Inclusive Gateways

The inclusive gateway simplifies the implementation of “and-or” logic in a process flow.

Gateways serve to split or join process flows. A split gateway is a point where a single inbound sequence flow diverges into two or more outbound branches. The inclusive (OR-split) gateway corresponds to an “and-or” decision where one or more alternative outflows are to be processed in parallel.

Let’s compare the OR-split gateway with the exclusive (XOR-split) gateway. Both are conditional. However, the OR-split gateway can lead to the activation of one or several outbound branches, while the XOR-split gateway can only activate one outbound branch.

A join gateway is where multiple inbound branches of a process flow merge into a single outbound branch. So, let’s compare the inclusive (OR-join) gateway with the parallel (AND-join) gateway. Both synchronize tokens from inbound sequence flows before proceeding. However, the OR-join gateway knows to wait for tokens from active inbound flows. In contrast, the AND-join gateway waits for a token from all inbound sequence flows before proceeding.

Consider the case in which inventory is maintained at two different warehouses: Warehouse A and Warehouse B. When an order is received, one of three possible conditions arises:

- All items are found in Warehouse A.
- All items are found in Warehouse B.
- Some items are found in Warehouse A, and some in Warehouse B.

We could model the process using a combination of exclusive and parallel gateways.

- In all cases, the process is instantiated when an order is received. Next, the location of ordered items is determined.
- If all items are in Warehouse A, the items are picked from Warehouse A, shipped, and the process ends.
- If all items are in Warehouse B, the process flow looks almost identical.
- However, if some items are in Warehouse A and the rest in Warehouse B, we must embed a parallel structure within the exclusive structure.

This modeling approach works, but it’s a little complicated. As the number of options increases, it can become difficult to follow.

Alternatively, consider a much simpler model using inclusive gateways. The inclusive OR-split gateway routes the process flow appropriately, whether all items are in Warehouse A, all are in Warehouse B, or some items are in Warehouse A, and some are in Warehouse B.

Note that when you use an inclusive (OR-split) gateway, you should use an inclusive (OR-join) gateway to merge the sequence flows.

- If you were to use an XOR-join gateway, for example, all tokens from inbound branches would pass through without delay. That’s a problem if ordered items are in both warehouses. That’s because the tokens from the two parallel paths would fail to synchronize, and downstream activities and events would be executed twice.
- If, on the other hand, you were to use an AND-join gateway, it would wait for a token from all inbound sequence flows before synchronizing the tokens and allowing the process to proceed. That can cause deadlock in this situation. If ordered items are all located in one

warehouse, only one inbound branch will be activated. However, the AND-join gateway will wait forever for a token from the other—inactive—inbound sequence flow.

The OR-join gateway, however, is smart enough to know when to wait and when to allow a token to proceed.

To sum up, implementing and-or logic using exclusive and parallel gateways can get complicated. The inclusive gateway can make the implementation of and-or logic easier.