

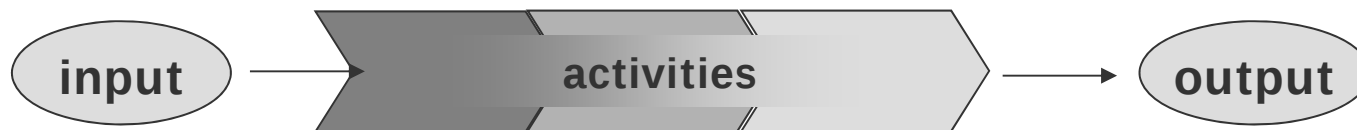
Business Process Modelling with BPMN

Knut Hinkelmann

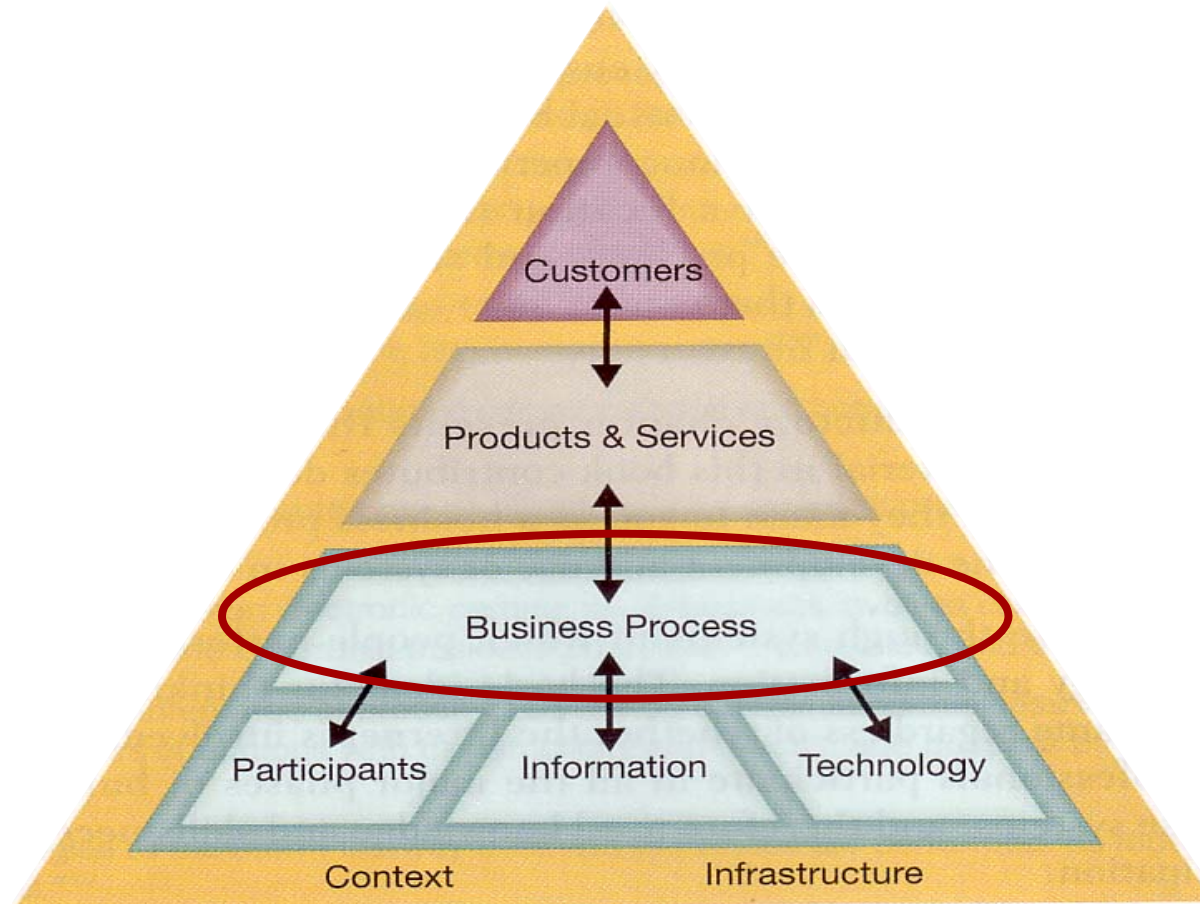


Process

- There are many definitions of a business process. Here are some important characteristics of a process
- A process is a systematic set of activities
 - ◆ which manipulate or transport material or information
 - ◆ in order to accomplish a specific purpose or objective
 - ◆ creating value for a customer (internal or external)
- Most processes
 - ◆ require some sort of input and
 - ◆ use and/or consume resources and
 - ◆ produce some sort of output – a service or a product



Work-Centered Analysis



Business Process Models and the Zachman Framework

■ Business Process Models are part of the HOW column of the Zachman Framework

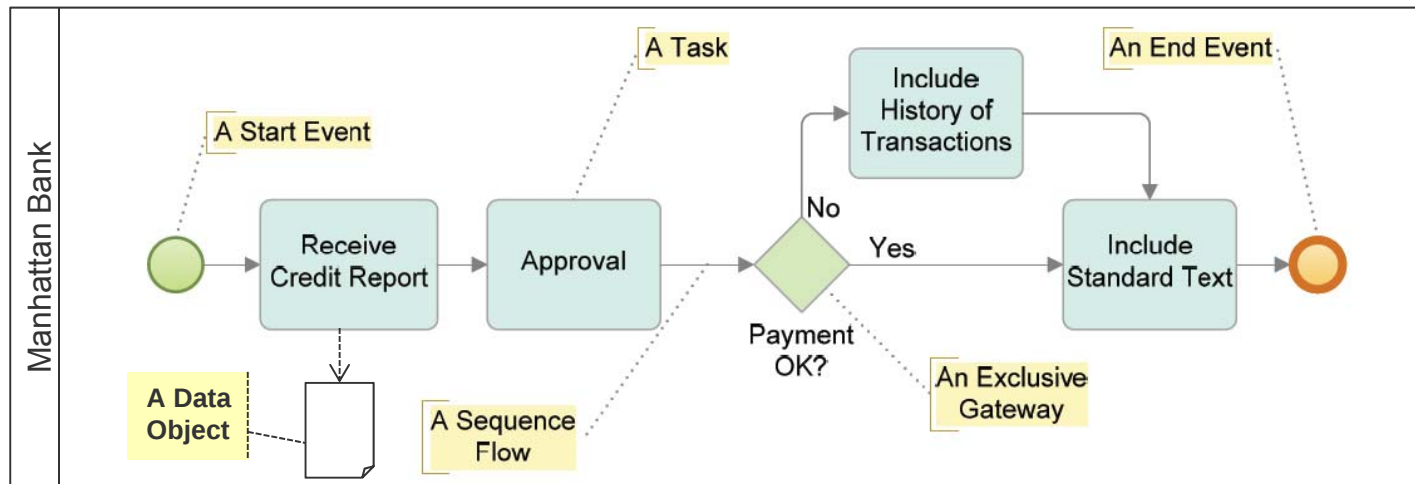
- ◆ Usually we use the term **"business process model"** if we model the process from the business perspective
- ◆ A **"workflow model"** or **"process implementation"** represents the process from the IT perspective



The Workflow Management Coalition defines "workflow" as the automation of a business process

What is BPMN?

- BPMN is a graphical modeling notation for business processes that is independent of a specific implementation environment



- BPMN was officially adopted as an OMG specification in 2006, updated in 2008 and now available in version 2.0 (<http://www.omg.org/spec/BPMN/2.0/>)
- BPMN provides a standardized bridge for the gap between the *business process design* and *process implementation*

Objectives of BPMN

- BPMN models processes from the business perspective
- BPMN has two somehow contradictory objectives
 - ◆ to provide an **easy to use process modeling notation**, accessible to business users and business analysts
 - ◆ provide facilities to **translate models into an executable form** (such as BPEL – Business Process Execution Language)
- To meet the requirements of the first goal, BPMN is structured with a
 - ◆ small set of elements (Activities, Events and Gateways) that have
 - ◆ distinct shape (rectangle, circle and diamond).

Events



Activities



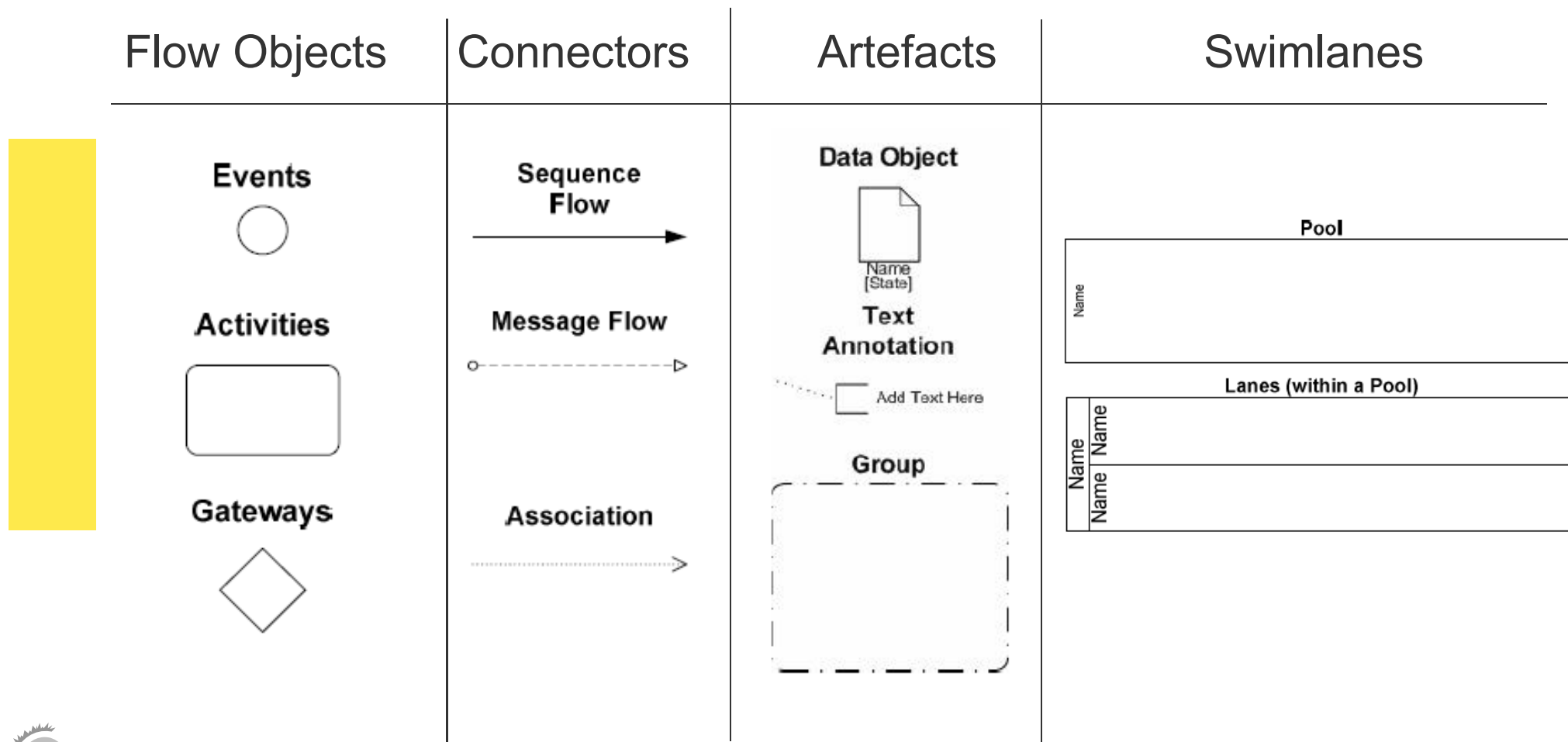
Gateways



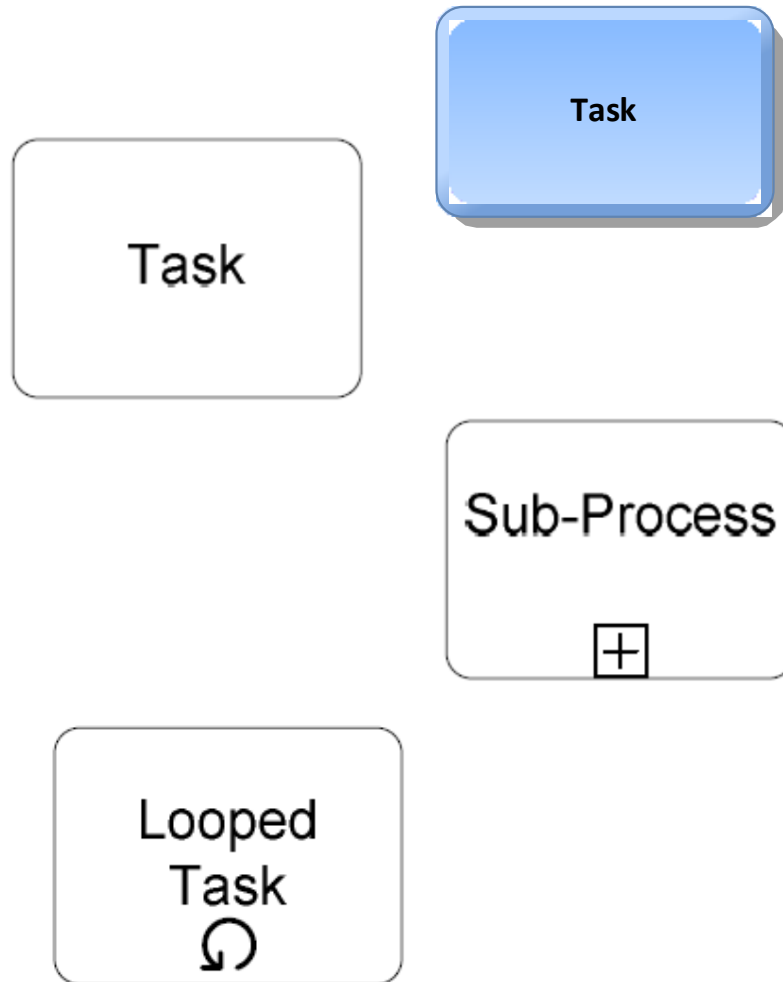
This small set supports simplicity and readability of models

Elements of BPMN

Elements of BPMN can be divided into 4 categories:



Activities



- An activity is work that is performed within a business process.
- Typically an activity is one step of a larger business process.
- Activities are rounded rectangles (some tools use colors)
- There are two types of activities :
 - ◆ **Task** (atomic)
 - ◆ **Sub-Process** (compound, i.e. consisting of several activities, marked by a [+])
- Activities can be performed once or can have internally defined loops

Names and Description of Tasks

Every task has attributes, which capture details of the work.

Here we only mention name and description.

■ **name:**

- ◆ typically short – consisting of a verb and an object/resource
- ◆ Example: **Check Reservations**

■ **description**

- ◆ details about the work, what it means, how it is performed. Also the applications used can be mentioned.
- ◆ Example:
Check the reservation book to see whether the reservation exists. Verify that the party arrived before the reservation time

The actual attributes available depend on the modeling tool

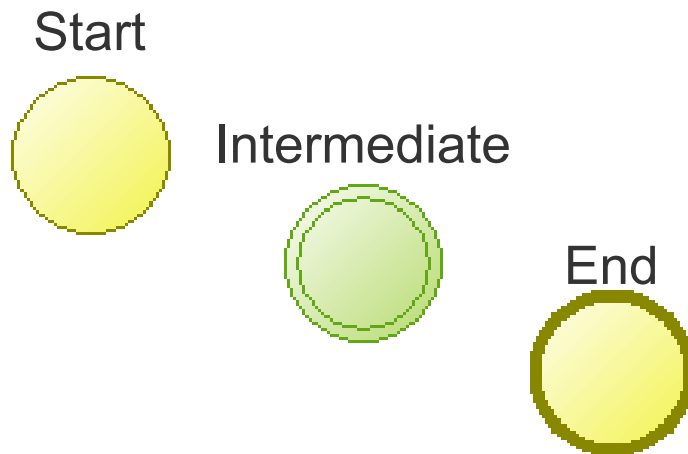
(Bridgeland & Zahavi 2009, p. 107f)

Sequence Flow

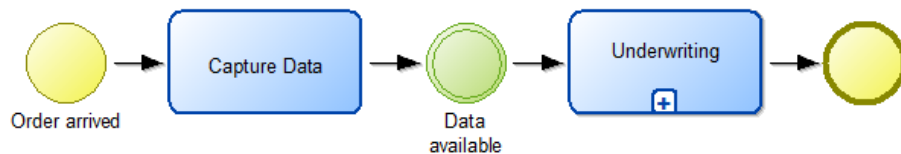
- A Sequence Flow is used to show the order that activities will be performed in a Process
- The source and target must be one of the following objects
 - ◆ Events
 - ◆ Activities
 - ◆ Gateways
- In a sequence of activities, the subsequent activity is performed after the previous activities is finished



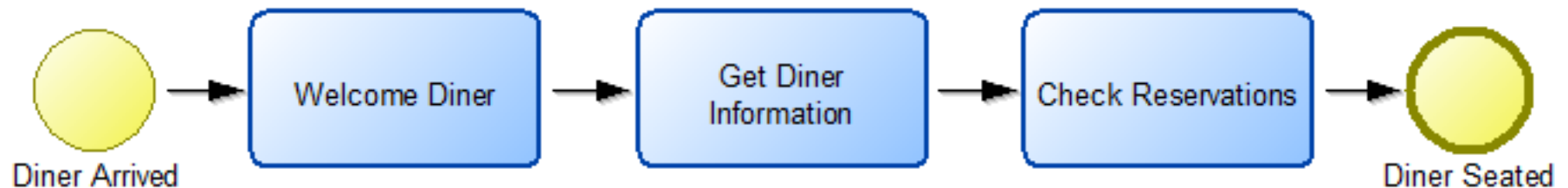
Events



- A process begins with a start event and ends with an end event
- Events are states that affect the flow of the process
 - ◆ they start, interrupt and finish the flow
 - ◆ they can trigger an activity or are its result
- Events are represented as circles. The type of boundary determines the type of Event
 - ◆ Start Event
 - ◆ Intermediate Event
 - ◆ End-Event
- Events can have descriptions, just as tasks.

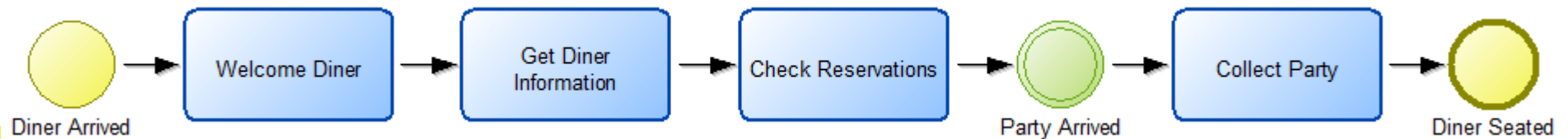


Example: A simple End-to-End Process



- **Diner Arrives** is the start event
- **Diner Seated** is the end event
- Note that the names of the events are different from the names of the activities
 - ◆ Activity names are typically imperative sentences, they sound like command. The verb is at the beginning of the name.
 - ◆ Event names are typically declarative sentences, describing a state or something that happens

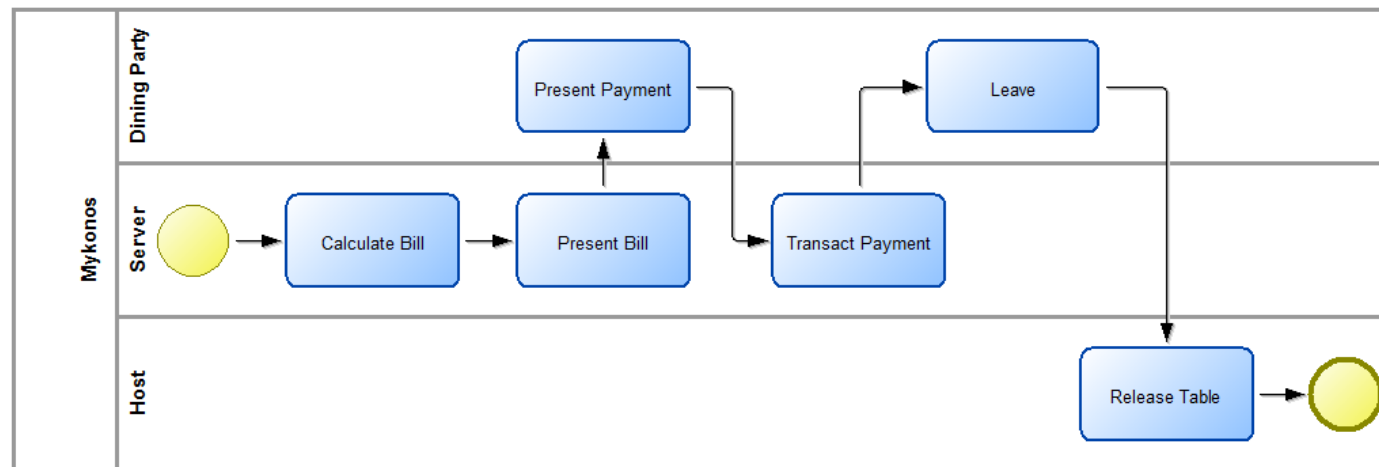
Example with an Intermediate Event



- An intermediate event happens after the process starts and before it ends
- In this example the events models a delay: But when the first diner of a party arrives the host checks the reservations but does not seat the diner until the rest of the party arrives.

Swimlanes – Pools and Lanes

- A business process model graphically shows who performs which activities.
- Pools represent independent organisational units. A pool is a container for a business process.
- Each participant that performs activities in a business process has a lane – a horizontal stripe like a lane in a swimming pool.
 - ◆ A lane can be a role, an organisational unit or a system.



(Bridgeland & Zahavi 2009, p. 110f)



Event-Typen

		None	Message	Timer	Con- ditional	Signal	Escalation	Error	Com- pensation	Multiple	Parallel Multiple	Link	Cancel	Terminate
Start Events	Top-Level													
	Event Sub- Process Interrupting													
	Event Sub- Process Non- Interrupting													
Inter- mediate Events	Catching													
	Throwing													
	Boundary Interrupting													
	Boundary Non- Interrupting													
End Events														



Event Types

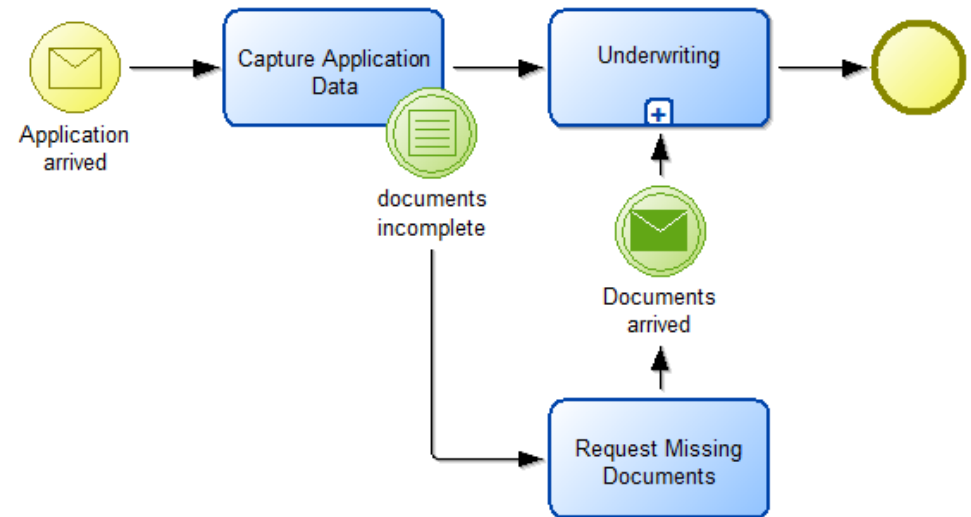
- **None:** Untyped events, indicate start point, state changes or final states.
- **Message:** Receiving and sending messages.
- **Timer:** Cyclic timer events, points in time, time spans or timeouts.
- **Conditional:** Reacting to changed business conditions or integrating business rules.
- **Signal:** Signalling across different processes. A signal thrown can be caught multiple times.
- **Escalation:** Escalating to an higher level of responsibility.
- **Error:** Catching or throwing named errors.
- **Compensation:** Handling or triggering compensation.
- **Multiple:** Catching one out of a set of events. Throwing all events defined
- **Parallel Multiple:** Catching all out of a set of parallel events.
- **Link:** Off-page connectors. Two corresponding link events equal a sequence flow.
- **Cancel:** Reacting to cancelled transactions or triggering cancellation
- **Terminate:** Triggering the immediate termination of a process.

Properties of Events

■ Start-Events:

- ◆ Top-level
- ◆ Event Sub-Process Interrupting
- ◆ Event Sub-Process Non-Interrupting

■ End-Event



■ Intermediate Events

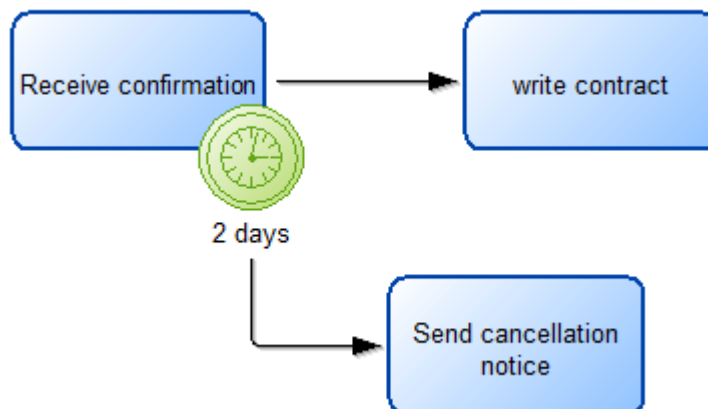
Between Activities:

- ◆ Throwing
- ◆ Catching

On the boundary of activities

- ◆ Boundary Interrupting
- ◆ Boundary Non-Interrupting

Intermediate Events



- Events that are placed within the process flow represent things that happen during the normal operations of the process. They can represent ...

... a «trigger» that initiates an activity – catching

... the result of an activity – throwing

- Events that are attached to the boundary of an activity can occur during the activity. They can ...

... interrupt the activity

... open an additional path without interrupting

Catching and Throwing Events



- A **throwing** intermediate event, with the black icon inside, means the process generates the trigger signal



- A **catching** intermediate event, with the wide icon inside, means the process waits for the trigger signal
 - ◆ A catching event interrupts a process and waits for the trigger signal to arrive
 - ◆ When the trigger signal arrives, the process resumes on the sequence flow out of the event

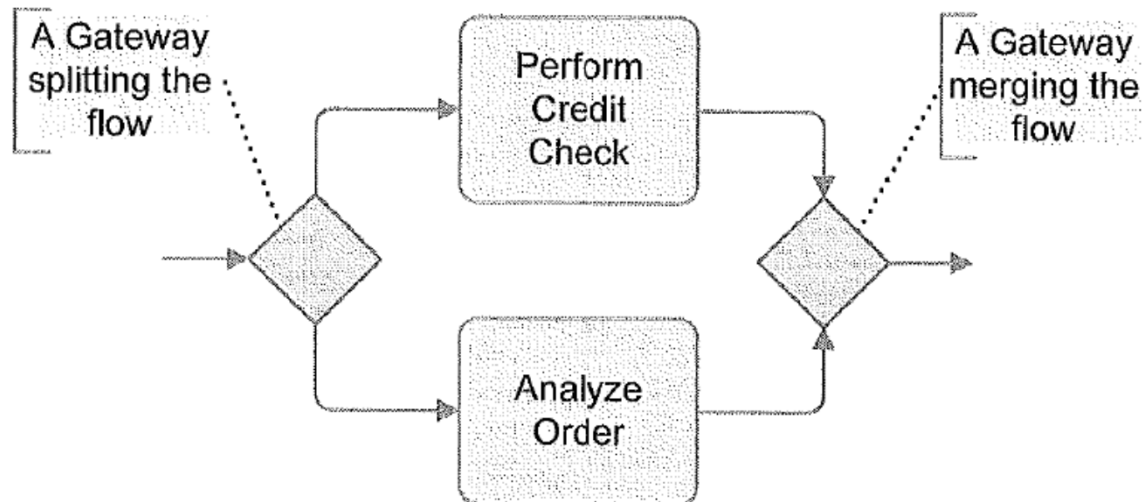
Exercise

The restaurant manager gave you the following description of the reservation at a Mykonos restaurant:

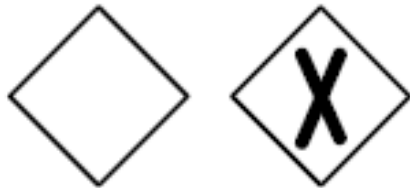
- When the guest sends a request for a reservation, the reservation is written into the reservation book.
- At 7:00 pm the guests are assigned to the tables and then the reservation cards are placed on the tables. At 8:00 pm the restaurant is opened.

Gateways

- **Gateways** model sequence flow alternatives, i.e. they represent points of control
- They split and merge the flow of a Process
- All types of Gateways are diamonds
- The underlying idea is that Gateways are unnecessary if the Sequence Flow does not require controlling



Gateways – Splitting and Merging



Exclusive Gateway: When splitting, it routes the sequence flow to exactly one of the outgoing branches. When merging, it awaits one incoming branch to complete before triggering the outgoing flow.



Event-based Gateway: Sequence flow is routed to the subsequent event/task which happens first.

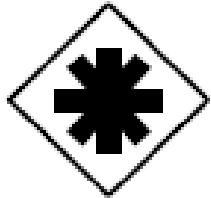


Parallel Gateway (AND): When used to split the sequence flow, all outgoing branches are activated simultaneously. When merging parallel branches it waits for all incoming branches to complete before triggering the outgoing flow.

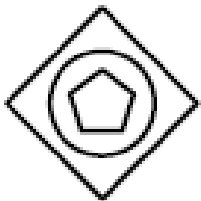


Inclusive Gateway (OR): When splitting, one or more branches are activated. All active incoming branches must complete before merging.

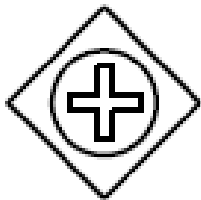
Gateways – Verzweigungen und Vereinigungen



- **Complex Gateway** Complex merging and branching behavior that is not captured by other gateways.



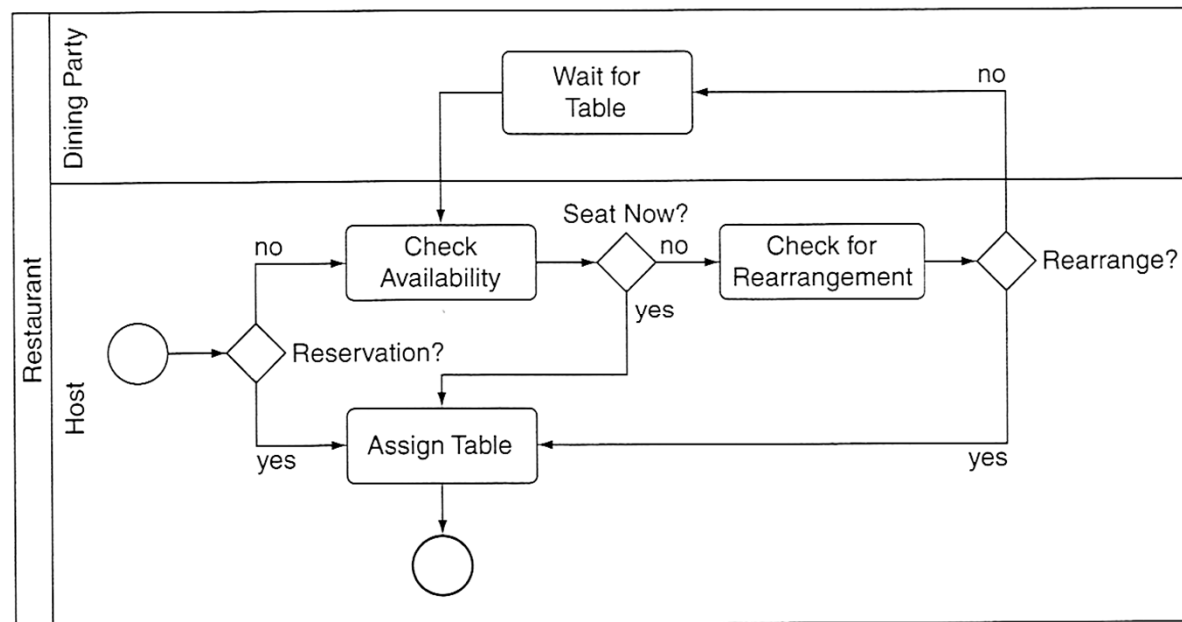
- **Exclusive Event-based Gateway (instantiate)** Each occurrence of a subsequent event starts a new process instance.



- **Parallel Event-based Gateway (instantiate)** The occurrence of all subsequent events starts a new process instance.

Exclusive Gateways

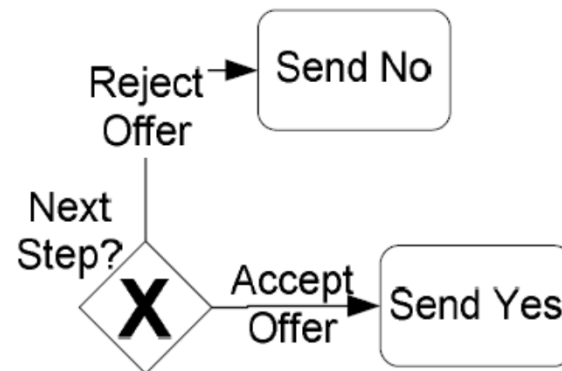
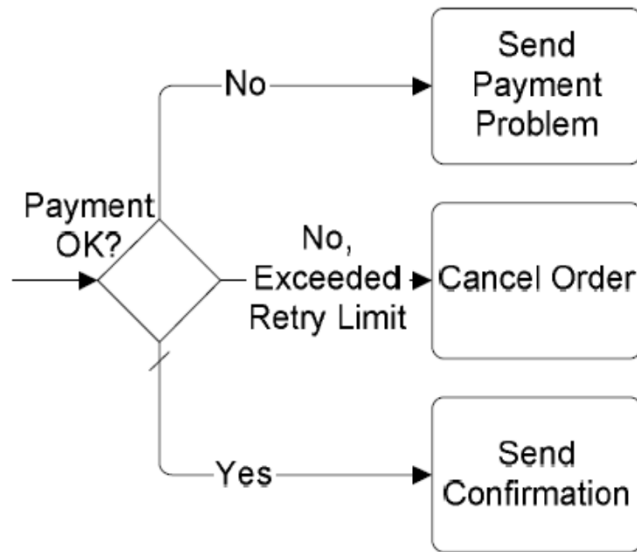
- For exclusive Gateways exactly one of the following sequence flows is selected
- The name of the gateway is a question with the alternative answers to the questions as labels on the outgoing sequence flows



(Bridgeland & Zahavi 2009, p. 113f)

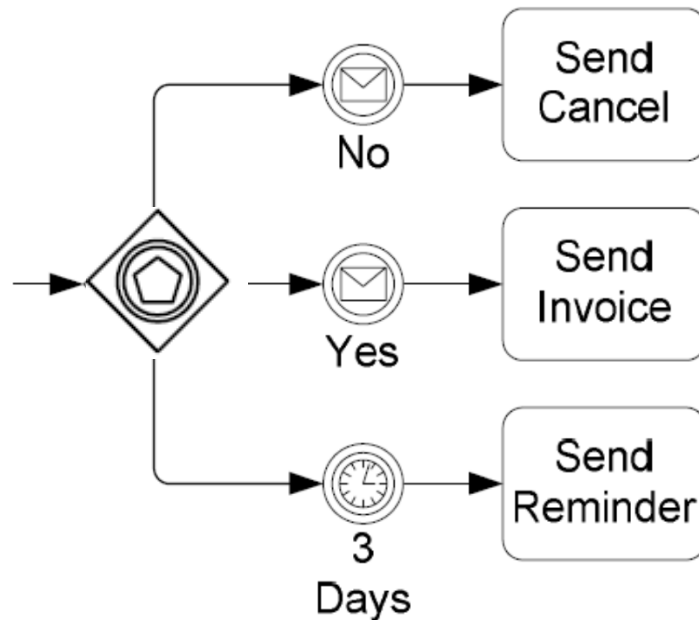


Exclusive Gateways based on Data



- The Gateway (Decision) creates alternative paths based on defined conditions.
- Exclusive Gateways based on Data are the most commonly used Gateways
- They can be shown with or without an internal „X“ marker. Without is the most common use.

Exclusive Gateways based on Events

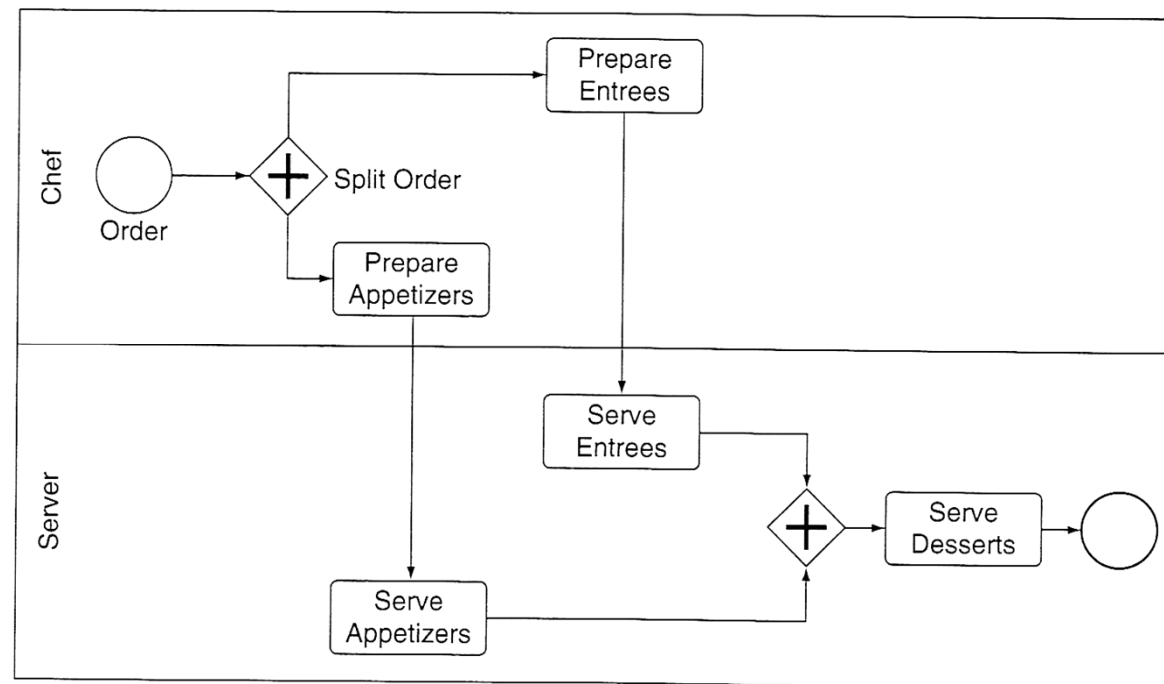


- Alternatives in this Decision are based on events that occur at the point in the process rather than conditions
- The Multiple Intermediate Event is used to identify this Gateway
- The Events that follow the Gateway Diamond determine the chosen path
 - ◆ The first Event triggered wins

Parallel Gateway

■ A parallel gateway

- ◆ starts parallel work, i.e. two (or more) sequence flows that then progress at the same time
- ◆ parallel flows can be joined back together by another parallel gateway

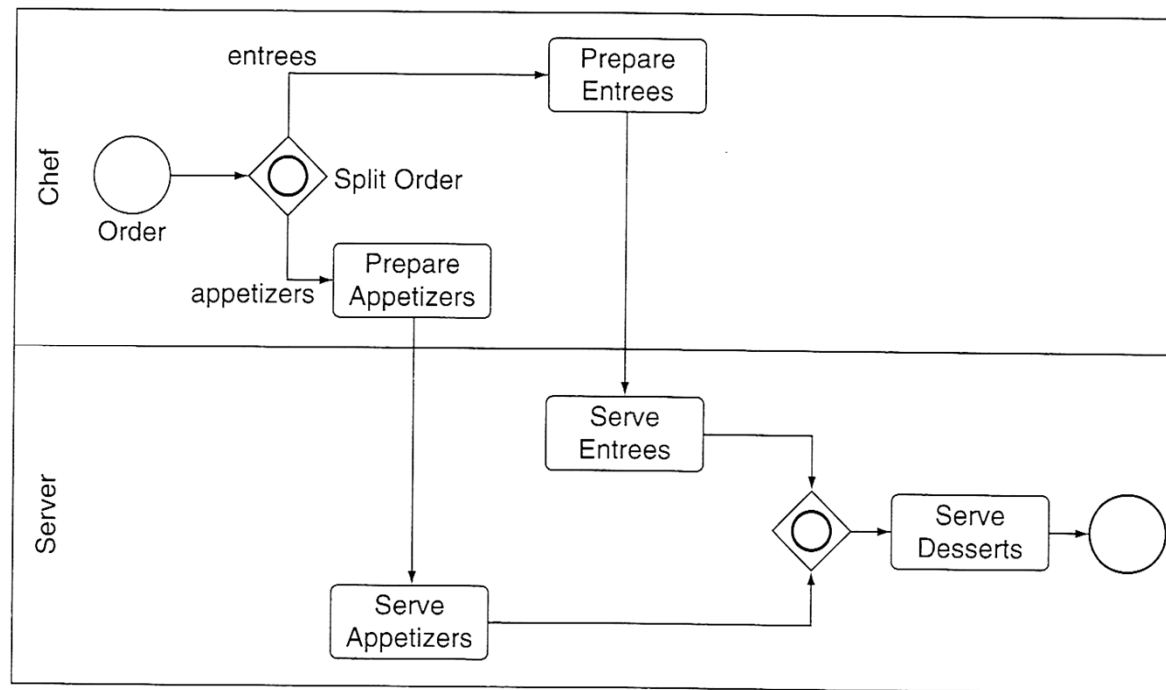


(Bridgeland & Zahavi 2009, p. 114f)



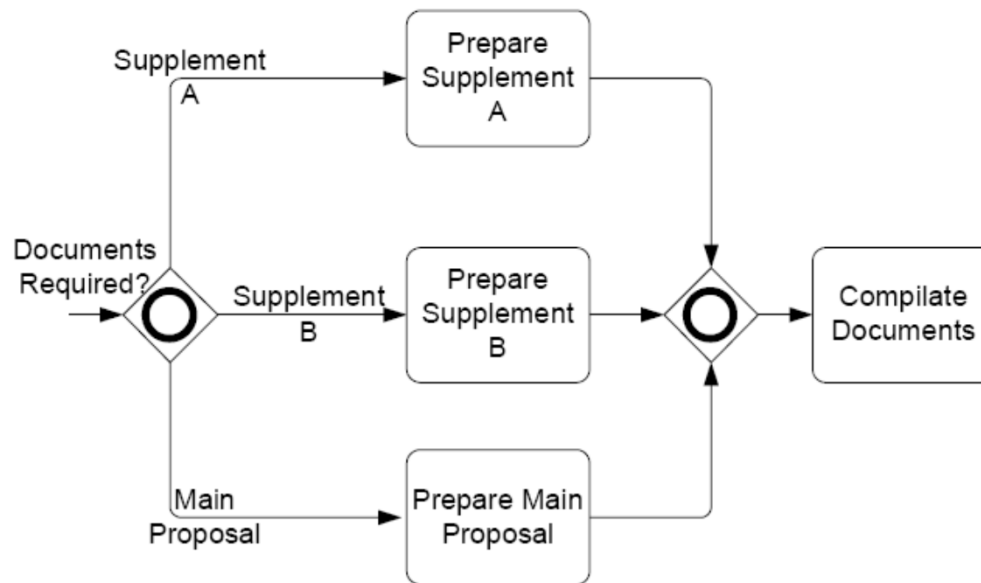
Inclusive Gateway

- An inclusive gateway allows either of the outgoing sequence flow to be taken or several in parallel.
- Example: The following process shows a process where the guests do not have both appetizers and entrees but can have only one of them.



(Bridgeland & Zahavi 2009, p. 114f)

Inclusive Gateways

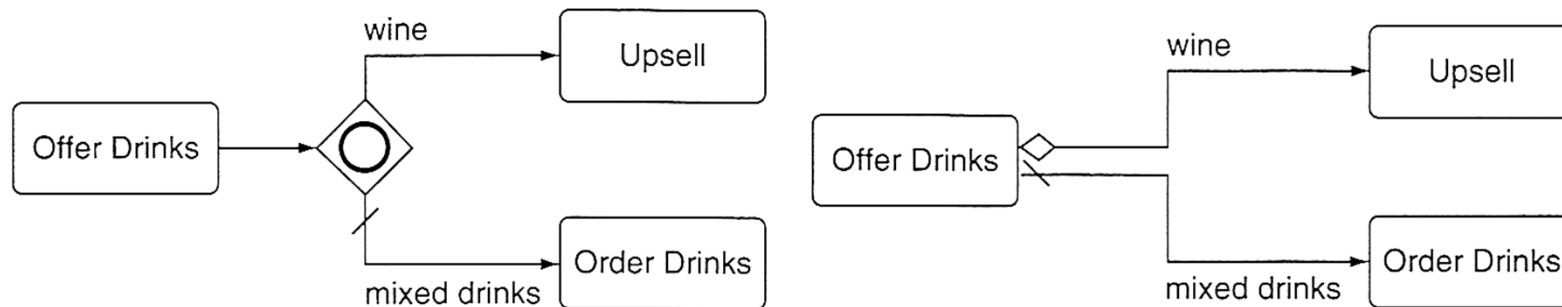


- Inclusive Gateways are Decisions where there is more than one possible outcome
- The „O“ marker is used to identify this Gateway
- They usually are followed by a corresponding merging Inclusive Gateway

Default Sequence Flow and Conditional Sequence Flow

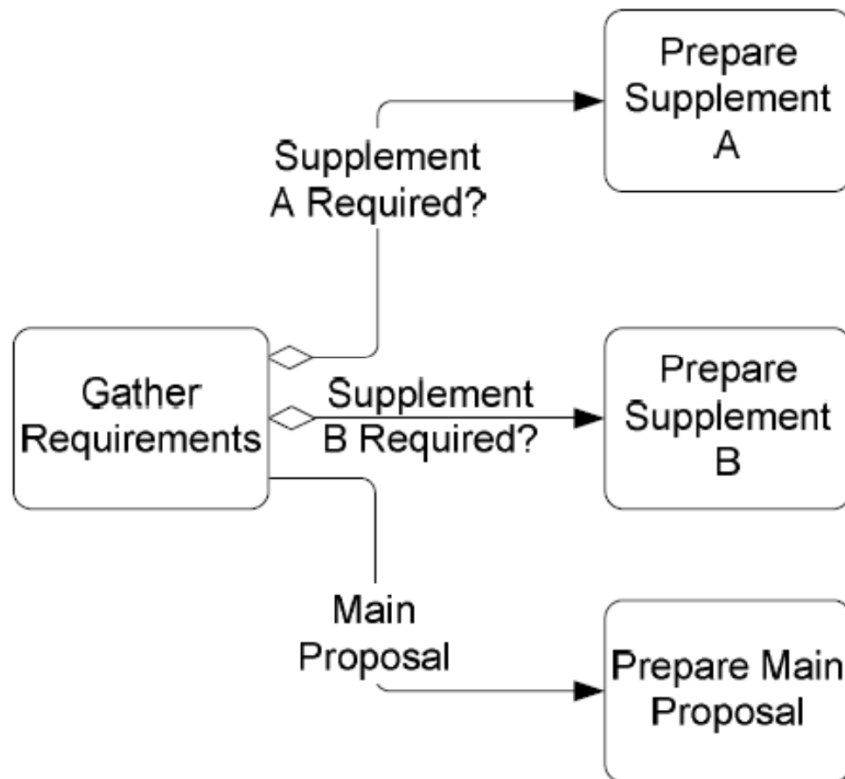
- One of the outgoing sequence flows from a gateway can be marked as default – the one that is taken if there is no reason to take another sequence flow.
- The default is modeled with a short line crossing the sequence flow.
- The same can be modeled without a gateway using a conditional sequence flow.
- A conditional sequence flow is a sequence flow that includes a condition

Example: Identical process with a gateway and with conditional sequence flow



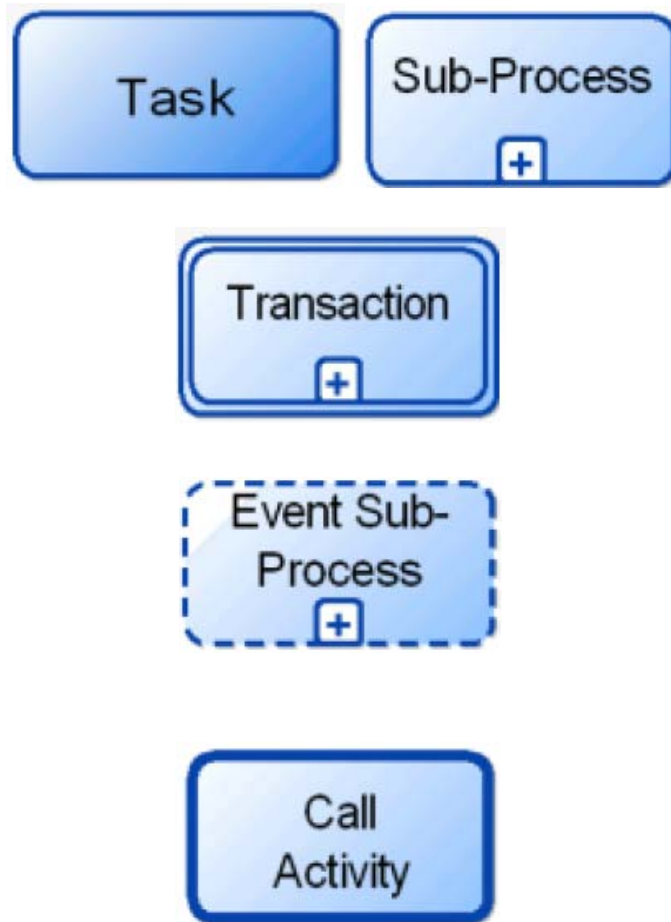
(Bridgeland & Zahavi 2009, p. 116)

Conditional Sequence Flow



- The condition of a sequence flow has to be true to allow the flow to continue down the Sequence Flow
 - ◆ A mind-diamond shows that the Sequence Flow has a condition
- At least one of the outgoing Sequence Flows must be chosen during Process performance
- Sequence flows without condition are followed in any case
 - ◆ In the example main proposal is prepared
- Defaults can be used to select a flow if no other condition is true

Activities

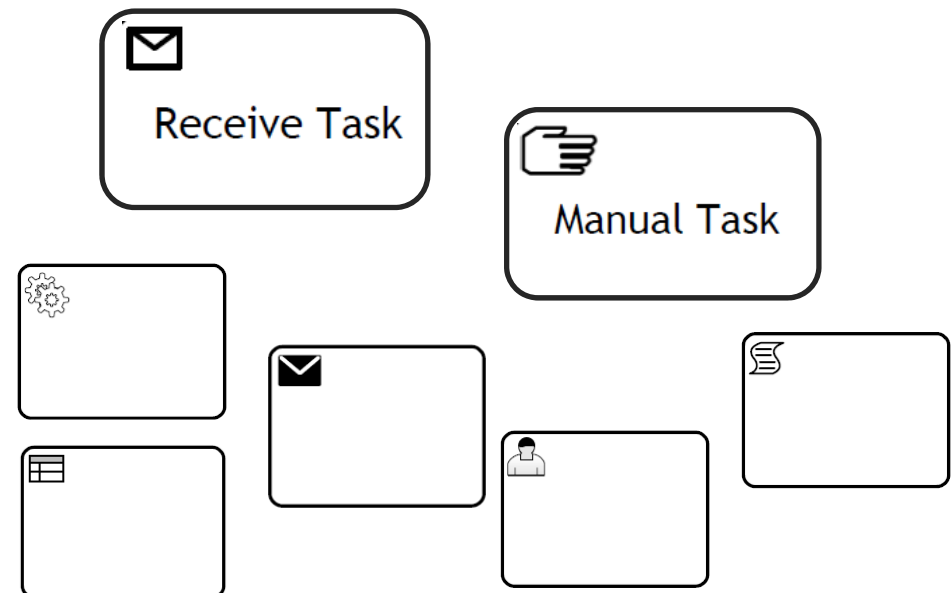


- A **Task** is a unit of work, the job to be performed. When marked with a [+] symbol it indicates a **Sub-Process**, an activity that can be refined.
- A **Transaction** is a set of activities that logically belong together; it might follow a specified transaction protocol. .
- An **Event Sub-Process** 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 Sub-Process or Task that is reused in the current process.

Task Types

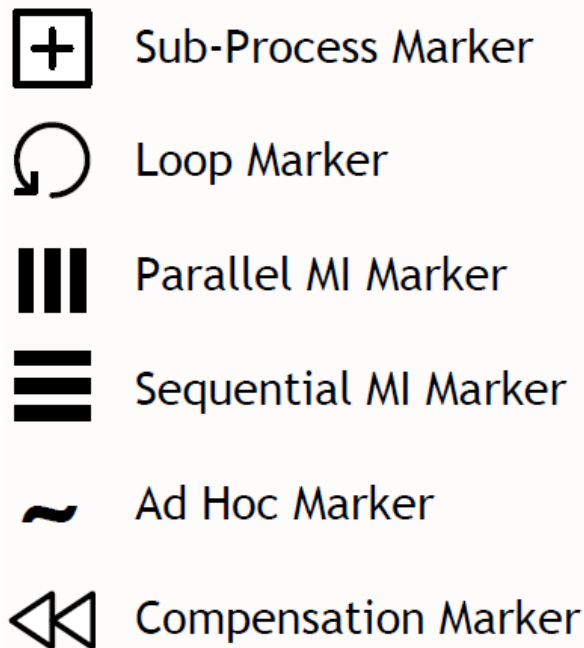
	Send Task
	Receive Task
	User Task
	Manual Task
	Business Rule Task
	Service Task
	Script Task

- Types specify the nature of the action to be performed .
- They can be identified by a symbol inside the object

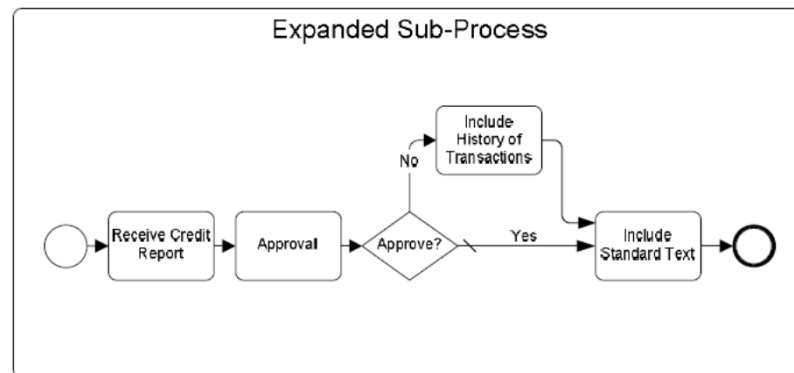
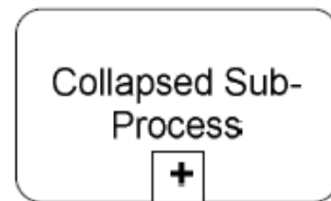


Activity Markers

- Markers indicate execution behavior of activities / subprocesses



Sub-Processes



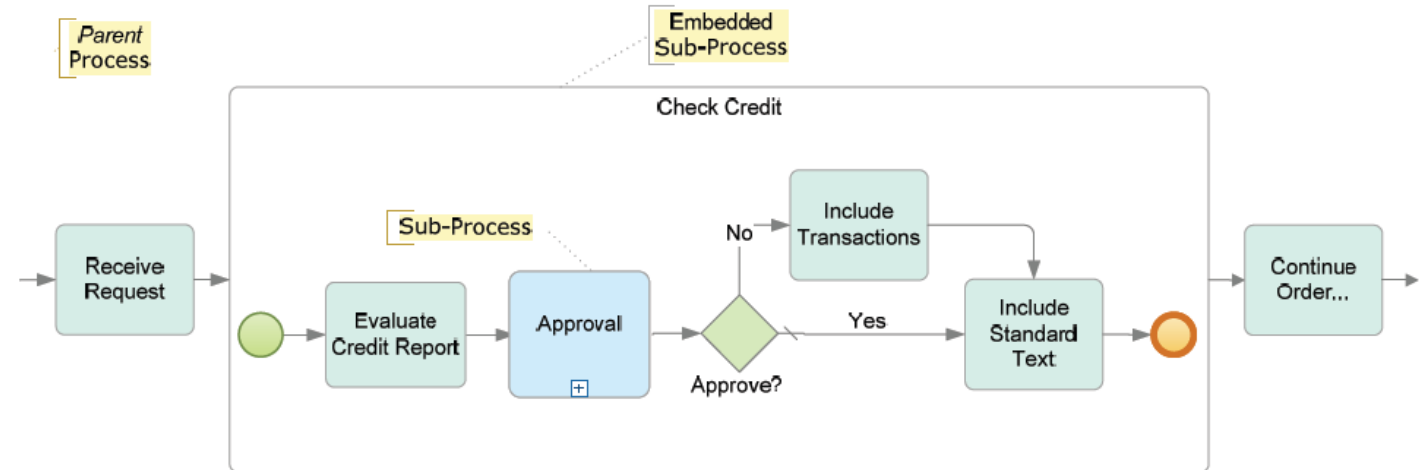
- A Sub-Process is a compound activity that is included within a Process.
 - ◆ A process can be broken down into a finer level of detail through a set of sub-activities
- Two kinds of representation
 - ◆ Collapsed: the details of the Sub-Process are not visible in the Diagram. A “plus” sign in the lower-center of the shape indicates that the activity is a Sub-Process and has a lower-level of detail.
 - ◆ Expanded: the details (a Process) are visible within its boundary.

Embedded and Independent Sub-Processes

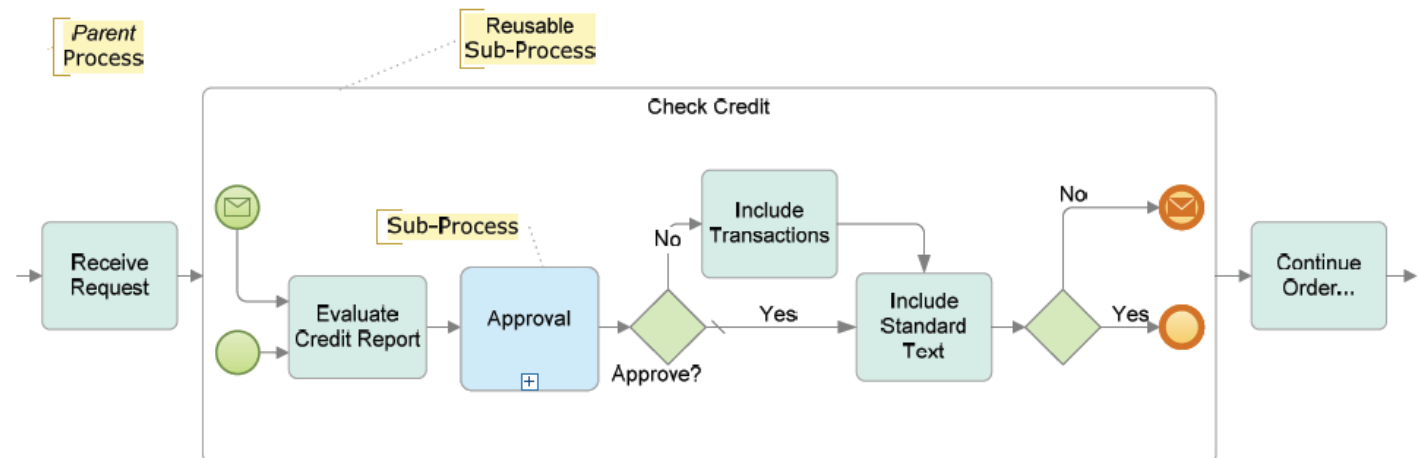
- Embedded - A modeled Process that is actually part of the *parent* Process.
 - ◆ *Embedded* Sub-Processes are not re-usable by other processes.
 - ◆ All "process relevant data" used in the *parent* Process is directly accessible by the *embedded* Sub- Process (since it is part of the *parent*).
- Independent¹⁾ - A separately modeled Process that could be used in multiple contexts.
 - ◆ Example: checking the credit of a customer
 - ◆ Any data must be transferred specifically between the *parent* and Sub-Process.
 - ◆ An independent Sub-Process can also be called Top-level process.

Examples of Embedded and Independent Sub-Processes

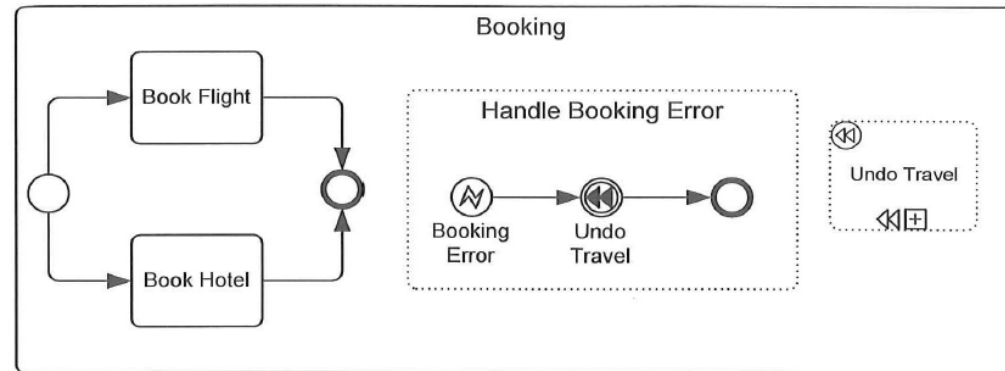
Embedded Sub-Process (expanded)



Independent Sub-Process (expanded)



Event Sub-Processes

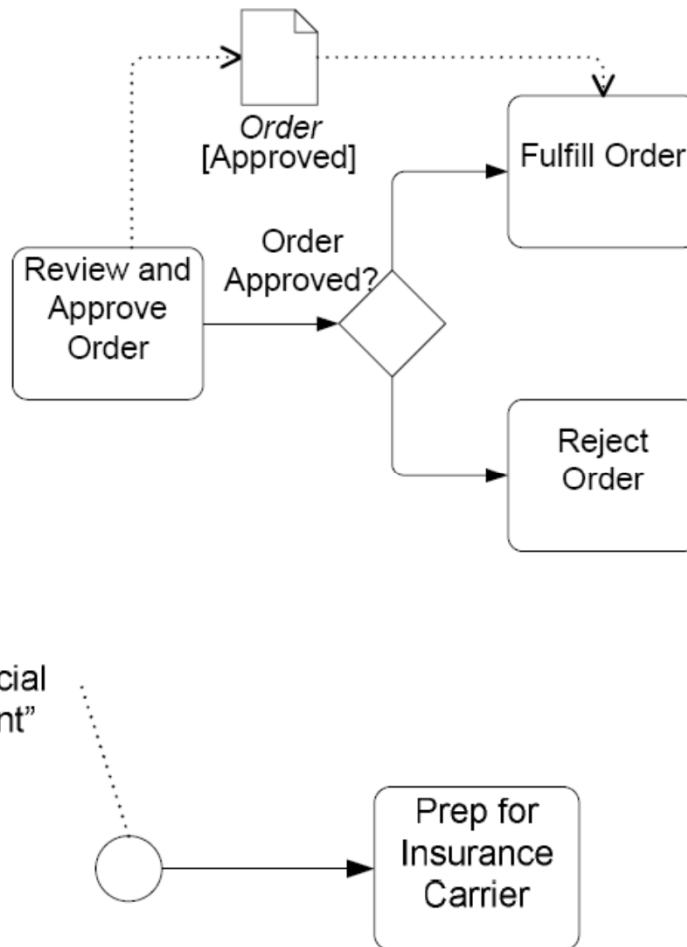


- An Event Sub-Process is contained within a Process, but it is not part of the main flow of the Process
 - ◆ An Event Sub-Process is a Sub-Process that is initiated only when its Start Event occurs.
 - ◆ As the Process flows from the normal Start Event to the End Event, the Event Sub-Process will not be initiated
- Event Sub-Processes are similar to boundary Events, except they are not associated to a single activity but the whole process

Artifacts

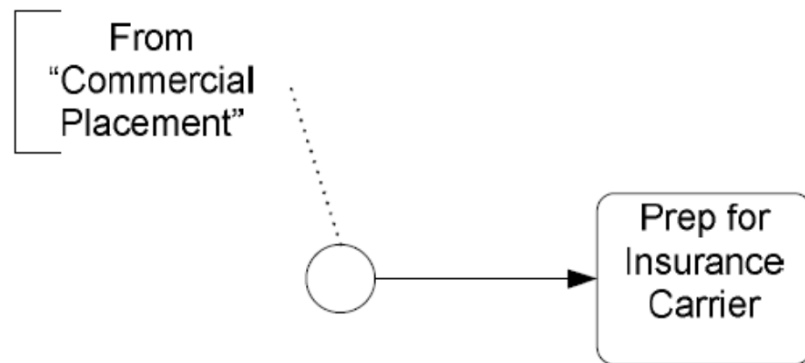
- Artifacts provide the capability to show information beyond the basic flow-chart structure of the Process
- There are currently three standard Artifacts in BPMN:
 - ◆ Data Objects
 - ◆ Groups
 - ◆ Annotations
- A modeler or tool can extend BPMN by defining new Artifacts

Associations

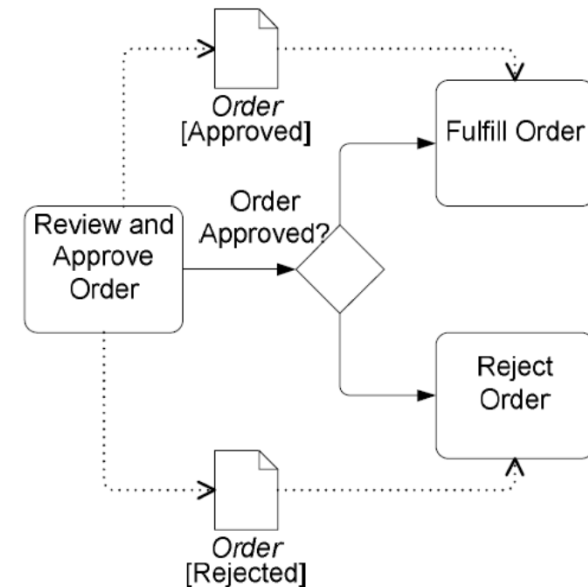


- An Association is used to associate data objects and artifacts with flow objects
- Associations are used to show how data is input to and output from Activities
- Text Annotations can be associated with objects

Text Annotations and Data Objects



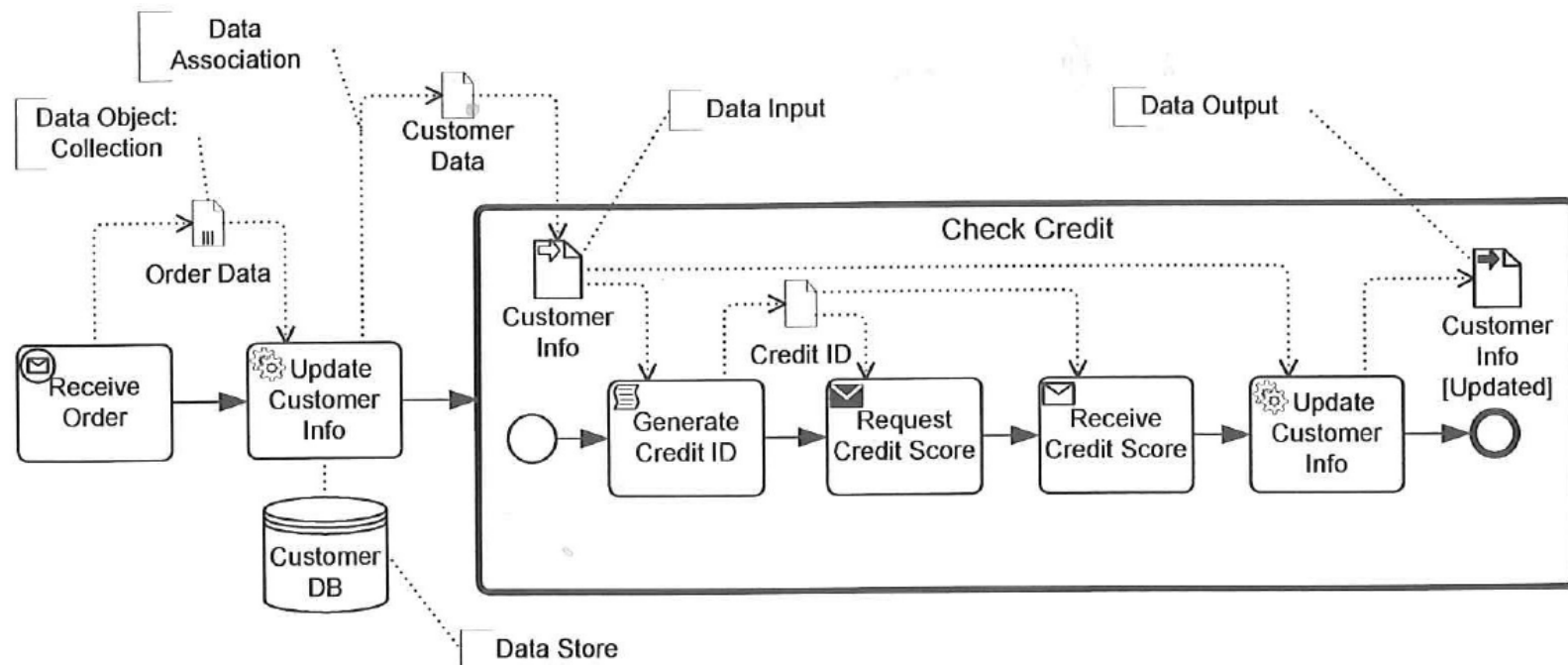
- Text Annotations are a mechanism for a modeler to provide additional information about a Process
- Text Annotations can be connected to a specific object on the Diagram with an Association



- Data Objects can be used to define inputs and outputs of activities
- Data Objects can be given a “state” that shows how a document may be changed or updated within the Process

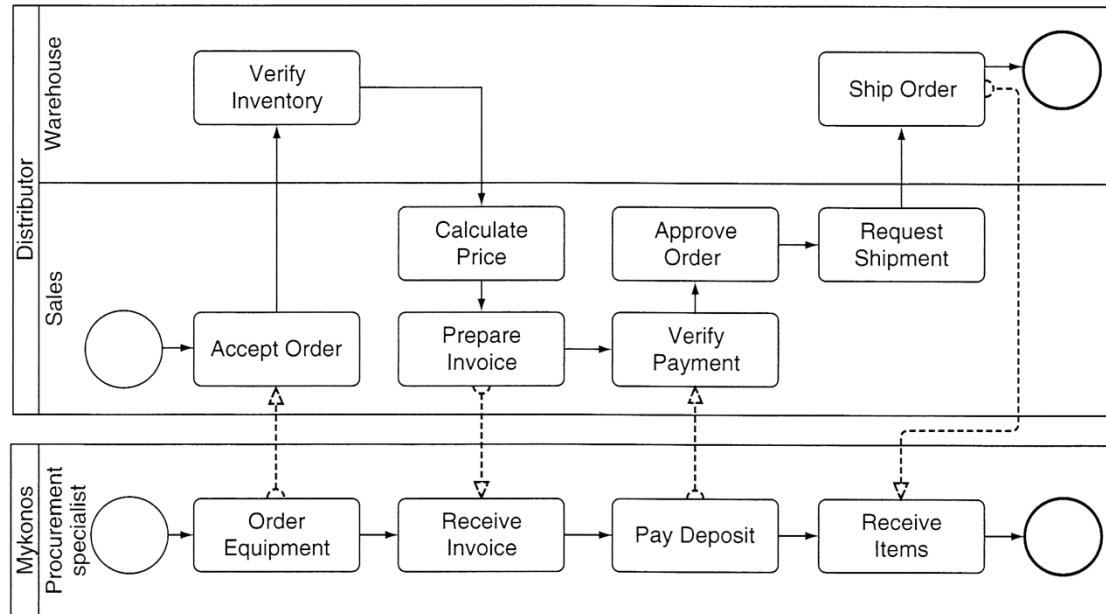
Data Elements in BPMN

- BPMN 2.0 contains new graphical elements to represent data
 - ◆ Data Associations: connecting Data Objects to Activities
 - ◆ Data Inputs and Outputs can be visualized
 - ◆ Data Stores represent repositories or databases
 - ◆ Collections, marked by [+], represent groups of Data Objects



Swimlanes

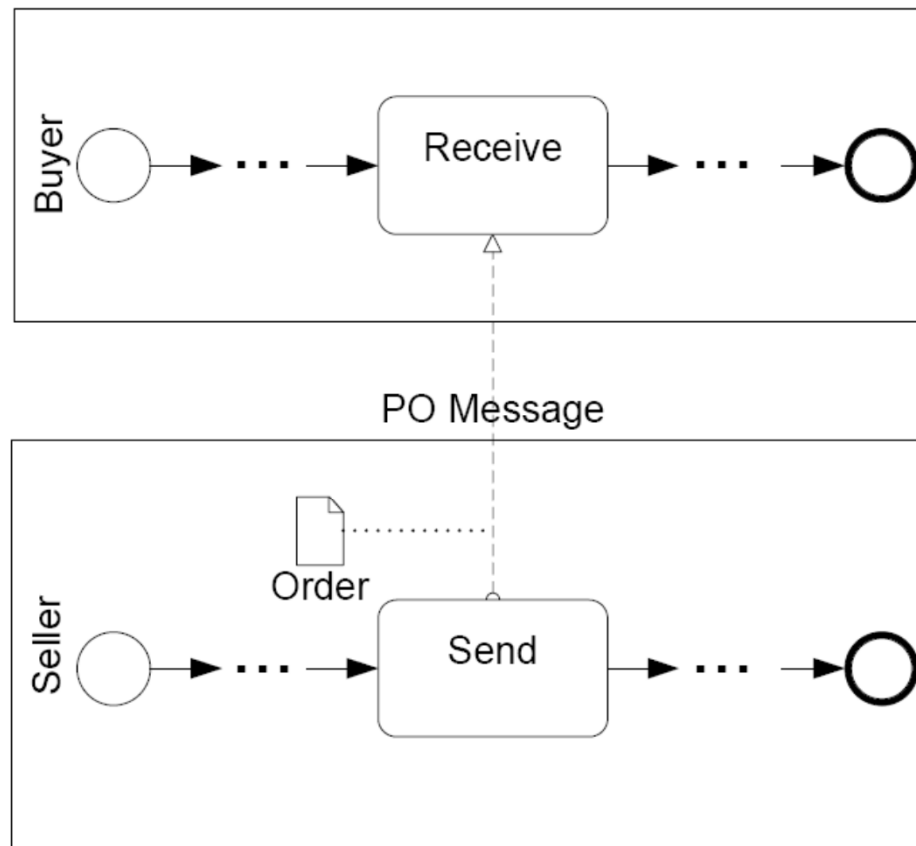
- Swimlanes partition and organise activities
- There are two main types of swimlanes: Pool and Lane
 - ◆ Pools represent Participants in an interactive (B2B) Business Process Diagram
 - ◆ Lanes represent sub-partitions for the objects within a Pool – they represent participants of a process



(Bridgeland & Zahavi 2009, p. 123)

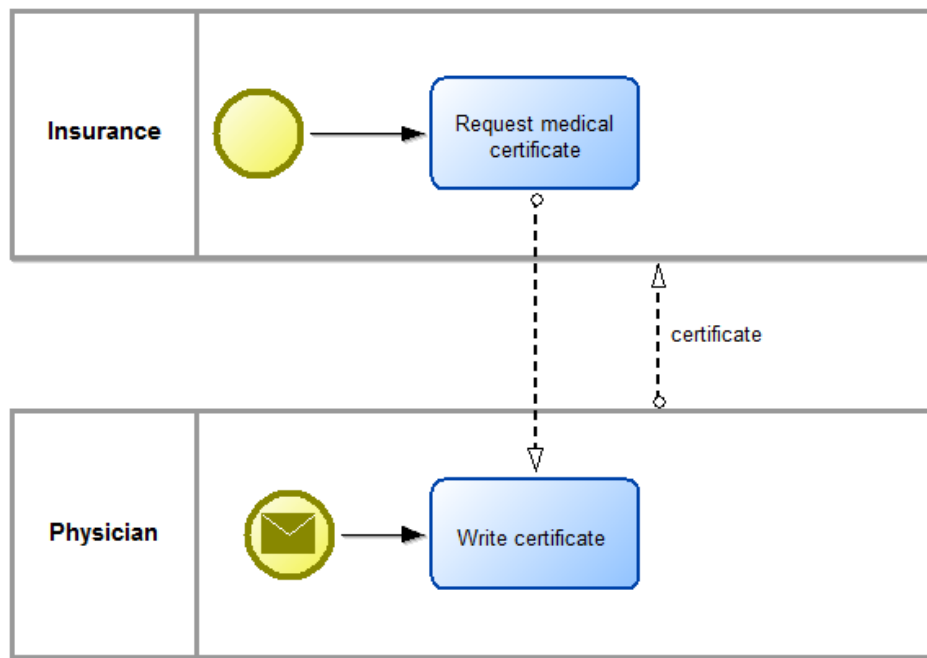


Pools



- Pools represent independent Participants in an interactive (B2B) Business Process
 - ◆ A Participant may be a business role (e.g. „buyer“ or „seller“) or a business entity (e.g. „IBM“ or „OMG“)
- A Pool may be a „black box“ or may contain a Process
- Interaction between Pools is handled through **Message Flow**
- Sequence Flow must not cross the boundary of a Pool (i.e. a Process is fully contained within a Pool)

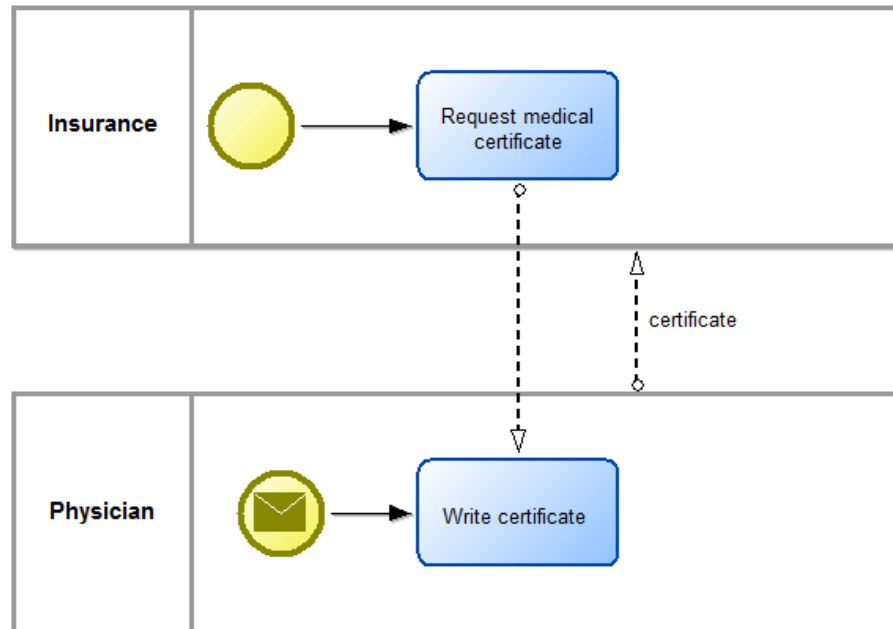
Message Flow



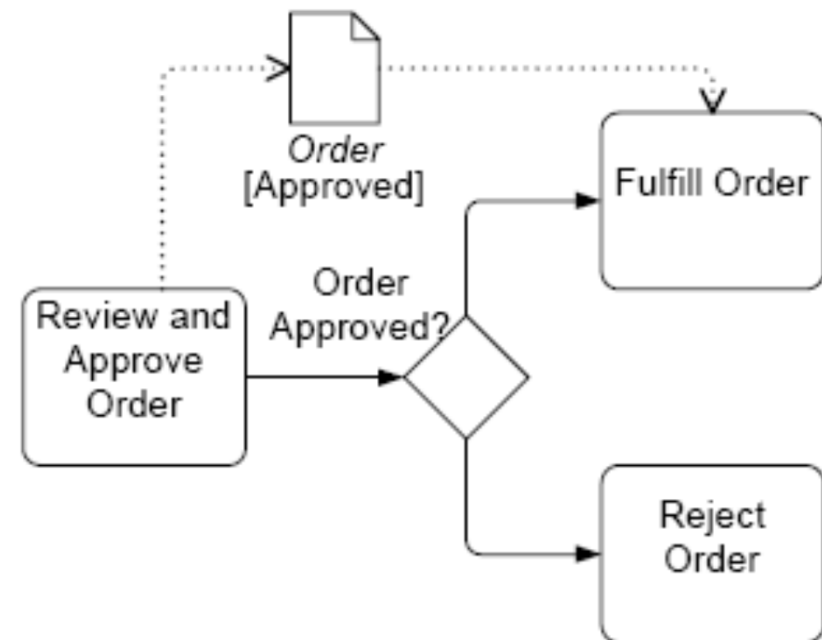
- A Message Flow is used to show the flow of messages between two Pools (i.e. between two processes, because pool contains a process)
- A Message Flow can connect to the boundary of the Pool or to an object within the Pool
- Message Flows are not allowed between objects within a single Pool

Data Transfer with Message Flow und Associations

Message Flow between pools:

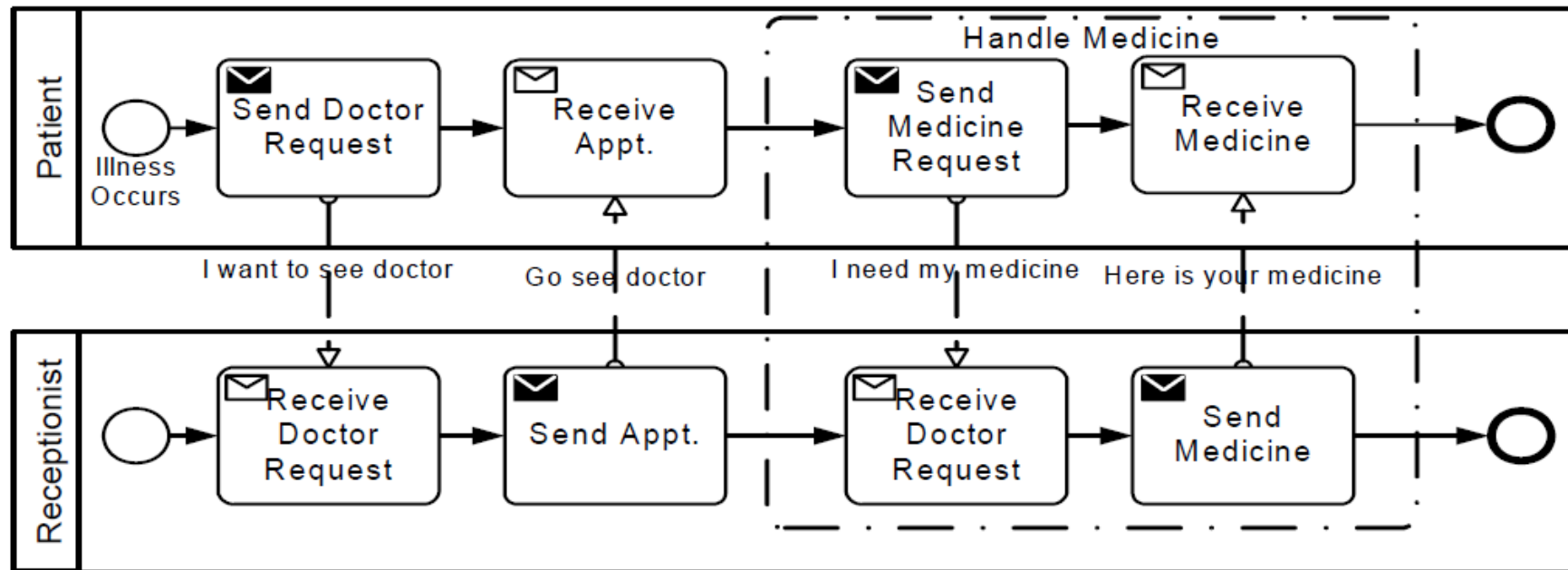


Data transfer inside a pool MUST NOT be modeled with Message Flow but with Associations :



Groups

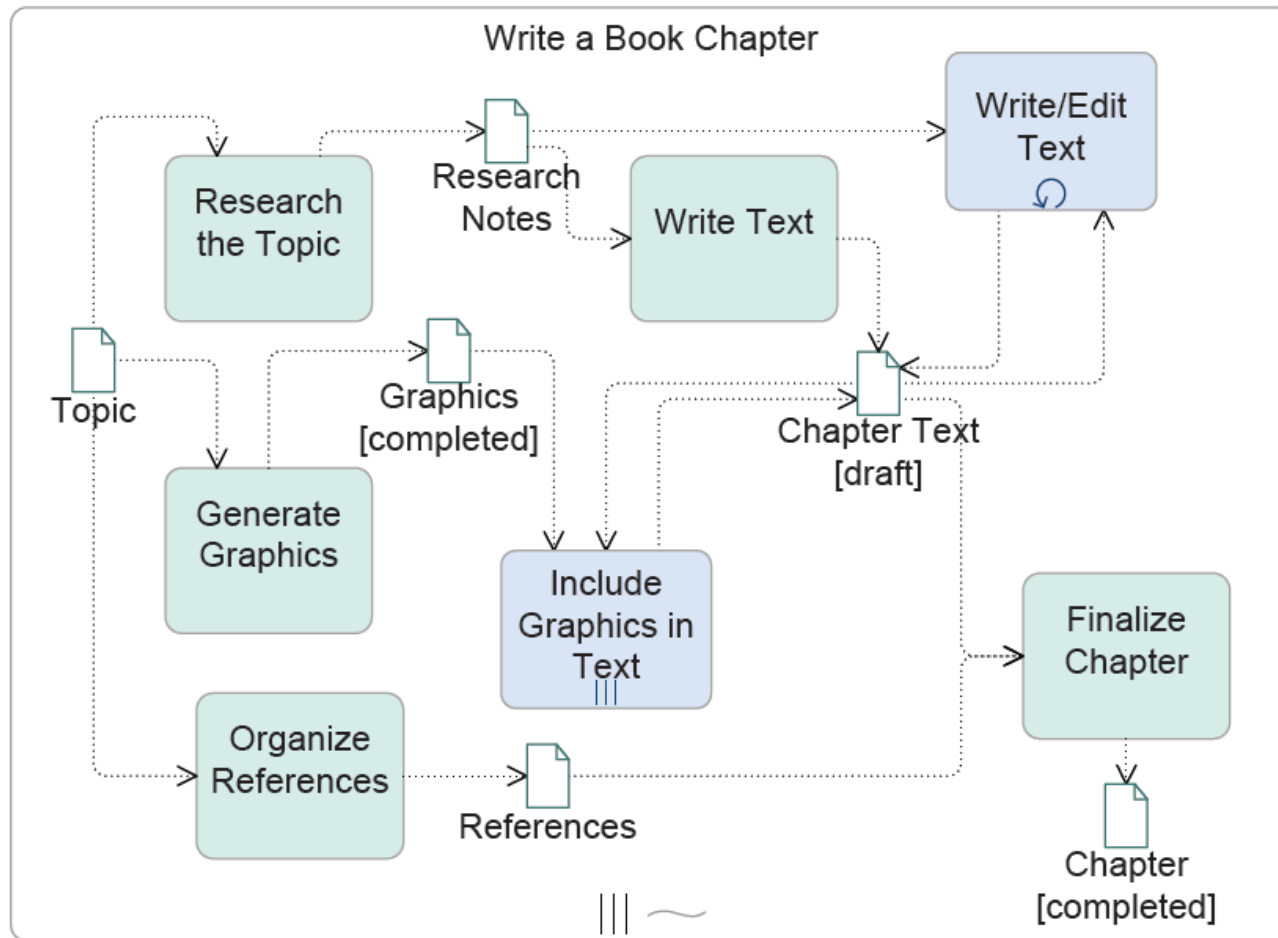
- The Group object is an Artifact that provides a visual mechanism to group elements of a diagram informally
- A Group can stretch across the boundaries of a Pool, often to identify Activities that exist within a distributed business-to-business transaction.



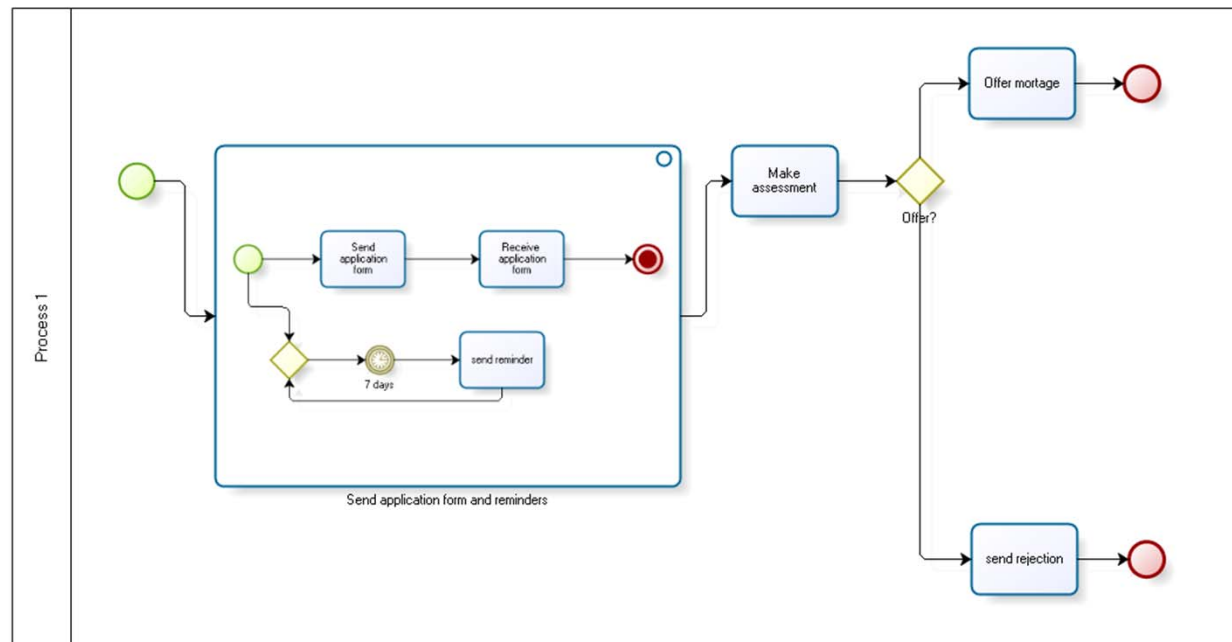
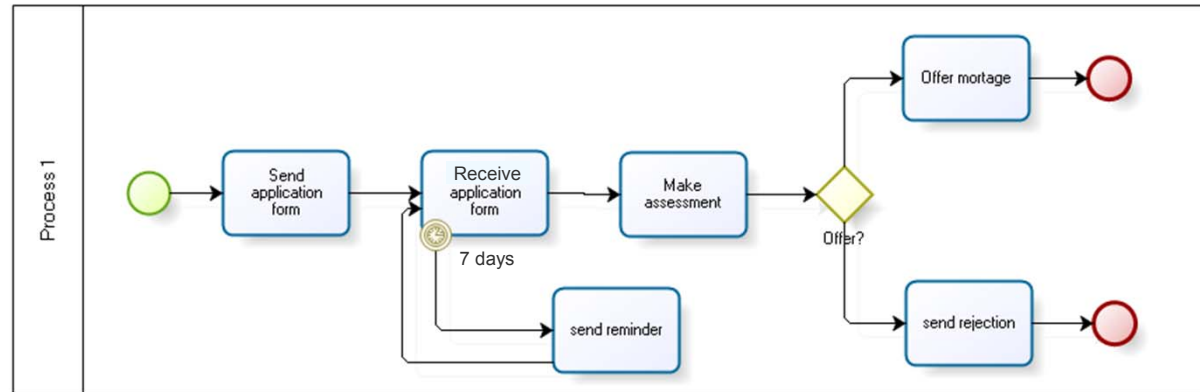
Ad hoc Processes

- The Ad hoc process represents processes where the activities can occur
 - ◆ in any order
 - ◆ In any frequency
- There is no specific ordering or obvious decisions
- It has a tilde (~) to show that it is ad hoc
- Typically, the activities in an ad hoc process involve human performers to make decisions as to what activities to perform, at which time and how many time
- It is possible, however, to use occasional sequence flow between some activities, but sequence flow does not imply that there are explicit start and end events.
- The ad hoc process has a non-graphical completion condition attributes. When the attribute becomes true (by updating the date expressed in the condition), the process terminates.

Example of an Ad hoc Process



Exercise – Timer



The following models represent a loop with a timer event.

- ◆ Are the two representations equivalent?
- ◆ Why do we need a subprocess in the second solution?
- ◆ Why do we need a termination end event in the subprocess?

Exercise

- What are the possible uses of process models?
- From this derive: What would you want to represent in a process model?
- → Brainstorming

Exercise

The restaurant manager gave you the following description of the reservation at a Mykonos restaurant:

- When the guest sends a request for a reservation, the reservation is written into the reservation book. **If all places are occupied, the guest is put on the waiting list. In this case the guest is informed at 7:30 about acceptance or cancellation.**
- At 7:00 pm the guests are assigned to the tables and then the reservation cards are placed on the tables. **This has to be completed by 7:20. If it is not, an alert is sent to the manager.** At 8:00 pm the restaurant is opened.