

# 6 Events

- ▷ Start Event
- ▷ End Event
- ▷ Intermediate Event



## Activities



A Task is a unit of work, the job to be performed. When marked with a  symbol it indicates a Sub-Process, an activity that can be refined.



A Transaction is a set of activities that logically belong together; it might follow a specified transaction protocol.









An Event Sub-Process is placed into a Process or Sub-Process. It is activated when its start event gets triggered and can interrupt the higher level process context or run in parallel (non-interrupting) depending on the start event.



A Call Activity is a wrapper for a globally defined Task or Process reused in the current Process. A call to a Process is marked with a  symbol.








## Activity Markers

Markers indicate execution behavior of activities:

-  Sub-Process Marker
-  Loop Marker
-  Parallel MI Marker
-  Sequential MI Marker
-  Ad Hoc Marker
-  Compensation Marker

## Task Types

Types specify the nature of the action to be performed:

-  Send Task
-  Receive Task
-  User Task
-  Manual Task
-  Business Rule Task
-  Service Task
-  Script Task



defines the execution order of activities.



is the default branch to be chosen if all other conditions evaluate to false.



has a condition assigned that defines whether or not the flow is used.

## Conversations



A Conversation defines a set of logically related message exchanges.



Conv

Colla

Pool (White Box)

Lane

Lane

Lane

Lane

Lane

Lane

Lane

Lane

## Choreographies



Participant A  
Choreography



































































Participant A  
Sub-Choreography


























































Participant A  
Call

## Events

	Standard	Start	Intermediate	End
Standard				
Event Sub-Process Interrupting				
Event Sub-Process Non-Interrupting				
Catching				
Boundary Interrupting				
Boundary Non-Interrupting				
Throwing				
Standard				
<b>None:</b> Untyped events, indicate start point, state changes or final states.				
<b>Message:</b> Receiving and sending messages.				
<b>Timer:</b> Cyclic timer events, points in time, time spans or timeouts.				
<b>Escalation:</b> Escalating to an higher level of responsibility.				
<b>Conditional:</b> Reacting to changed business conditions or integrating business rules.				
<b>Link:</b> Off-page connectors. Two corresponding link events equal a sequence flow.				
<b>Error:</b> Catching or throwing named errors.				
<b>Cancel:</b> Reacting to cancelled transactions or triggering cancellation.				

## Events

	Start	Intermediate	End
Standard			
Event Sub-Process Interrupting			
Event Sub-Process Non-Interrupting			
Catching			
Boundary Interrupting			
Boundary Non-Interrupting			
Throwing			
Standard			
<b>None:</b> Untyped events, indicate start point, state changes or final states.			
<b>Message:</b> Receiving and sending messages.			
<b>Timer:</b> Cyclic timer events, points in time, time spans or timeouts.			
<b>Escalation:</b> Escalating to higher level of responsibility.			
<b>Link:</b> Off page connectors. In corresponding link events use a sequence flow.			
<b>Error:</b> Catching or throwing named errors.			
<b>Cancel:</b> Reacting to cancelled transactions or triggering cancellation.			
<b>Compensation:</b> Handling or opening compensation.			
<b>Signal:</b> Signaling across different processes. A signal thrown once is caught multiple times.			
<b>Link:</b> Catching one out of a set of events. Throwing all events defined.			
<b>Parallel Multiple:</b> Catching out of a set of parallel events.			
<b>Terminate:</b> Triggering the immediate termination of a process.			

## Data



A Data Object represents information flowing through the process, such as business documents, e-mails, or letters.



A Collection Data Object represents a collection of information, e.g., a list of order items.



A Data Input is an external input for the entire process. A kind of input parameter.



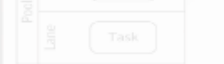
A Data Output is data result of the entire process. A kind of output parameter.



A Data Association is used to associate data elements to Activities, Processes and Global Tasks.



A Data Store is a place where the process can read or write data, e.g., a database or a filing cabinet. It persists beyond the lifetime of the process instance.



Pools (Participants) and Lanes represent responsibilities for activities in a process. A pool or a lane can be an organization, a role, or a system. Lanes subdivide pools.



Message Flow symbolizes information flow across organizational boundaries. Message flow can be attached to pools, activities, or message events. The Message Flow can be decorated with an envelope depicting the content of the message.



The order of message exchanges can be specified by combining message flow and sequence flow.



# Events

something that happens during a business process

## ▷ Starts




Triggers a process and marks the beginning of the flow

## ▷ Ends

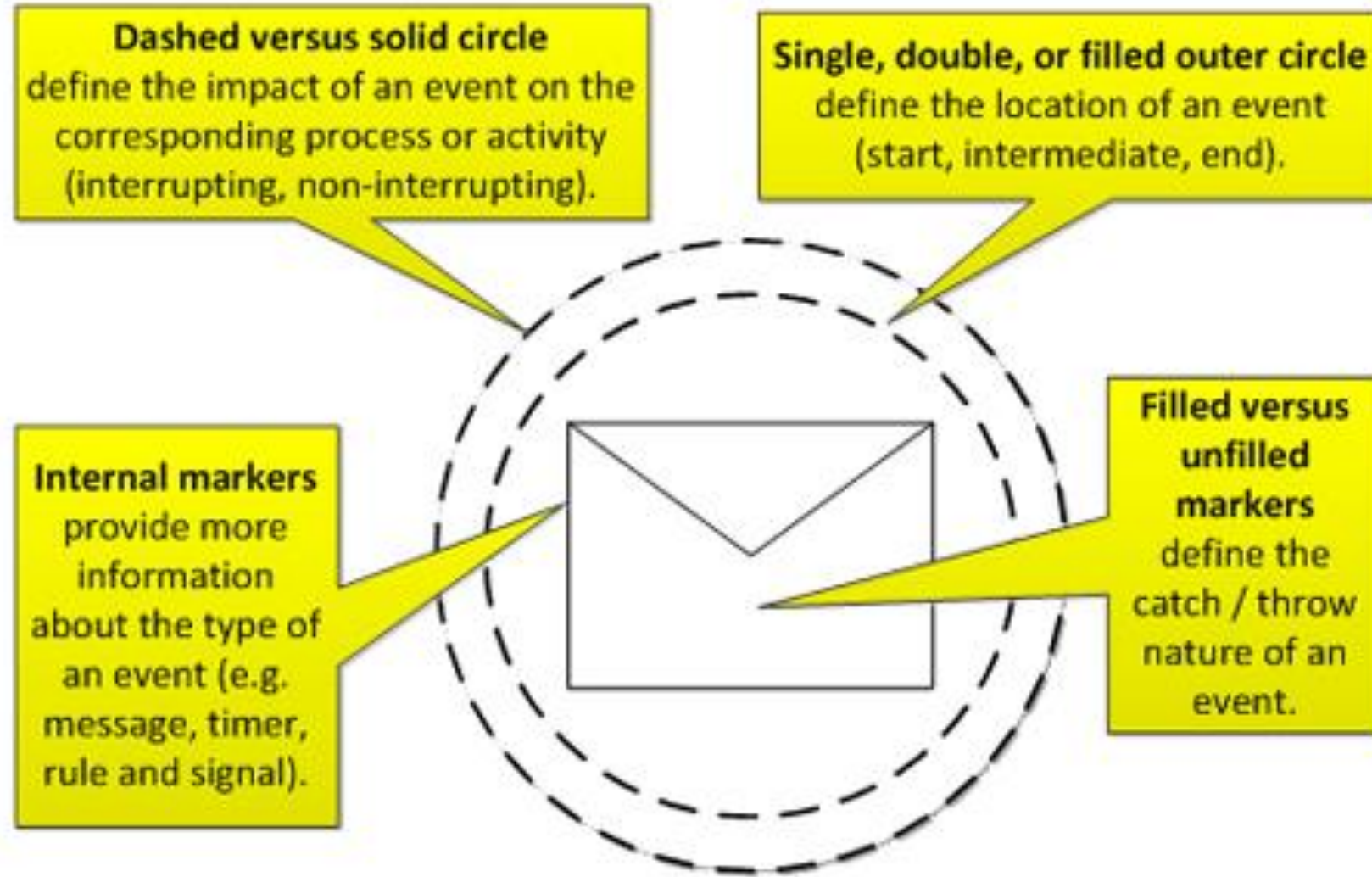
Marks the process end , after all Tokens reached , the process instance is terminated.

## ▷ Intermediate (Delays, or interrupts the flow)

Occurs between the start and the end event and influences the flow

Start event	Intermediate event	End event
		

# Events - Markers



## Events – Real World Example



Events like Triggers

1

Start Event

# Start Event

the Start Event indicates where a particular Process will start.




In terms of Sequence Flows, the Start Event starts the flow of the Process, and thus, will not have any incoming Sequence Flows

Start Event is **OPTIONAL**, Process MAY (is NOT REQUIRED to) have a Start Event

If there is an End Event, then there MUST be at least one Start Event.





All Flow Objects that do not have an incoming Sequence Flow SHALL be **instantiated** when the Process is instantiated

# Start Event – triggers Types

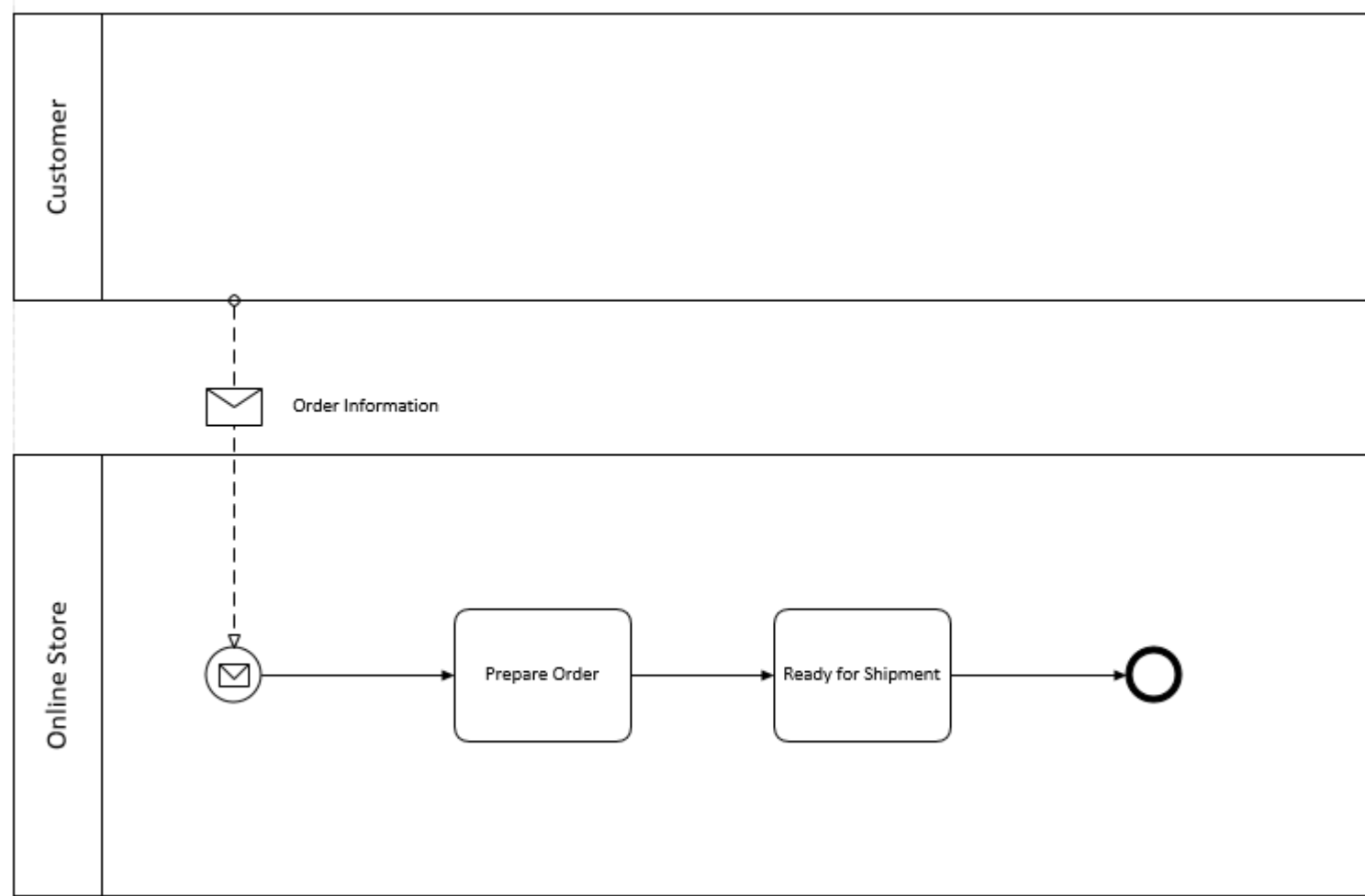
Trigger name	Representation	Description
None		The none start event does not have a defined trigger.
Message		This trigger starts the process by receiving a message from a participant.
Timer		This trigger starts the process in a specific time-date or a specific cycle (e.g. every Friday).



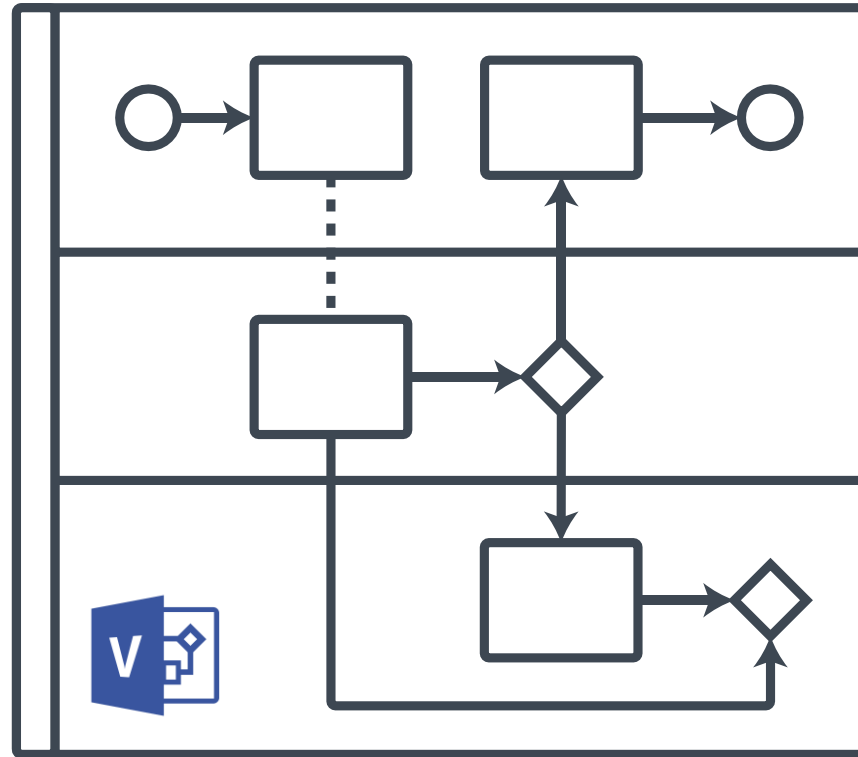
# Start Event – triggers Types

Trigger name	Representation	Description
Error		This trigger starts an in-line event sub-process when an error occurs. Note that this trigger can only be used with an event-sub-process.
Escalation		This trigger starts or not to start an in-line event sub-process when the constraint specified is not satisfied. Note that this trigger can only be used with an event-sub-process.
Compensation		This trigger starts an in-line event sub-process when an compensation occurs, which requires undoing some steps. Note that this trigger can only be used with an event-sub-process.
Conditional		This trigger starts the process when a specific condition becomes true.

# Start Event – BPMN Example



# Start Event - BPMN Hands-on



BPMN Demo using Microsoft Visio

2

End Event

# End Event

End Event indicates where a Process will end.





In terms of Sequence Flows, the End Event ends the flow of the Process , and thus, will not have any outgoing Sequence Flows—no Sequence Flow can connect from an End Event

End Event is **OPTIONAL** , Process MAY (is NOT REQUIRED to) have a End Event




If there is an Start Event, then there **MUST** be at least one End Event.

**All the tokens** that were generated within the Process **MUST be consumed** by an End Event before the Process has been completed.

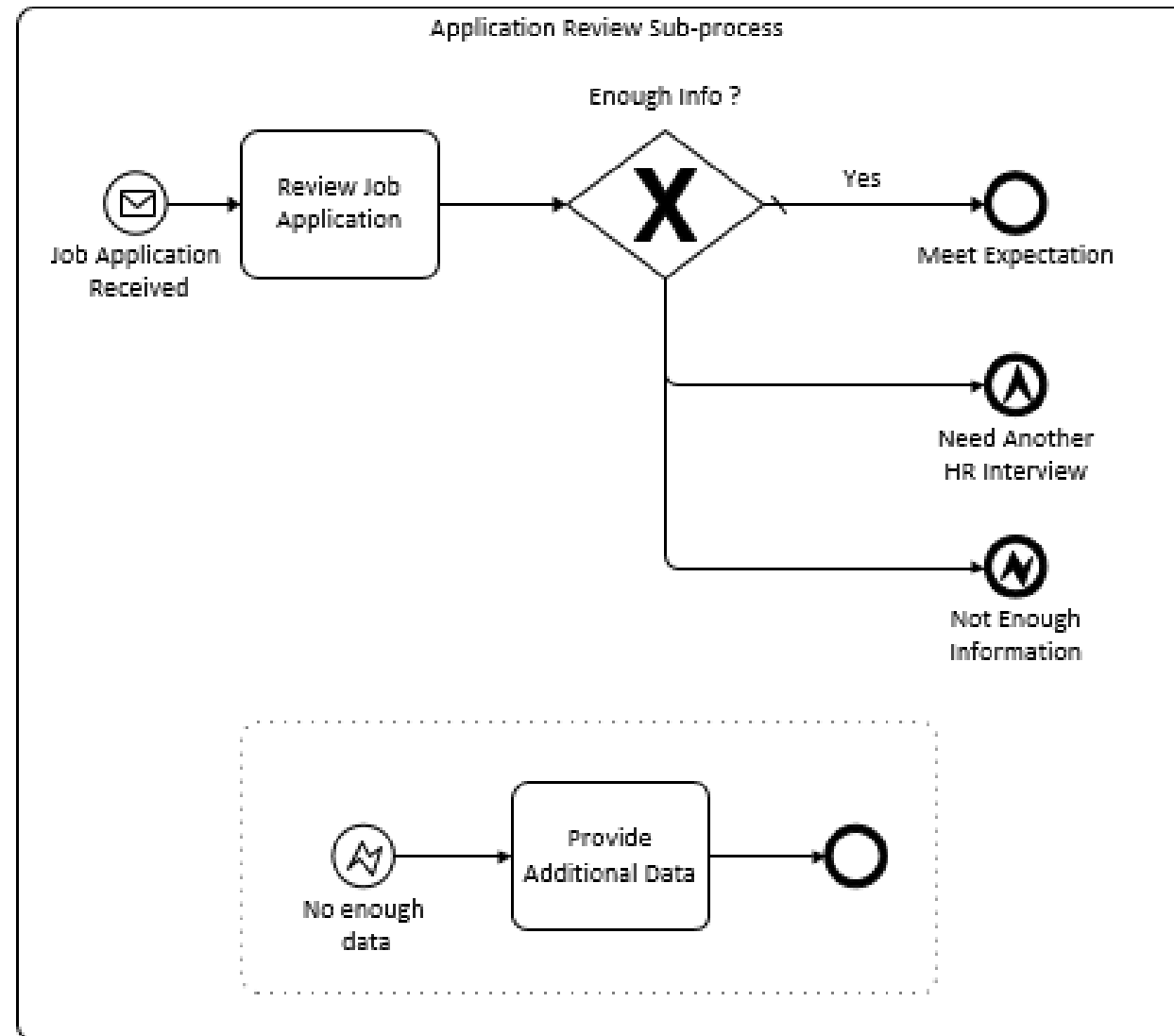
# End Event – triggers Types

Trigger name	Representation	Description
None		The none end event does not have a defined result.
Message		This result ends the process by sending a message to a participant.
Error		This result indicates the generation of a named error when the process ends.
Terminal		This result indicates that all activities in the process should be immediately ended.

# End Event – triggers Types

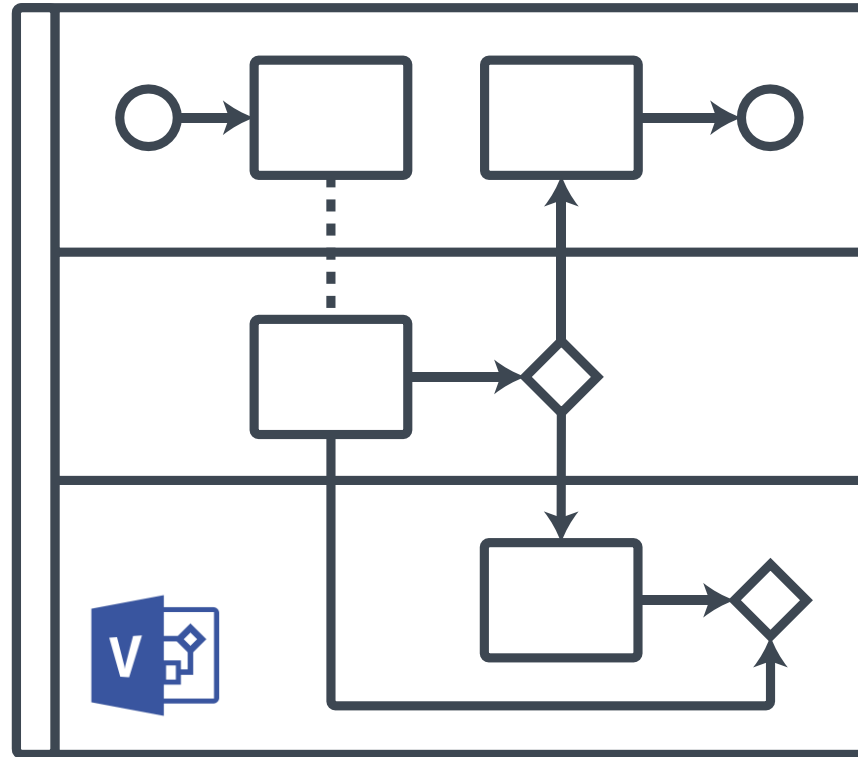
Trigger name	Representation	Description
Escalation		This result indicates the trigger of escalation when the process ends.
Cancel		This result indicates that the transaction should be cancelled.
Compensation		This result indicates the need of compensation, which require undoing some steps.

# End Event – BPMN Example





## End Event - BPMN Hands-on



BPMN Demo using Microsoft Visio

3

Intermediate Event

# Intermediate Event





Intermediate Event indicates where something happens (an Event) somewhere between the start and end of a Process

It will affect the flow of the Process, but will not start or (directly) terminate the Process





Intermediate Events can be used to:

- Show where Messages are expected or sent within the Process,
- Show delays are expected within the Process
- Disrupt the normal flow through exception handling, or
- Show the extra work needed for compensation

# Intermediate Event – triggers Types

Trigger name	Representation	Description
None		The none intermediate event does not have a defined trigger. It is used to indicate change of state in the process. You can only use a none intermediate event in a normal flow.
Message		This trigger represents either a send or receive of message
Timer		This trigger acts as a delay mechanism on a specific date-time or cycle (e.g. every Friday). You can only use a timer intermediate event in a normal flow.
Error		This trigger reacts to a named error or to any error if no name is specified.

# Intermediate Event – triggers Types

















































Trigger name	Representation	Description
Escalation		The trigger indicates where an escalation is raised. You can only use an escalation intermediate event in a normal flow.
Cancel		This trigger will be fired when a cancel end event is reached within the transaction sub-process. It also shall be triggered if a Transaction Protocol “Cancel” message has been received while the Transaction is being performed.
Compensation		The trigger indicates the need of compensation.
Conditional		The event will be triggered when the condition specified become true.

# Throw & Catch Events

You can set an event to be catch or throw.

**Catch** means to react to a trigger,

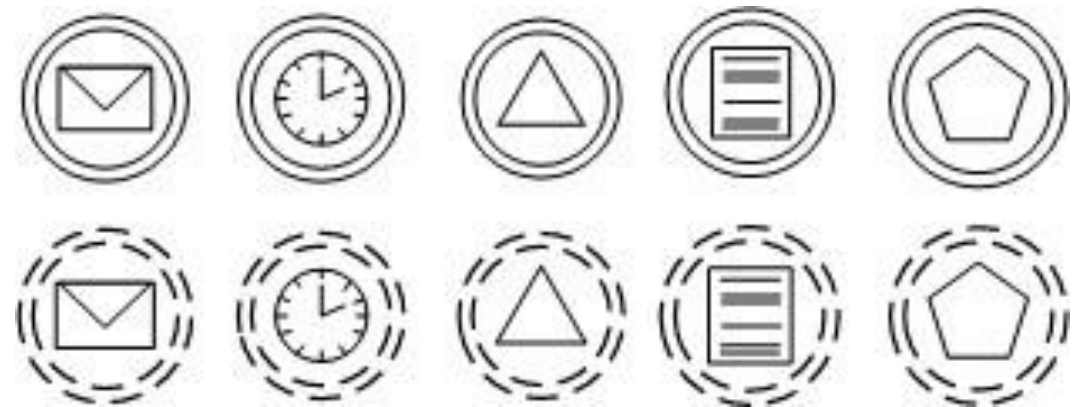
while **Throw** means to create a trigger

	“Catching”		“Throwing”		Non-Interrupting	
Message						
Timer						
Error						
Escalation						
Cancel						
Compensation						
Conditional						
Link						
Signal						
Terminate						
Multiple						
Parallel Multiple						

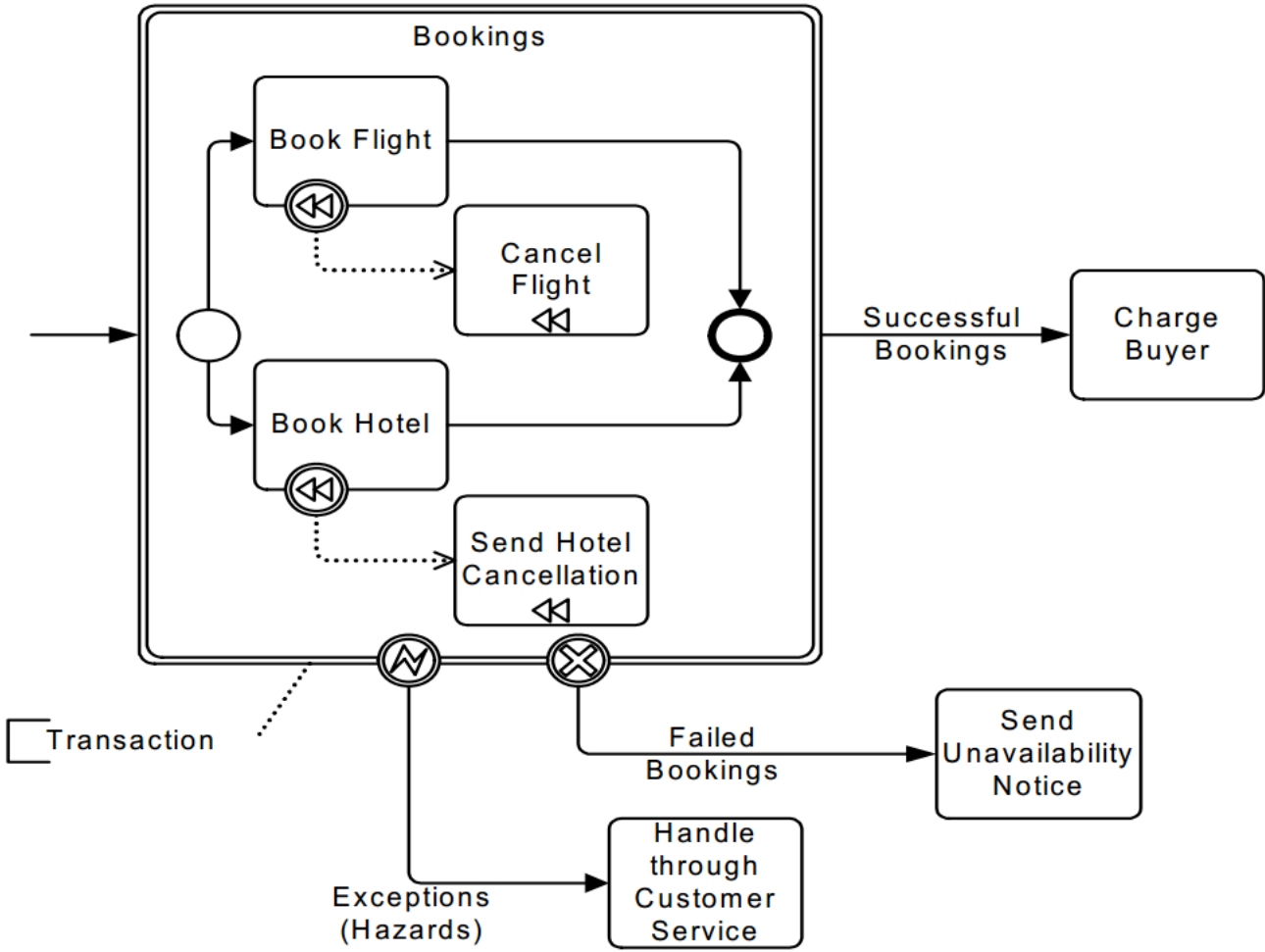
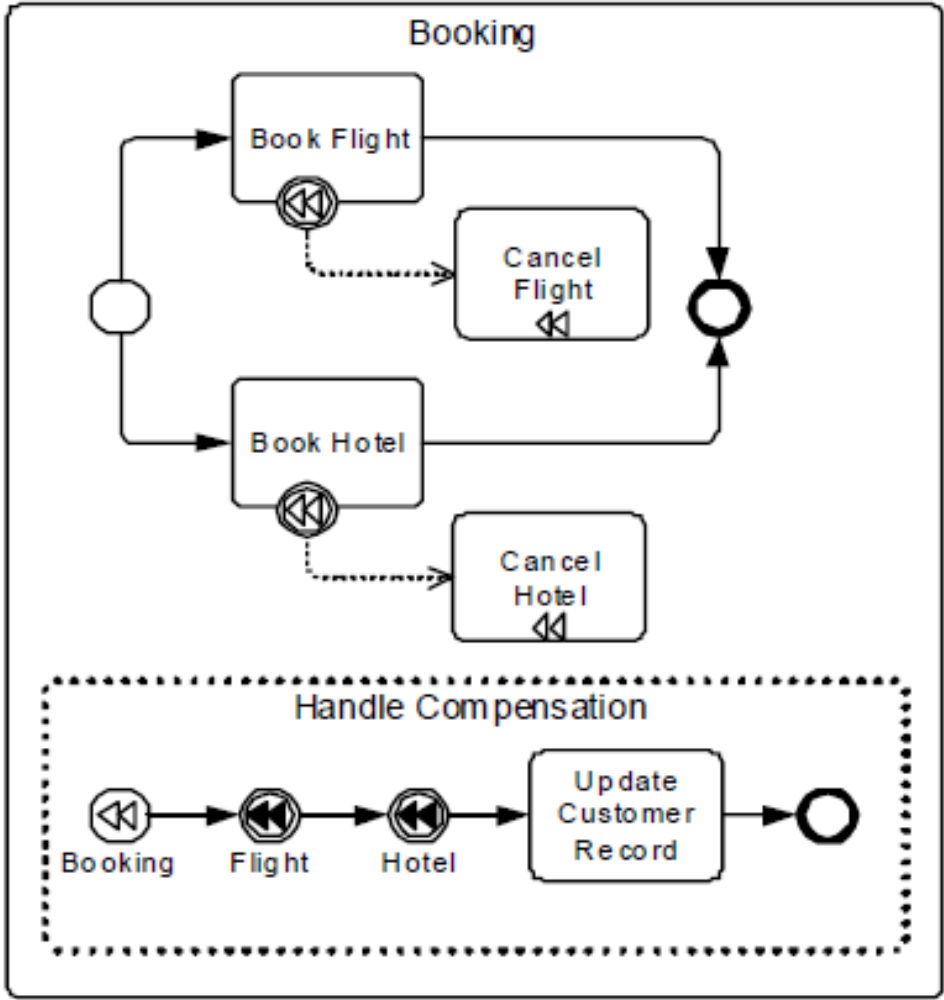
# Interrupting & Non- Interrupting

**Non-interrupting** events :

The difference is that when the event is triggered, the exceptional flow  
**occurs in parallel** to the main flow

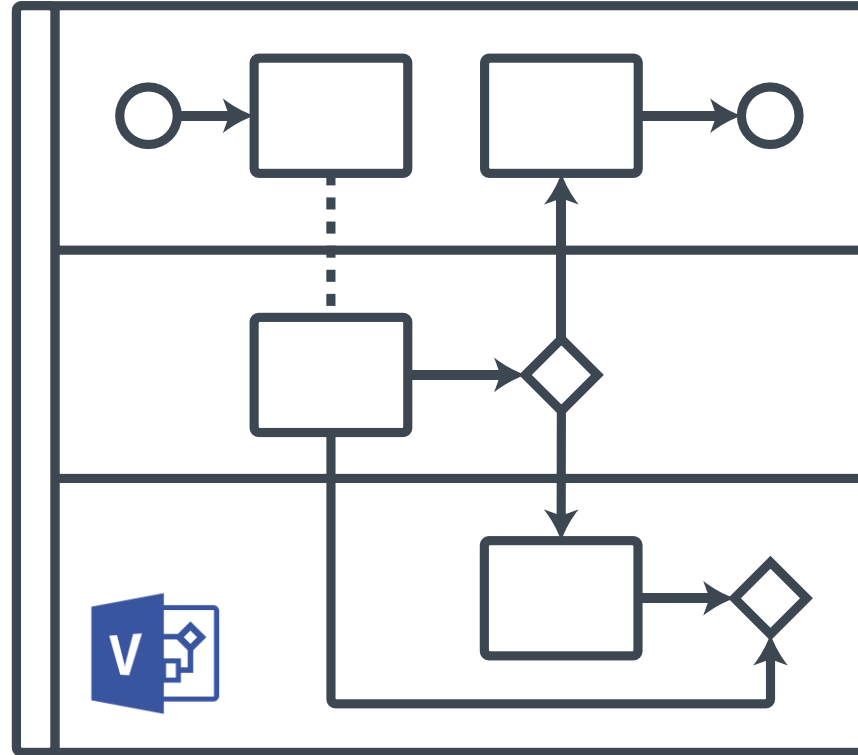


# Intermediate Event - BPMN Example





# Intermediate Event - BPMN Hands-on



BPMN Demo using Microsoft Visio



*To Be continue .. 😊*

# Thanks!

## Any questions?

You can find me at:  
@MohamedZekus  
eng.mohamedzakarya@gmail.com