Business Process Management

Paolo Bottoni

DIPARTIMENTO
DI INFORMATICA

SAPIENZA
UNIVERSITÀ DI ROMA

Lecture 6: Collaborations and Coreographies

Resources

Active resources:

- Process participant
- Software system
- Equipment

Resource class:

A **group of (active) resources** that are interchangeable, e.g. a role, an organizational unit or the whole organization.

Resources in the order-to-cash example

The order-to-cash process is carried out by a seller's organization which includes two departments: **Sales** and **Warehouse & Distribution**.

The purchase order received by the **Sales department** has to be checked against the stock. This is done via an **ERP** module within the **Warehouse & Distribution department**. If the purchase order is confirmed, the **Warehouse & Distribution department** ships the goods. Meantime, the **Sales department** emits the invoice. The process concludes with the order being archived by the **Sales department**.
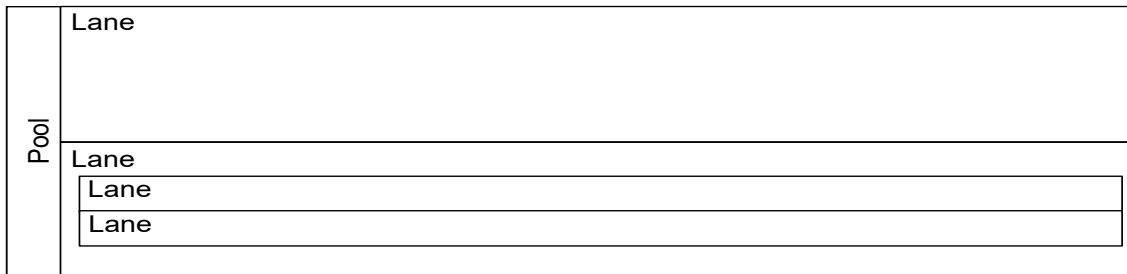
BPMN Elements – Pools & Lanes

Pool

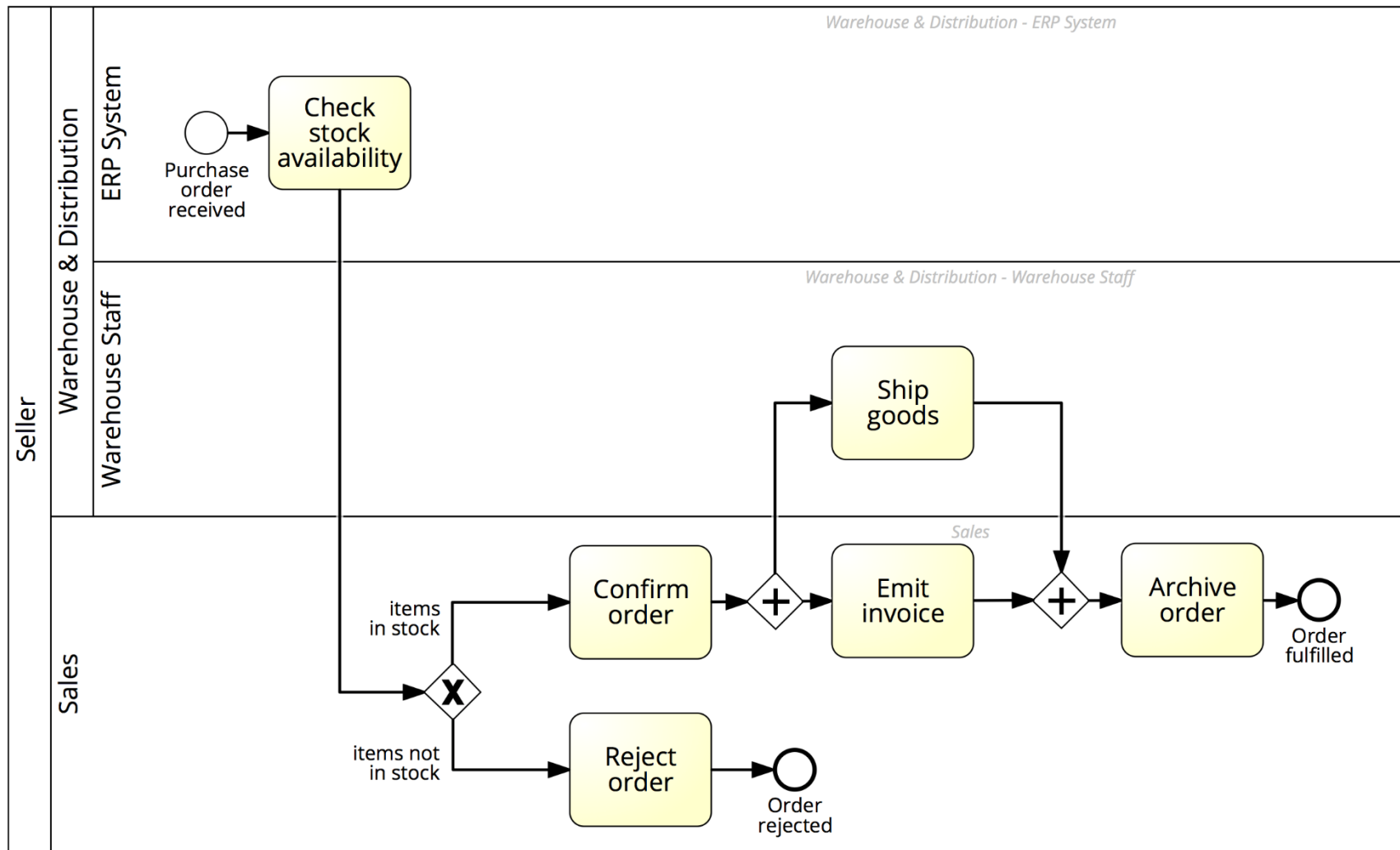Captures a resource class. Generally used to model a business party (e.g. a whole company)

Lane

Captures a *resource sub-class* by partitioning a pool. Generally used to model departments (e.g. shipping, finance), internal roles (e.g. Manager, Associate), software systems (e.g. DBMS, CRM) or equipment (e.g. Manufacturing plant)

Solution

Order-to-cash

# Exchanging information between business parties

Order-to-cash

The purchase order **sent by the Customer** is received by the Sales department and checked against the stock. This is done via an ERP module within the Warehouse & Distribution department. If the purchase order is not confirmed, the Sales department **sends an order rejection to the Customer**, otherwise it **sends an order confirmation**.

Next, the Warehouse & Distribution department ships the goods and **sends a shipment notification to the Customer**. Meantime, the Sales department emits the invoice and **sends it to the Customer**. The process concludes with the order being archived by the Sales department.
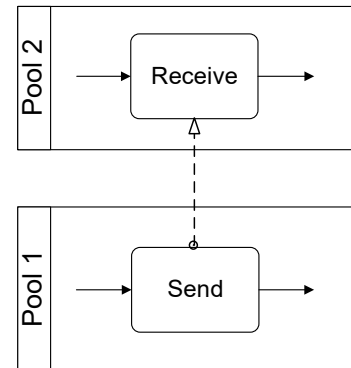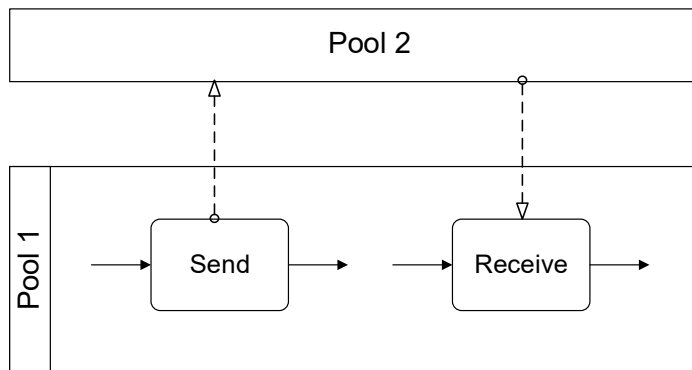
BPMN Elements – Message Flow

A *Message Flow* represents a flow of information between two process parties (Pools)
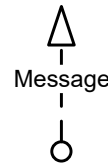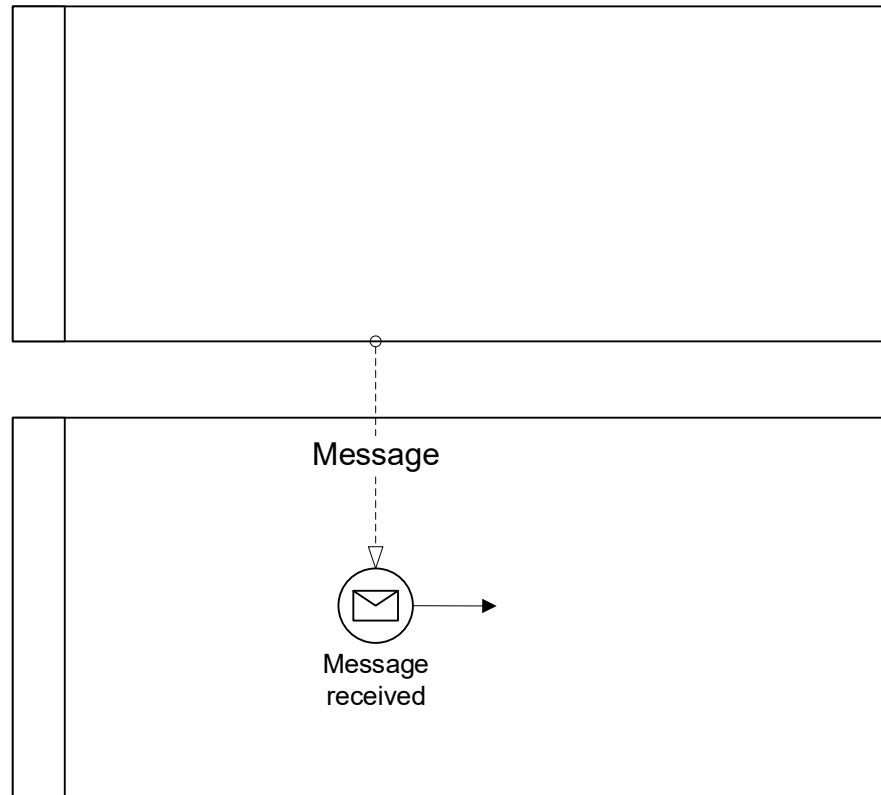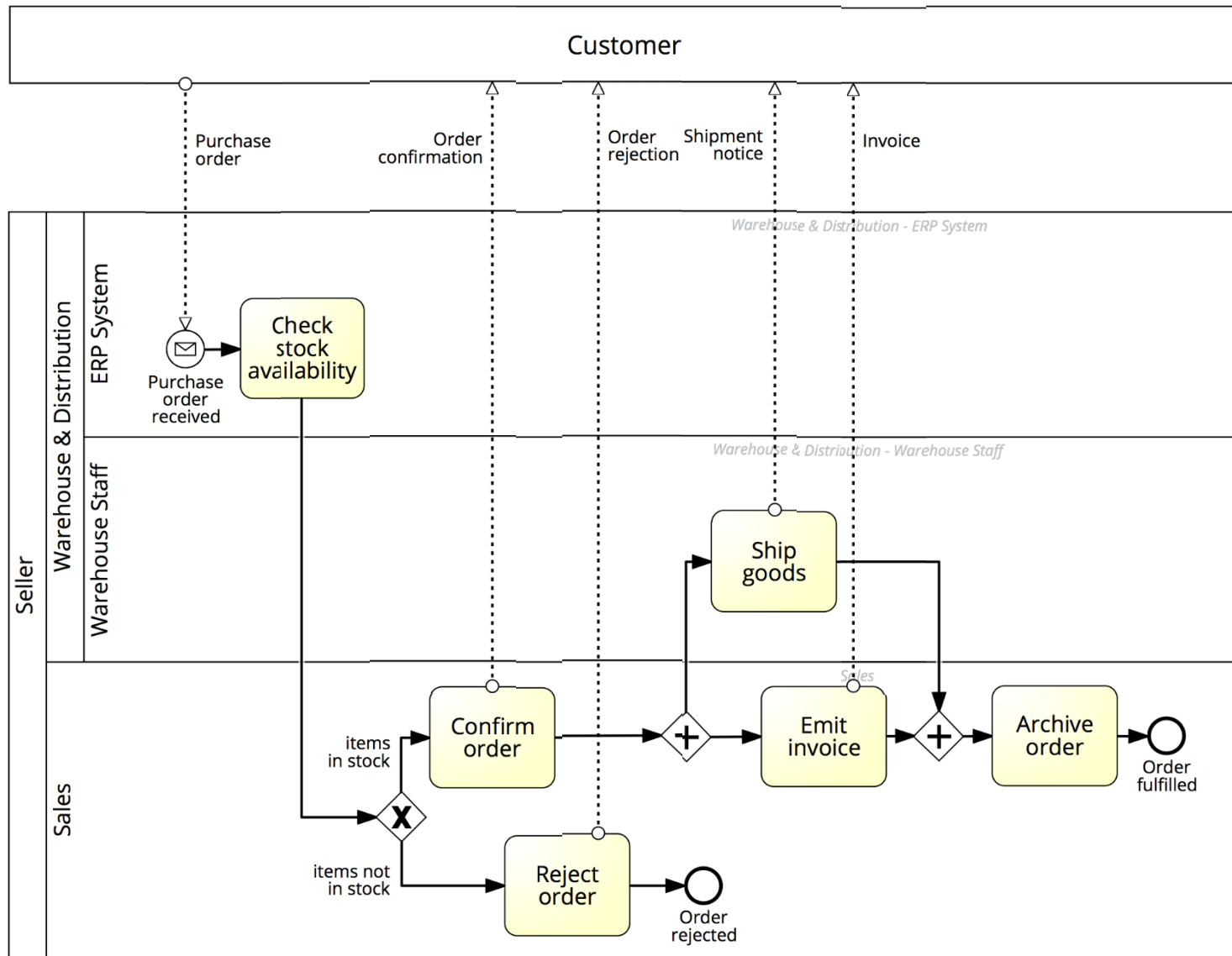
Message

A Message Flow can connect:

- directly to the boundary of a Pool ➔ captures an *informative* message to/from that party
- to a specific activity or event within that Pool ➔ captures a message that triggers a *specific* activity/event within that party

BPMN Elements – Start Message Event

The *start message event* triggers a process upon message receipt when an incoming message flow is connected to the event

Message

Message
received

Solution   Order-to-cash
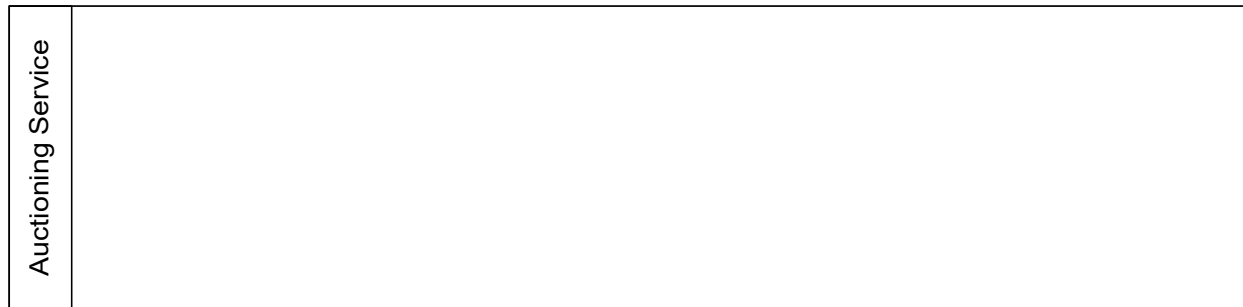
Pools, Lanes and Message Flows: syntax

1. The Sequence Flow **cannot** cross the boundaries of a Pool
2. Both Sequence Flow and Message Flow **can cross** the boundaries of Lanes
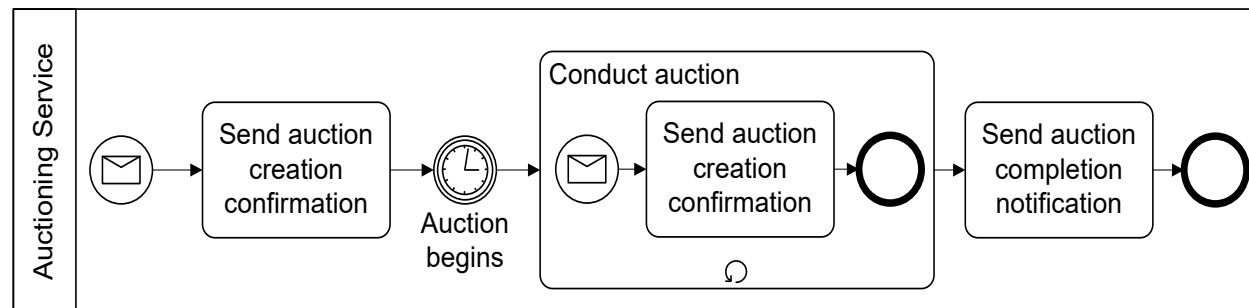3. A Message Flow **cannot connect** two flow elements within the same pool
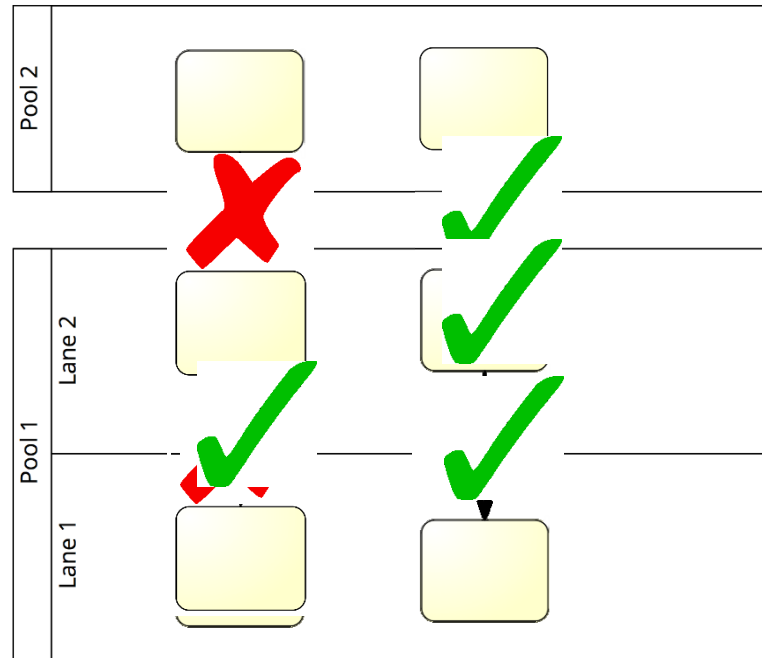
Process (or Orchestration) Diagram

Models a single business party and can be:
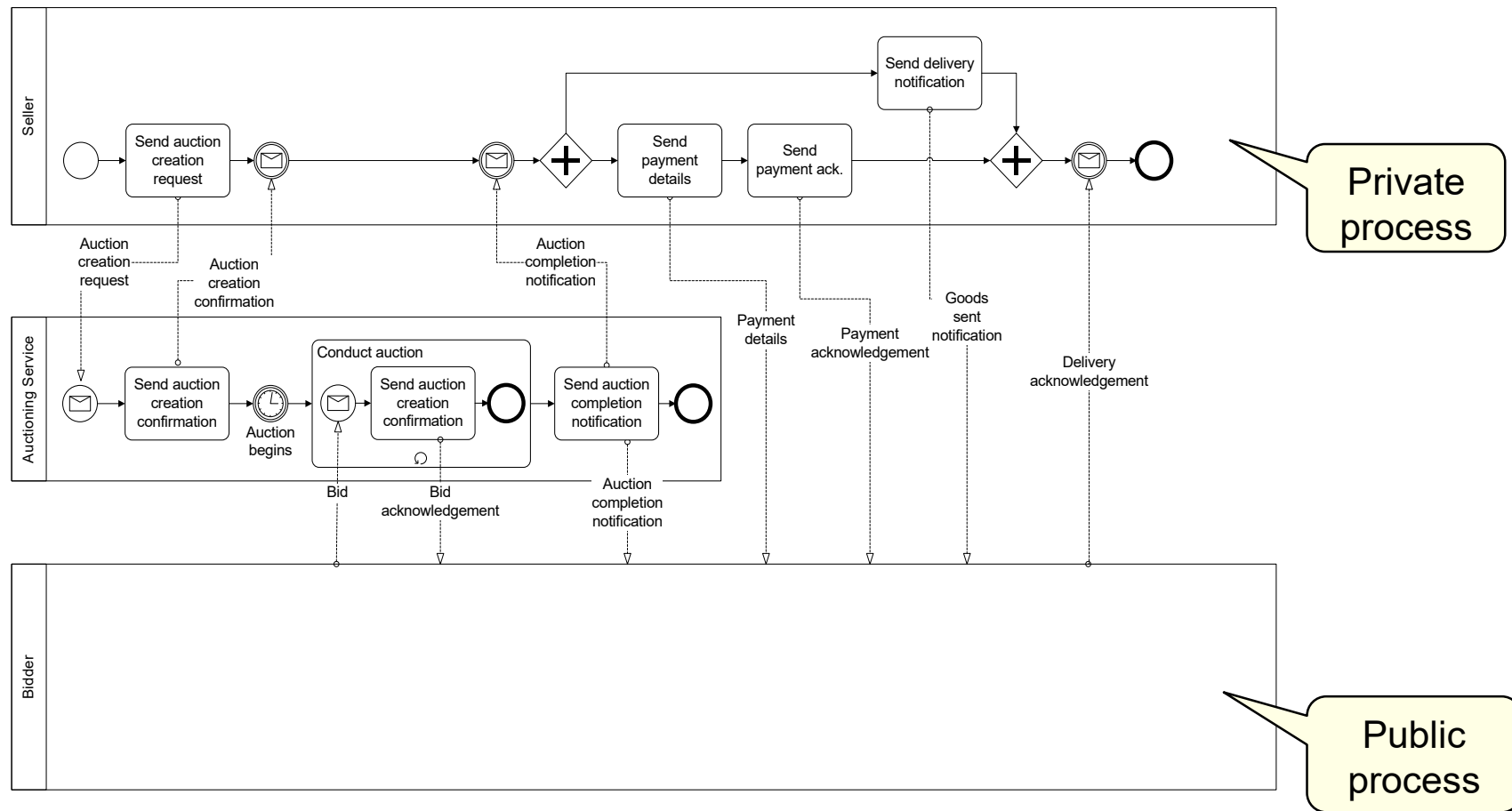
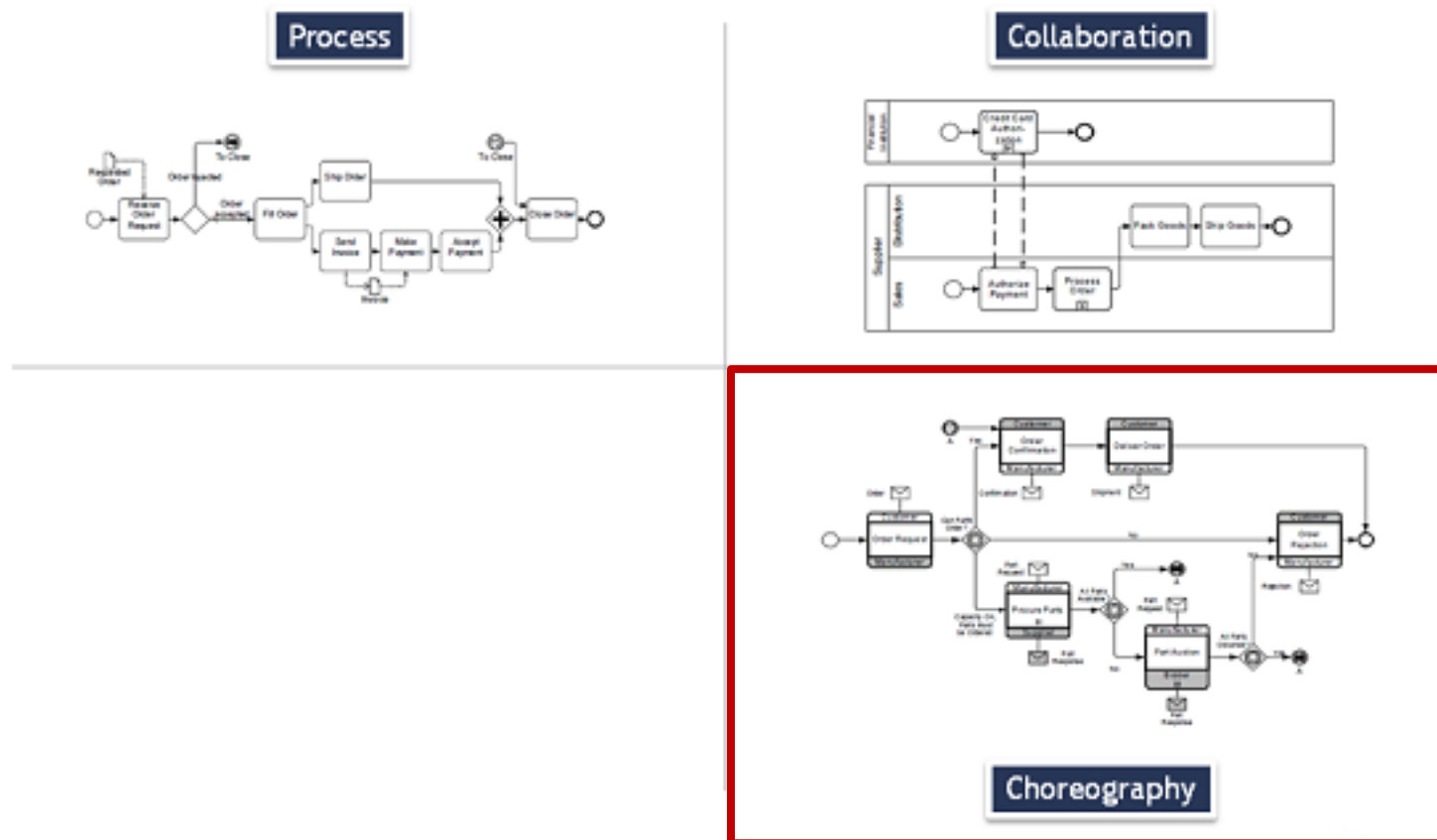**Public view** (*black box*)

Private view (*white box*)

Collaboration Diagram

Models a global business process between at least two business parties (each modelled by a Pool)
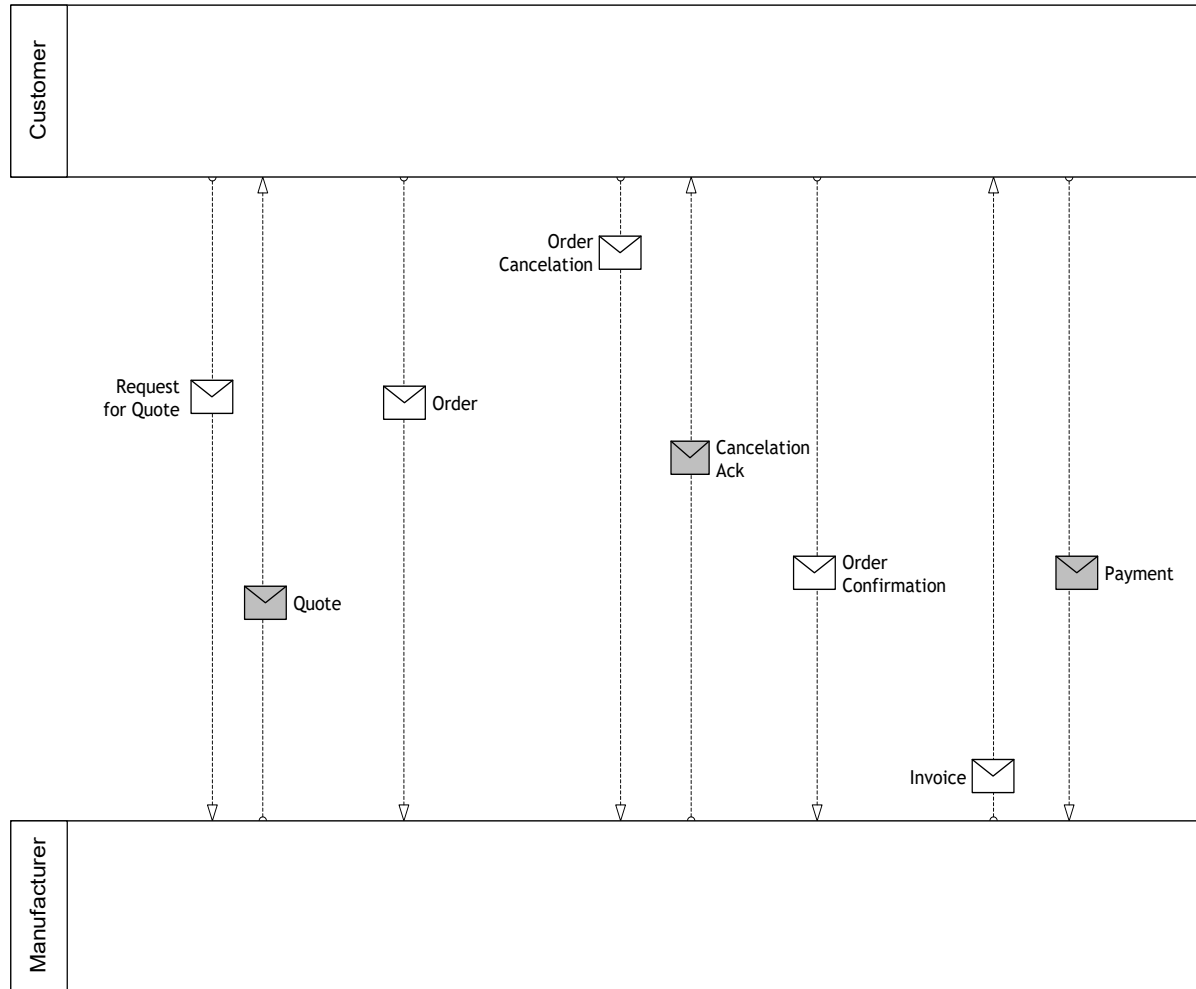
BPMN model types

Process
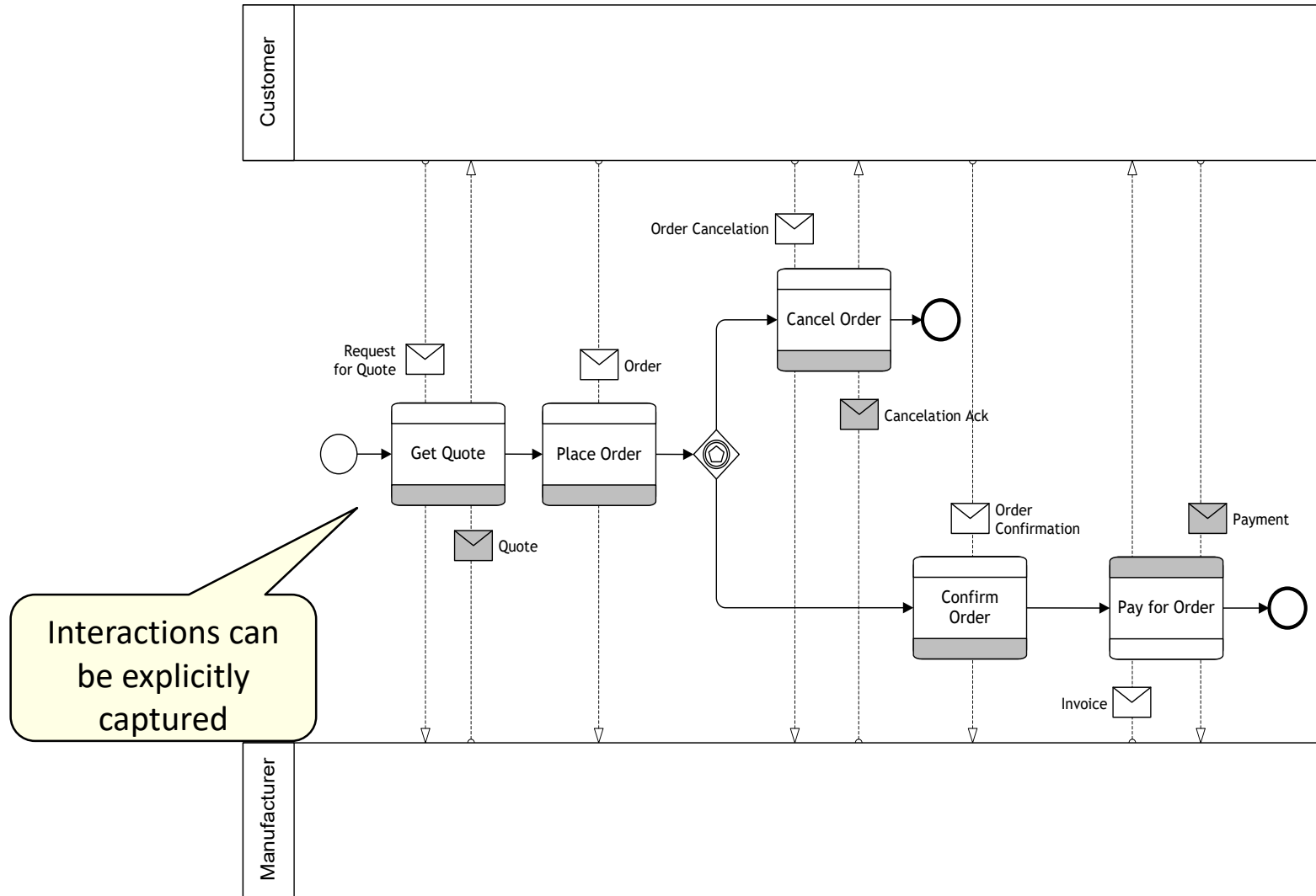
Collaboration

Choreography

Choreography Diagram

Focuses on the **interactions** among two or more business parties.

Choreography Diagram (cont'ed)

Choreography Diagram (cont'ed)

Pools can be dropped

Order Cancelation

Customer
Cancel Order
Manufactuer

Cancelation Ack

Request for Quote

Order

Customer
Get Quote
Manufacturer

Customer
Place Order
Manufacturer

Quote

Used in place of data-based XOR-split if data used to take the decision is **not "shared"** by the parties via a preceding message

Order Confirmation

Payment

Customer
Confirm Order
Manufactuer

Customer
Pay for Order
Manufacturer

Invoice

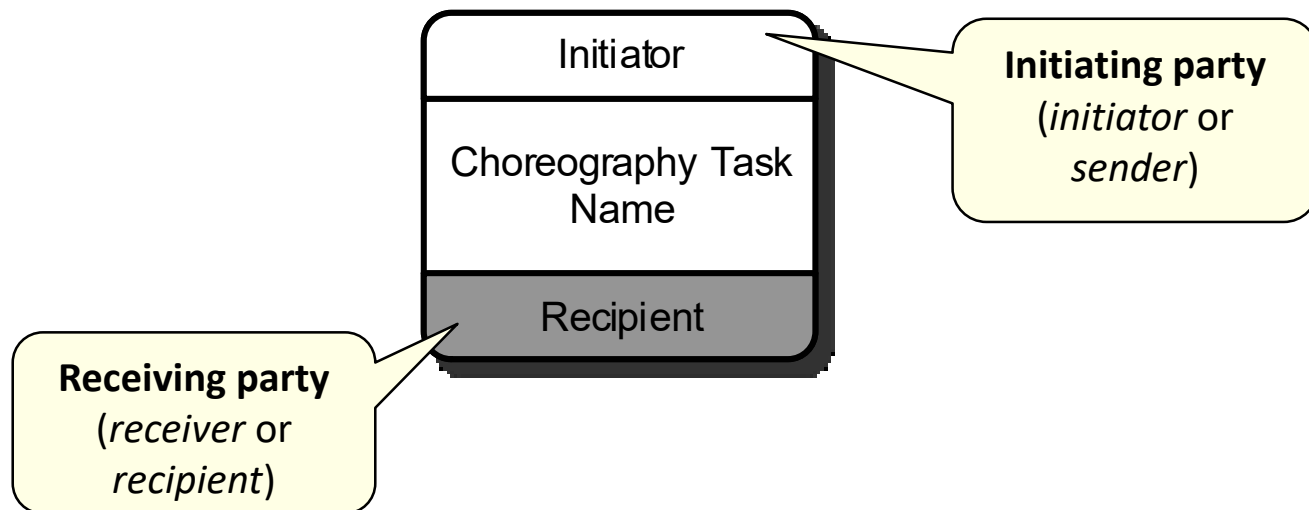# Choreography

Process model of the interactions taking place between two or more business parties

- Focuses on message exchange between parties
- Acts as a contract between parties
- Can be refined into private processes or into a collaboration diagram
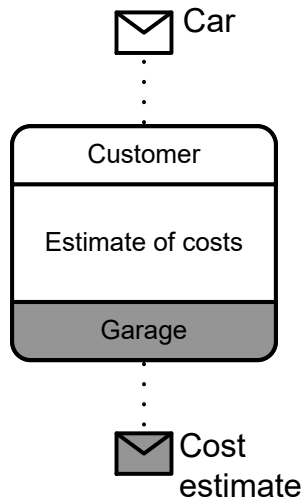
Choreography Task

- Represents an <u>interaction</u> between two business parties
- Either <u>one-way</u> (asynchronous) or <u>two-way</u> (synchronous)
- <u>Atomic</u>: max to messages being exchanged: request + response
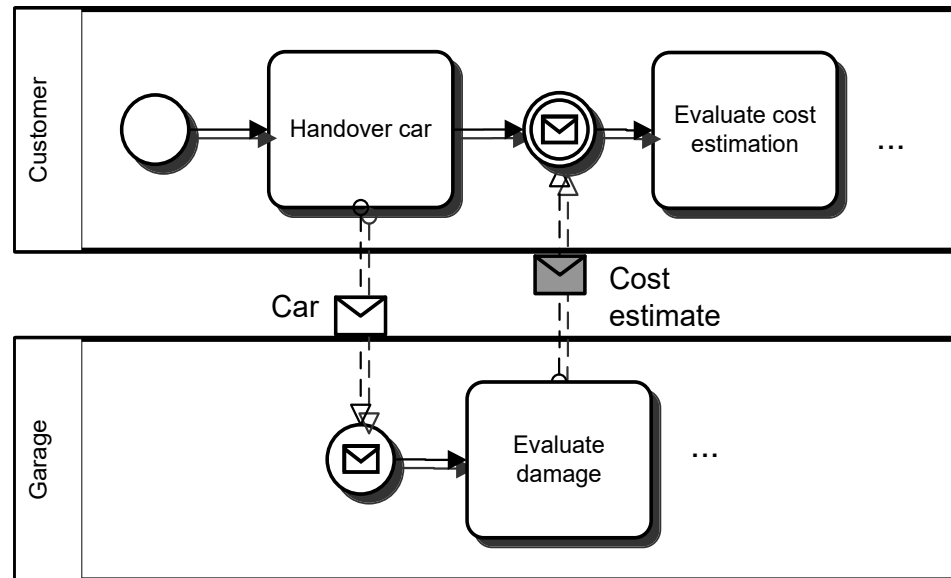- Distinction between *initiating* and *receiving* party

Initiator

Choreography Task Name

Recipient

Initiating party
(*initiator* or *sender*)

Receiving party
(*receiver* or *recipient*)

Choreography Task

- Band of initiating party unfilled
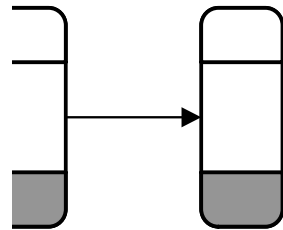- Message icons optional, follow band colors

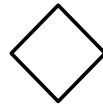Choreography Task

Collaboration View
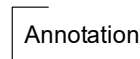
Basic Choreography Elements

Sequence flow – Connects and orders choreography tasks, events and gateways

Events – Most process events allowed. No non-interrupting events

Gateways – All process gateways allowed: Exclusive, Inclusive, Parallel and Event-based gateways
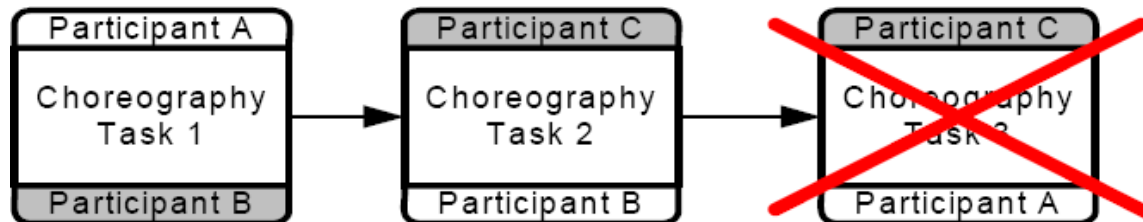
Annotation

Text annotations – No restrictions on their use

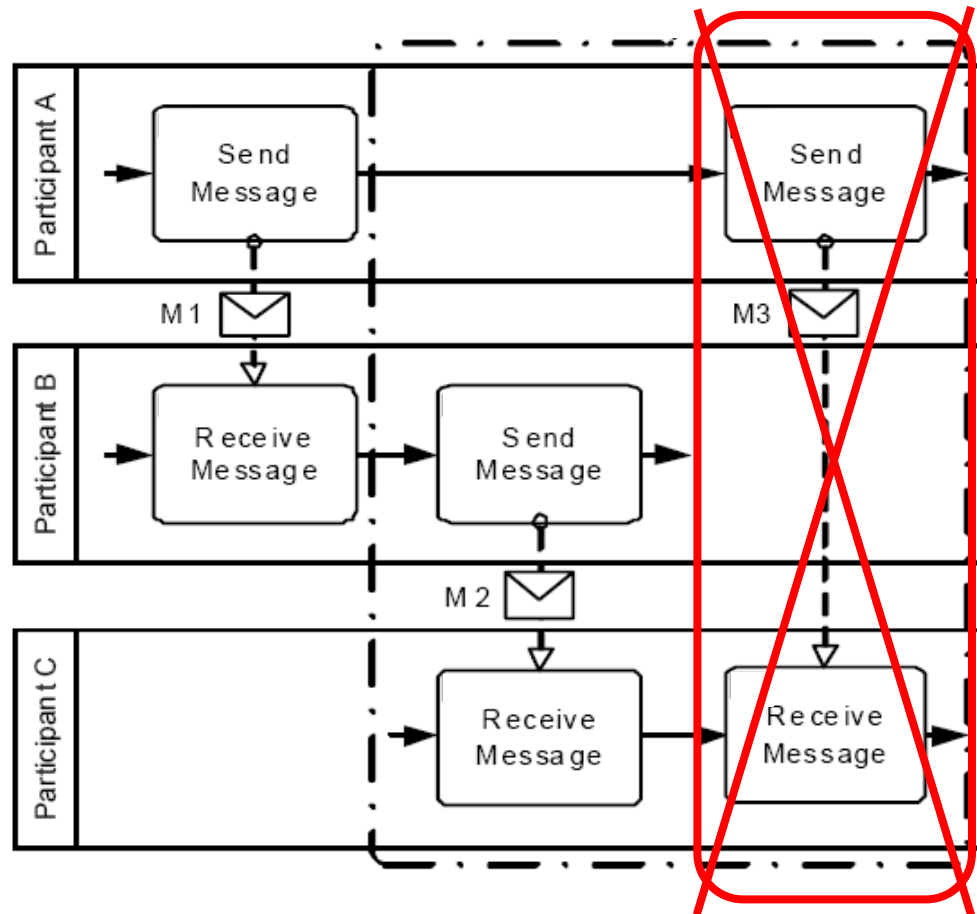# Syntax: choreography sequencing constraints

The initiator of a choreography activity must have been involved in the previous activity (excluding first activity)

Why?

Syntax: choreography sequencing constraints

Example: Choreography of an auction

A Seller sends information about an item they want to sell to an auction Provider. The Provider publishes the auction by offering the item to the Bidder. Once the auction has started the Bidder may place an offer. In case the item is sold the auction Provider finalises the purchase with the Buyer, otherwise the Provider notifies the Seller that his item has not been sold.
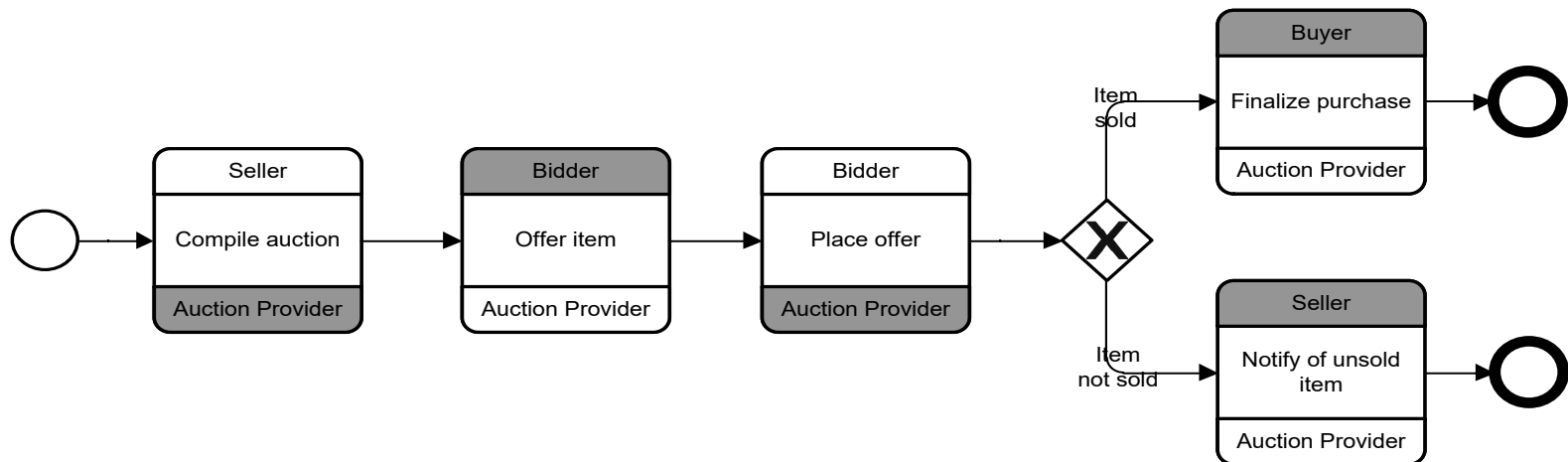
Solution: Choreography of an auction

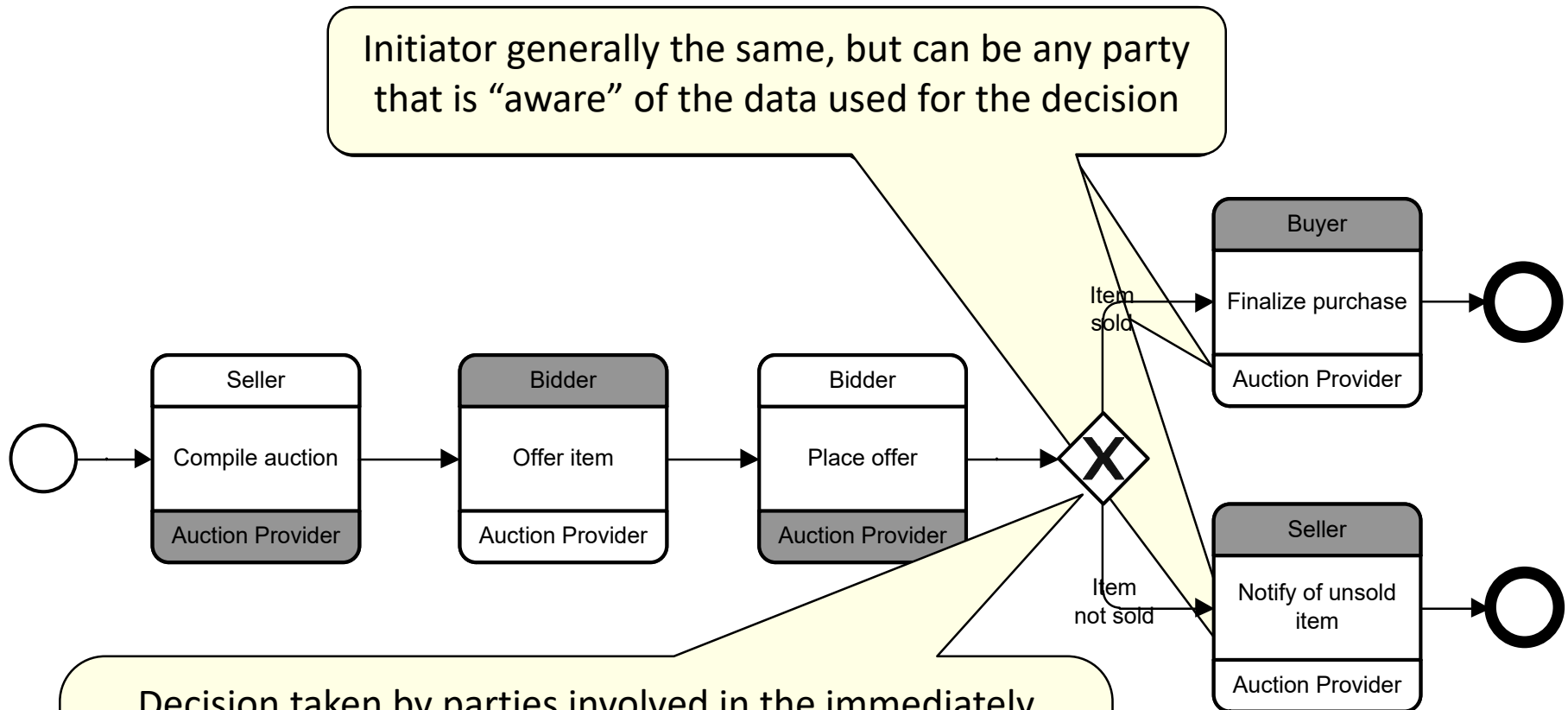Initiator generally the same, but can be any party that is "aware" of the data used for the decision

Decision taken by parties involved in the immediately preceding interaction, based on available data which was **"shared"** between the parties via a message in a preceding interaction. In this case, either Bidder or Auction Provider

If data is not shared: use event-based XOR-split

Seller		Bidder		Bidder
Compile auction	→	Offer item	→	Place offer
Auction Provider		Auction Provider		Auction Provider

Item sold → Buyer / Finalize purchase / Auction Provider

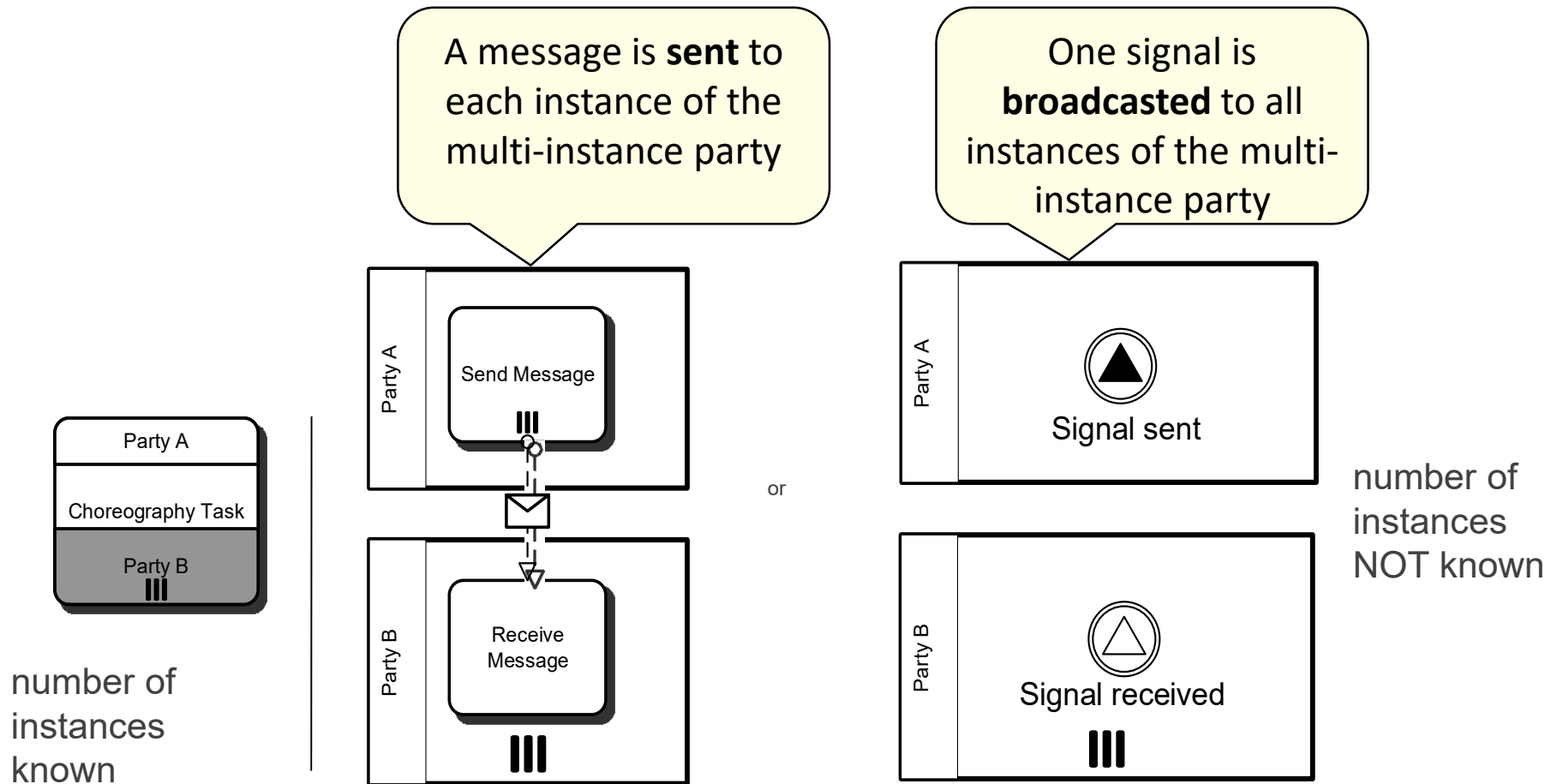Item not sold → Seller / Notify of unsold item / Auction Provider

Choreography Task – Internal Markers

Only one of the loop or multi-instance applicable
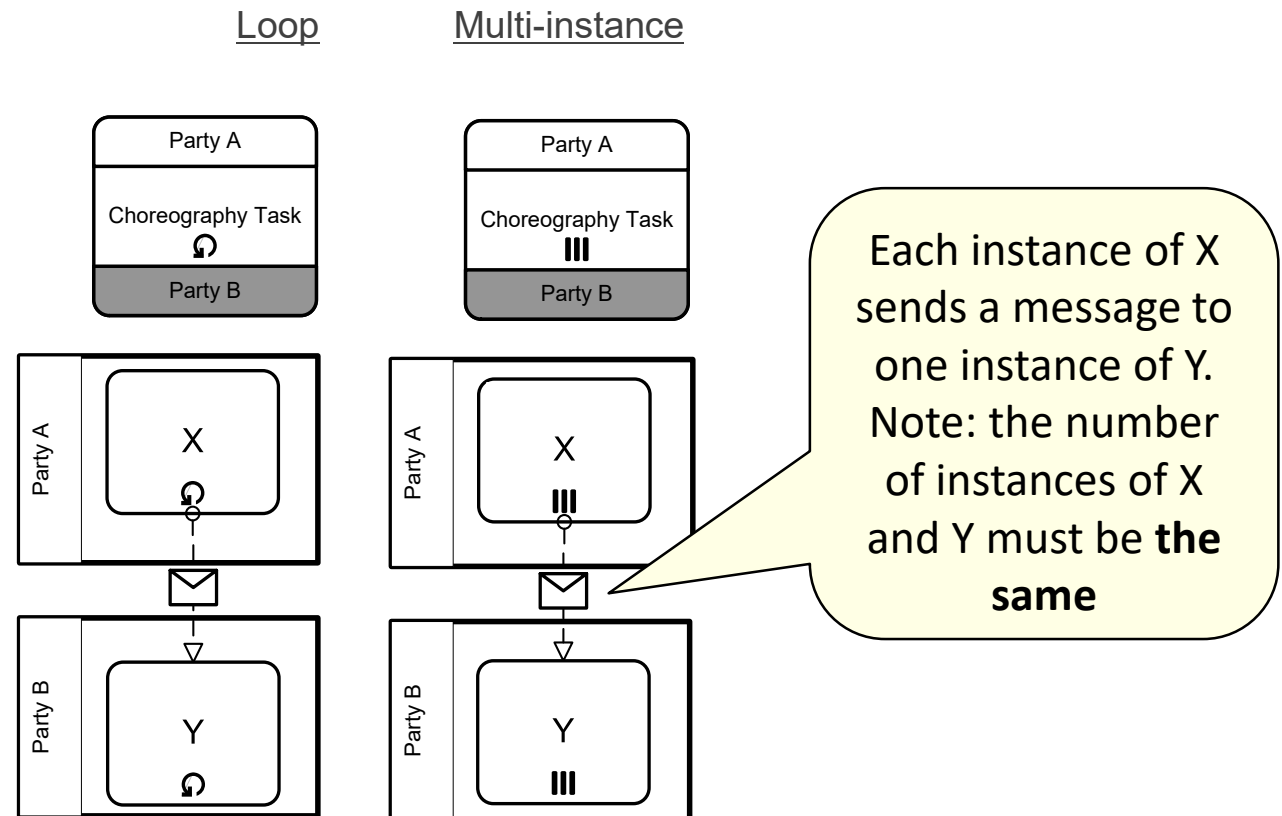
Multi-instance Party marker

Parties may be multi-instance, e.g. customers or shippers

A message is **sent** to each instance of the multi-instance party

One signal is **broadcasted** to all instances of the multi-instance party

Party A

Choreography Task

Party B
III

number of instances known

Party A

Send Message
III

Party B

Receive Message
III

or

Party A

Signal sent

Party B

Signal received
III

number of instances NOT known

Example: Choreography of an auction

Where can we use the MI party in the auction example?
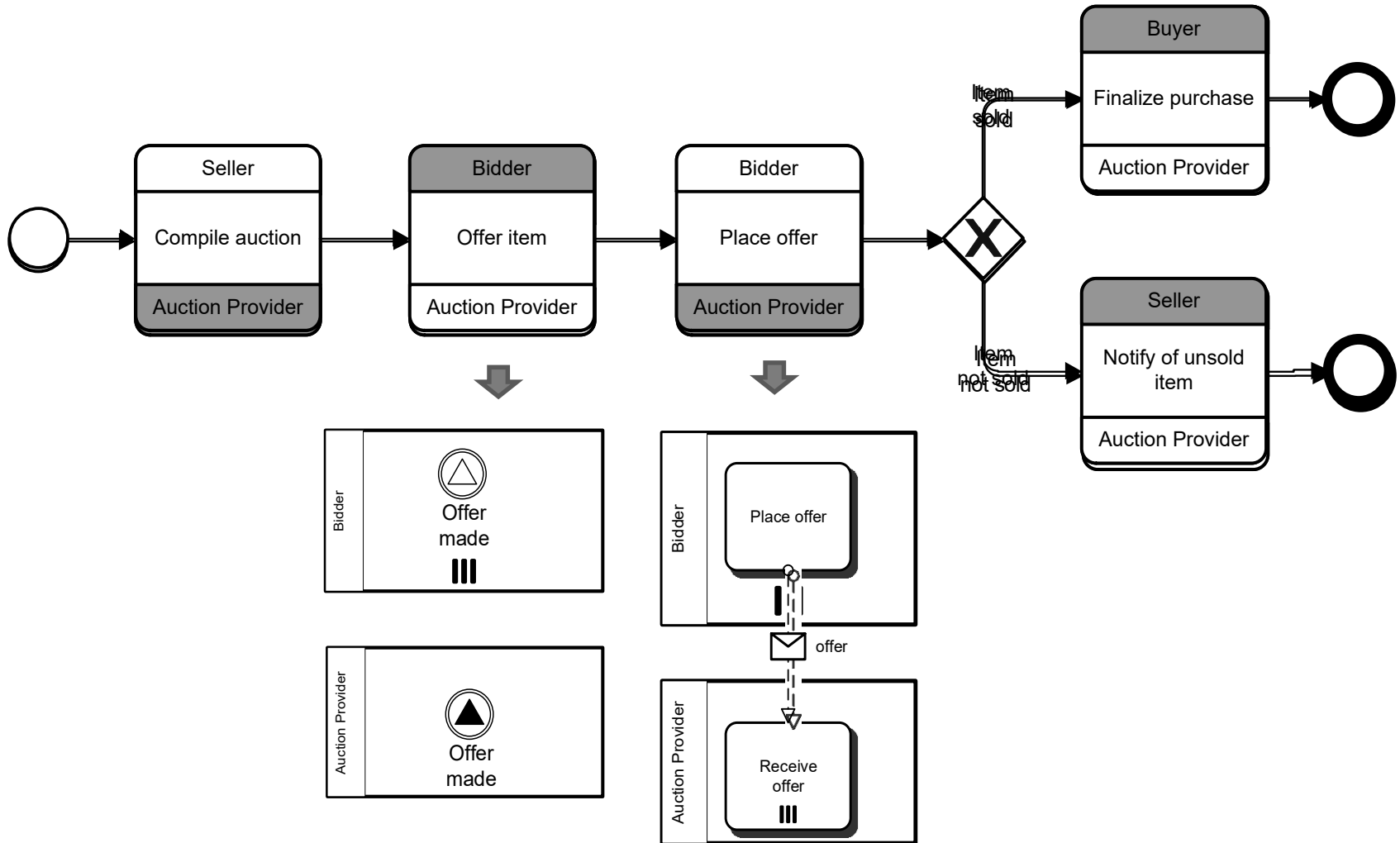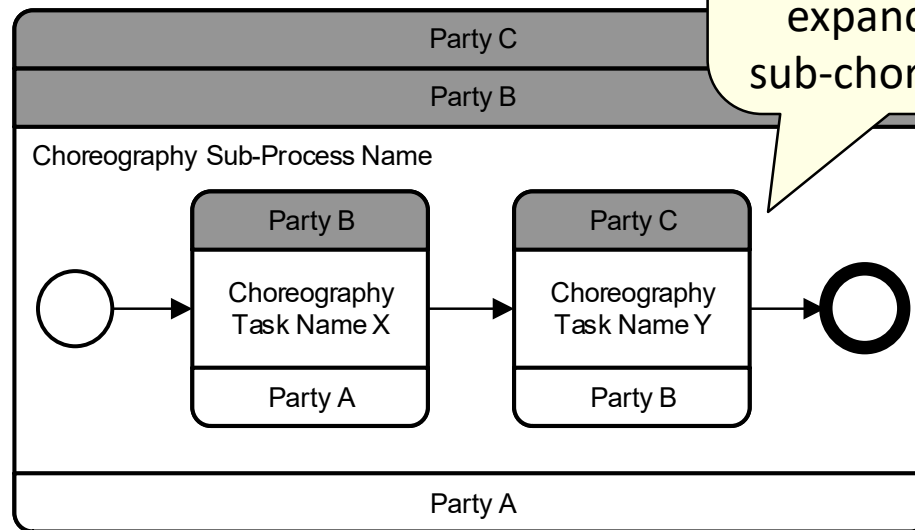
Sub-Choreography

- Compound activity of a choreography
- ~~at least~~ two business parties
- ~~activity and MI party markers are~~

Bands beyond first two are **optional** and their position is **irrelevant**

Exact order of messages can only be seen when expanding the sub-choreography
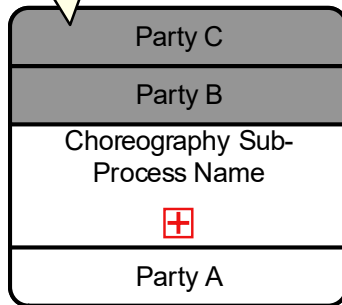
All we know from this is that **A sends a message to B** and then C is involved

| Party C |
| Party B |
| Choreography Sub-Process Name ⊞ |
| Party A |

| Party C |
| Party B |

Choreography Sub-Process Name

| Party B |
| Choreography Task Name X |
| Party A |

| Party C |
| Choreography Task Name Y |
| Party B |

| Party A |

What is this choreography doing?

The interactions following an event-based split must all have the same receiver or the same initiator, which must be involved in the interaction(s) immediately preceding the split
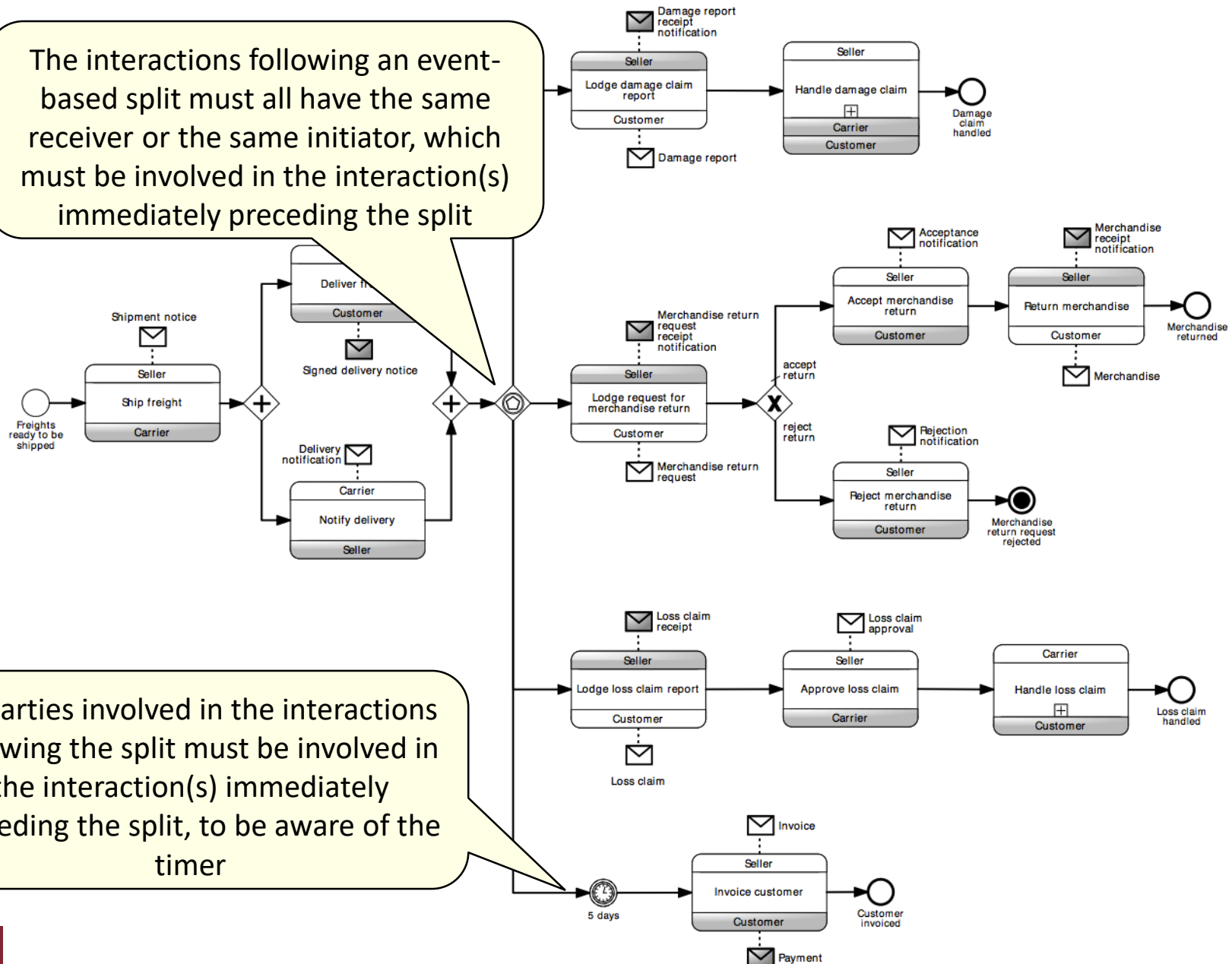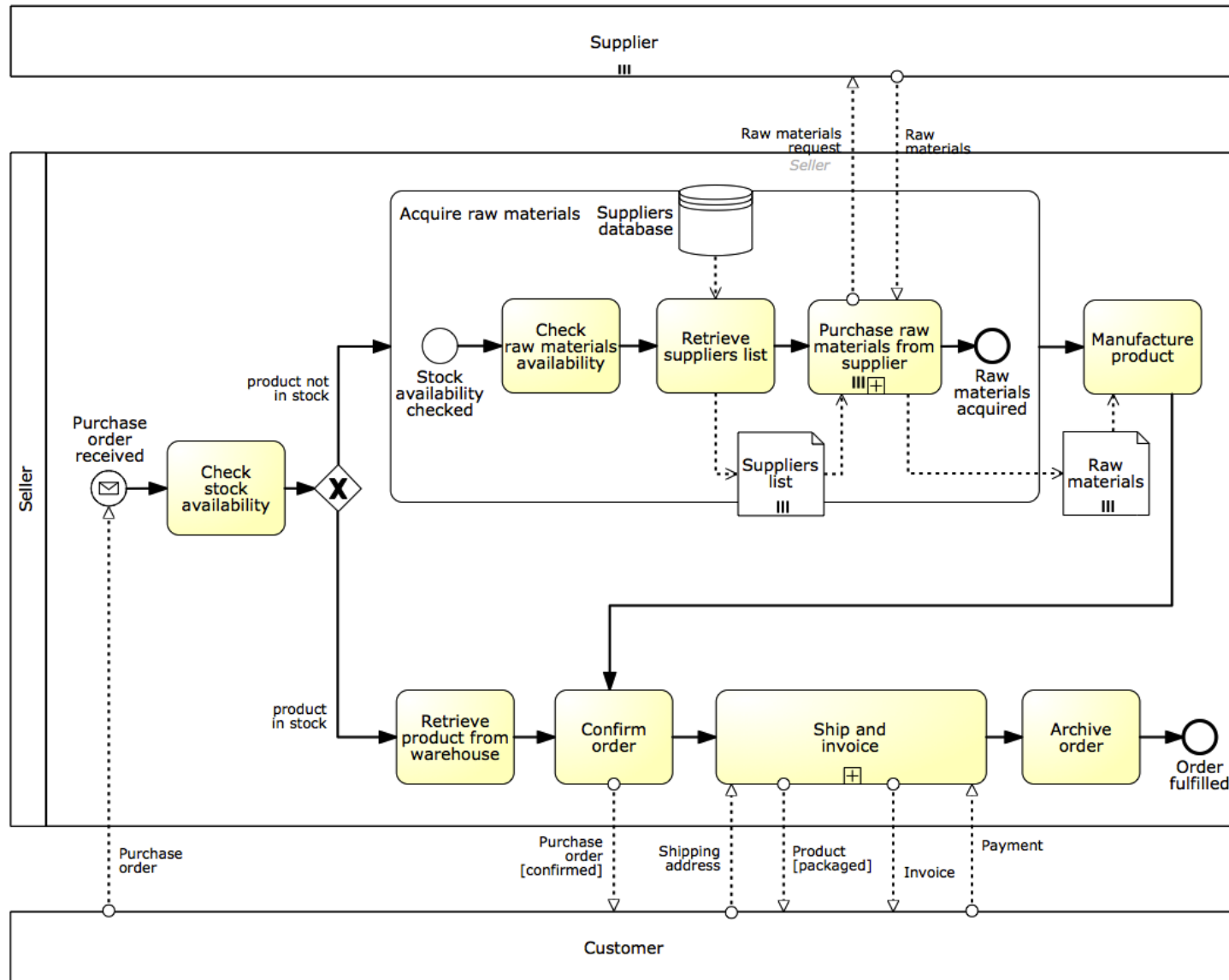
All parties involved in the interactions following the split must be involved in the interaction(s) immediately preceding the split, to be aware of the timer

From Choreography to Collaboration diagram

Use this diagram as a template to build the corresponding collaboration diagram
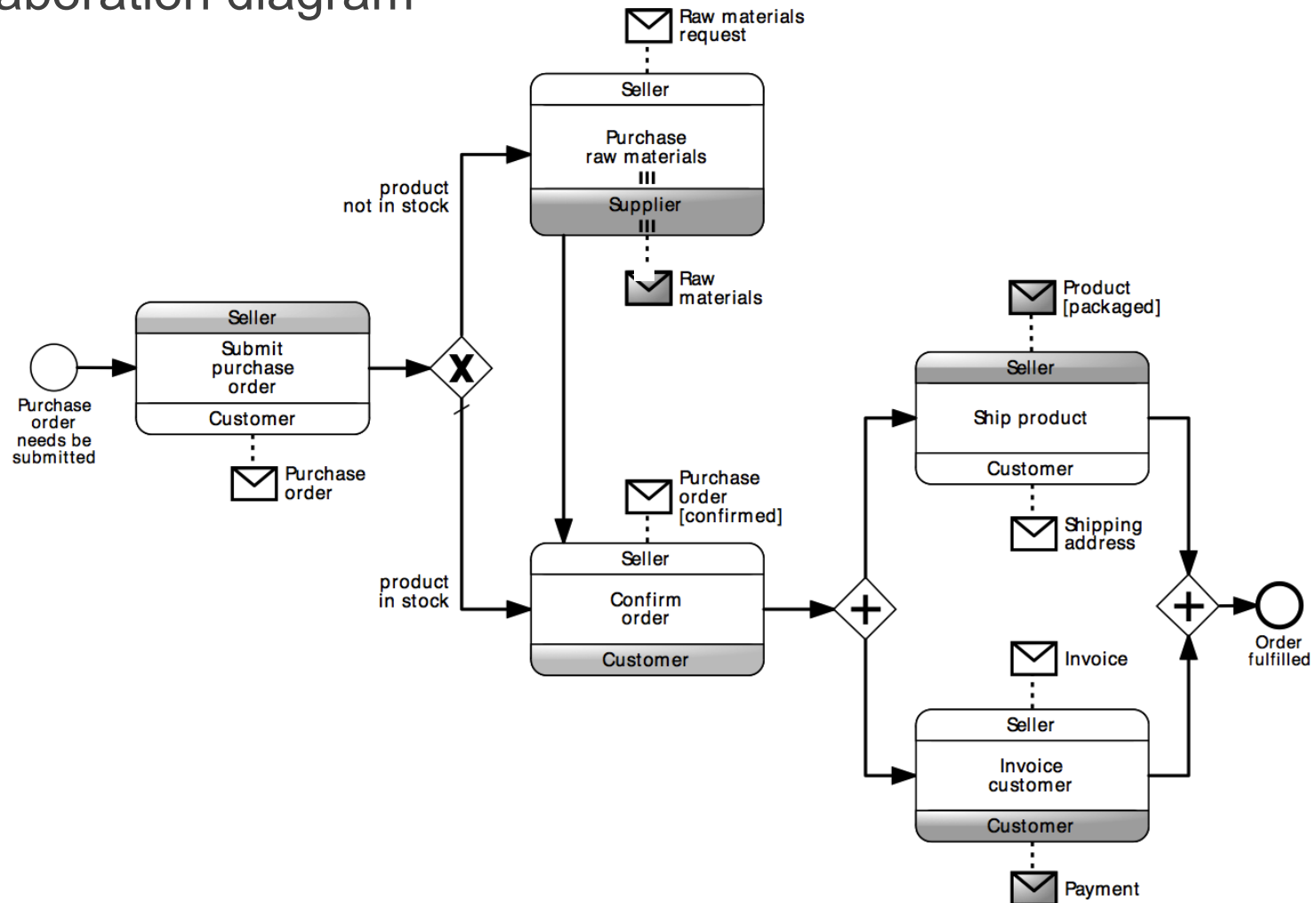
A possible solution

So, what's the difference between collaboration and choreograph diagrams?