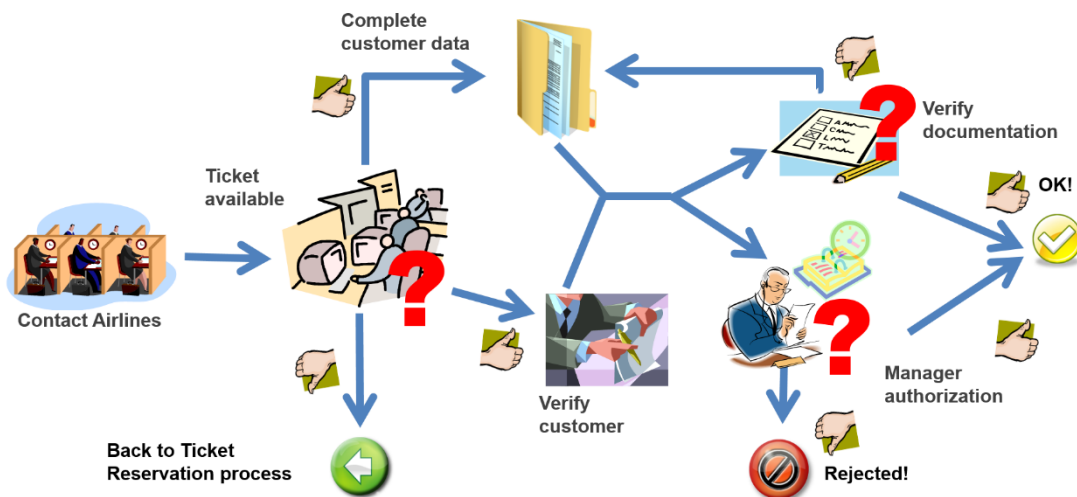
最後に、ゲートウェイを「Add customer information required for traveling」タスクに接続し、コネクタに「No」を追加します。



書類が正常に検証され、カスタマーケアマネージャーが承認したら、予約は有効であると見なされます。

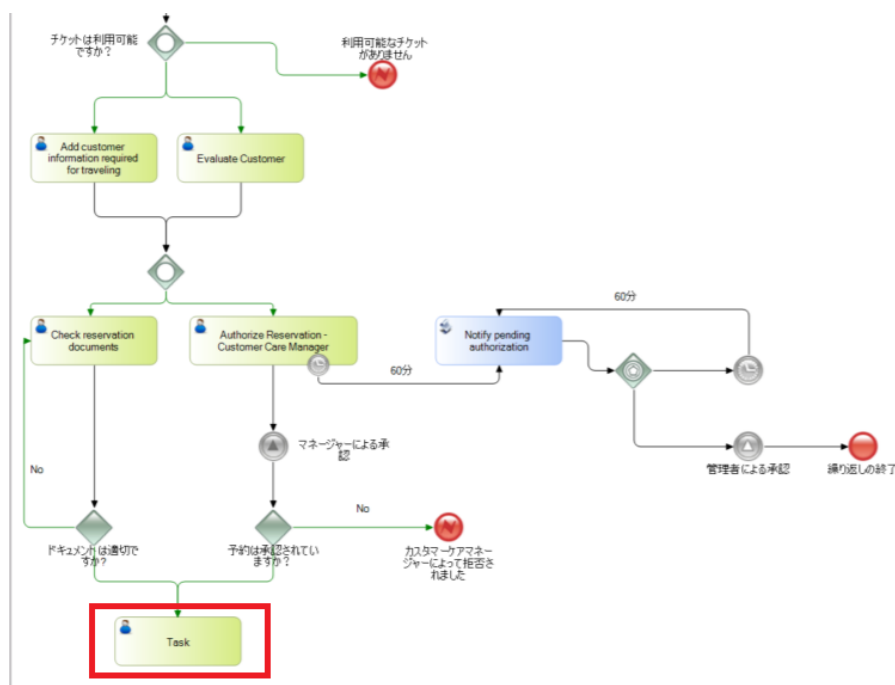
GeneXus

### Validate Reservation process

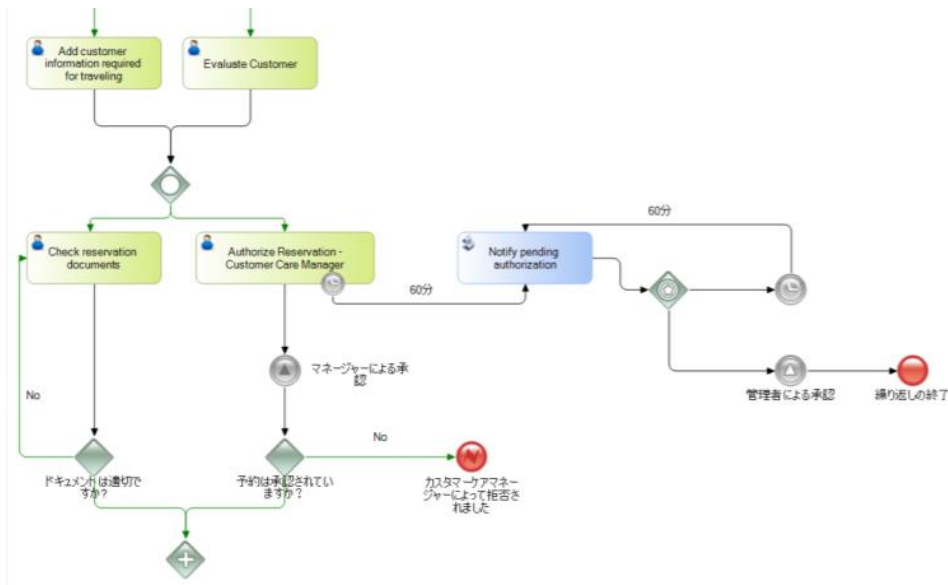


続行するには、両方のタスクを完了する必要があります。 必要に応じて、システムはタスクが終了するまで待つ必要があります。

単純に両方のパスを結合するのではなく、同期が必要です。もし、下図の実装の場合、分岐したパスのうち、一方のタスクが実行されると、次のタスクが実行されます。次に、もう一方のパスのタスクが実行されると、結合に続くタスクがもう一度実行されます。

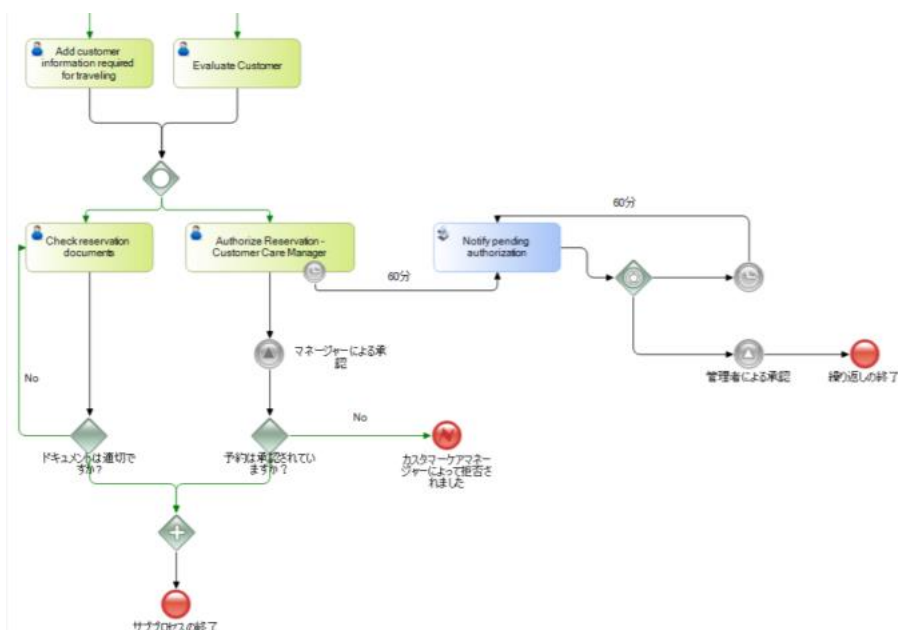


両方のパスを待機する必要があるこの同期を実行するには、並列ゲートウェイを追加し、排他的なゲートウェイから次のタスクへ進みます。

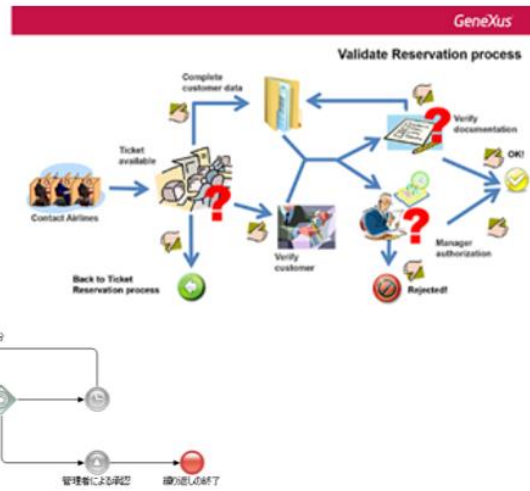
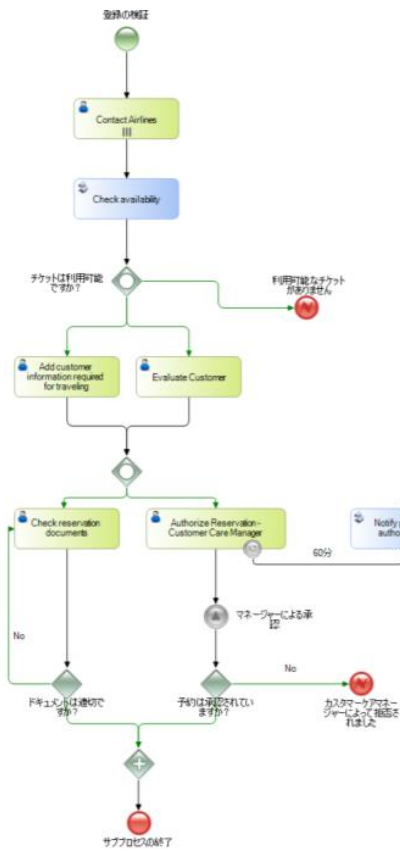


排他的なゲートウェイから出てくる両方のパスがデフォルトパスであることを示すために、対応する「Condition Type」プロパティをデフォルトに設定します。

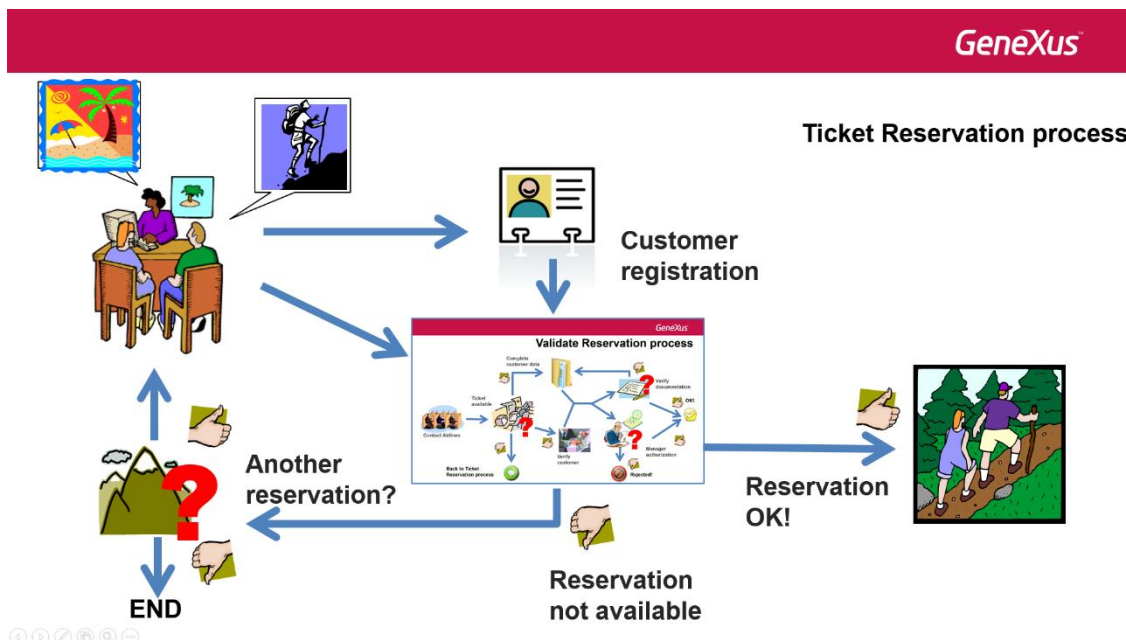
両方のパスが終了したら、チケット予約の検証プロセスを終了する必要があるため、「終了イベント/トリガーなし」を配置し、並行ゲートウェイから接続します。サブプロセスの終了と説明に定義します。



ダイアグラムを右クリックし、フロー図全体が表示されるまでズームアウトしてズームを調整します。



このようにして、チケット予約検証サブプロセスのダイアグラムが完成しました…  
これは航空券予約の主要なプロセスの一部です。



ここでは、モデリング – パート 1 と合わせて、プロセス モデリングの概要、BPMN 標準の簡単な紹介、この標準に準拠した図を構築するために GeneXus が提供する利点を説明しました。

次からは、自動化、監視、最適化、展開、メンテナンスなど、ビジネスプロセスモデリングの他の側面について説明します。