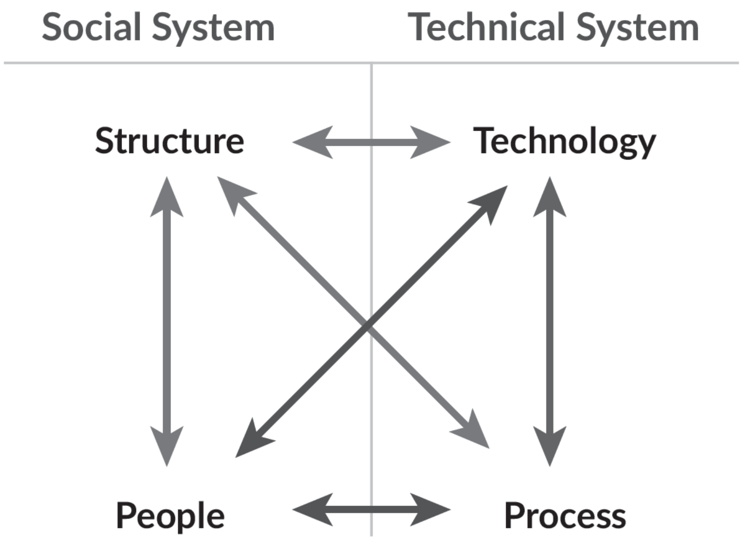
Low level models – process

http://softeng.polito.it

Low level models

- Structure
- People
- Process
  - Process
  - Data
- Technology



Social System

Technical System

Structure

Technology

People

Process

- Goal: describe activities, flow of actions

Process models

- Text / tables
- BPMN
- UML activity diagram

Free text

- Receive order
- Manage production of order
- Manage warehouse
- Record order payment and issue receipt
- Hire new employee

Tables

Process name	input	output	description	Organizational units involved
Receive order	order (from customer)	Order accepted	Verify order consistency Verify availability of items ordered Verify price of items Verify payment Send order to production/ warehouse Send order confirmation to customer	Customer, sales
Manage production of order	Order (from sales)	Items in order available	Collect raw materials for items in order Produce components from raw materials, assemble components Package order Notify order available for shipping	Warehouse, factory

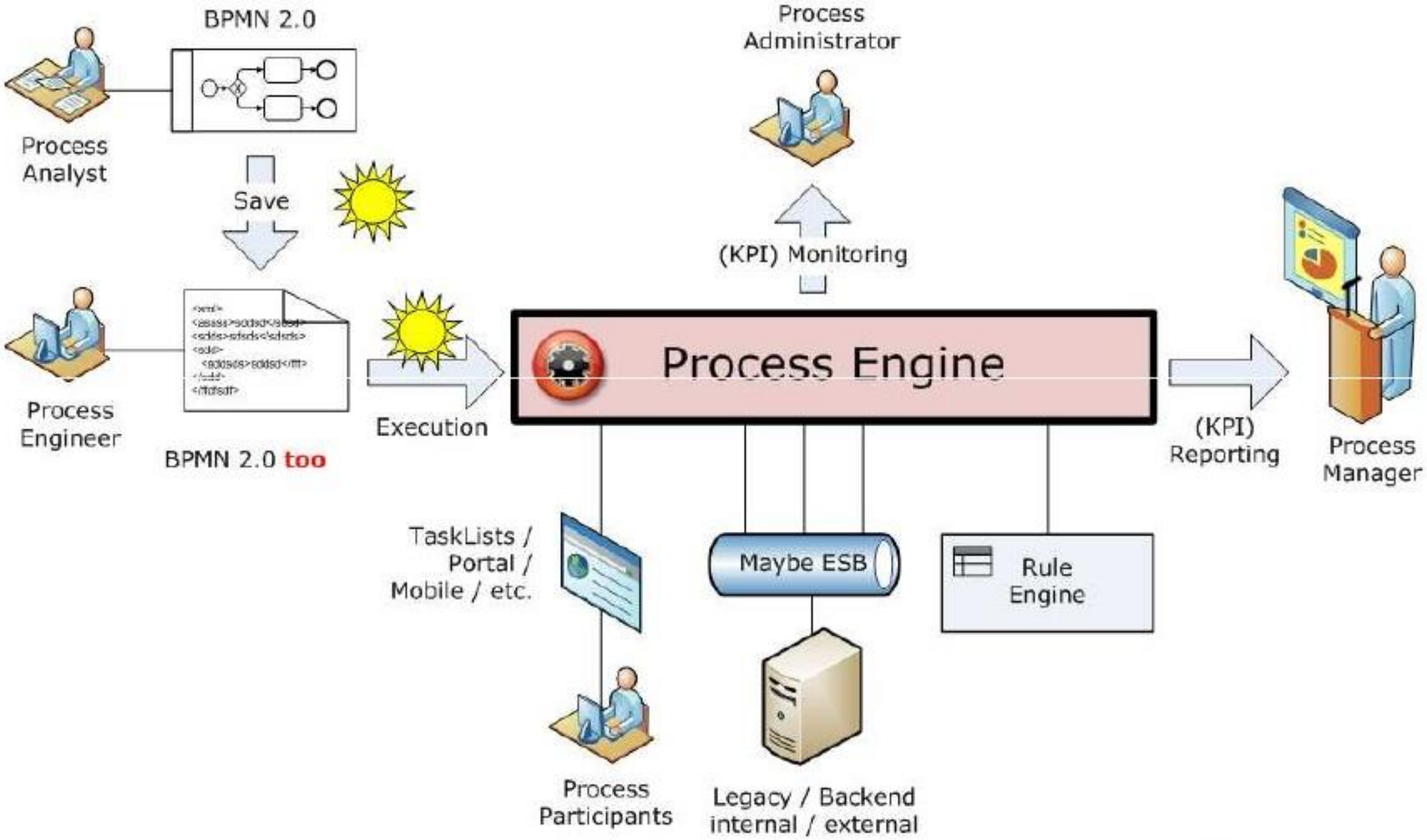
Tables

Process name	input	output	description
Entrance declaration		Entrance declaration (name, SSN, entry date, origin of trip, final destination, address)	Citizen declares his/her entrance in Italy from a Covid-risk country. Ministry of health receives the declaration, records it, and dispatches it to the relevant ASL
Declaration check	Citizen (name, ssn)	Check result	Border authority can check, when the citizen enters, if she has submitted the entrance declaration
Swab reservation	Entrance declaration	Swab reservation (date, hour, place)	The relevant ASL defines when and where the swab will be performed. The reservation is sent to the citizen and to the unit in charge of doing the swab
Swab execution	Citizen (name, SSN)	Swab (physical) Swab record (date, time)	An ASL unit performs the swab on the citizen
Swab analysis and result	Swab (physical) Swab record	Swab result	A lab analyzes the swab, and sends the result to the citizen, his/her base doctor, the ASL itself
Monitor			Ministry can check number of entrances, per country, and % of positive swabs per country of entrance / per ASL / per point of entrance, or else

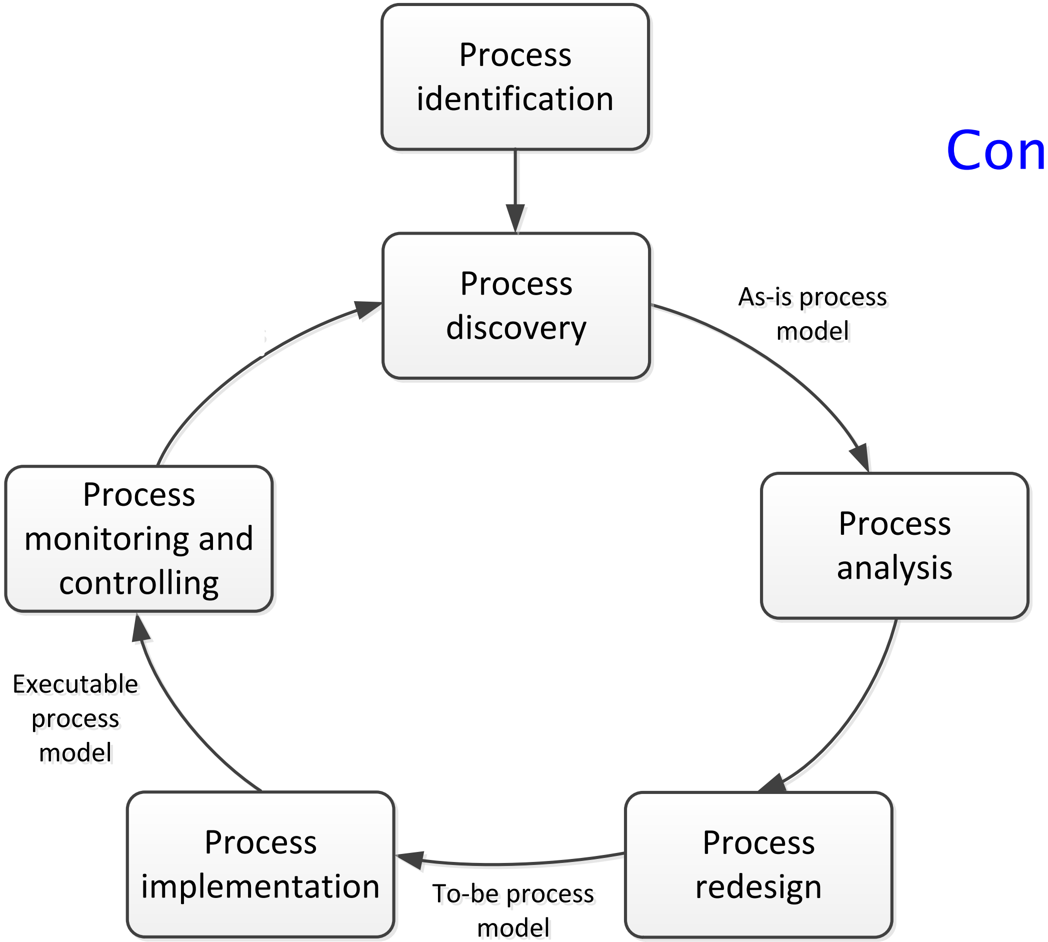
BPMN

- Usage
  - Modeling (old , new)
  - Enactment

Desired Architecture with BPMN 2.0



Business process lifecycle

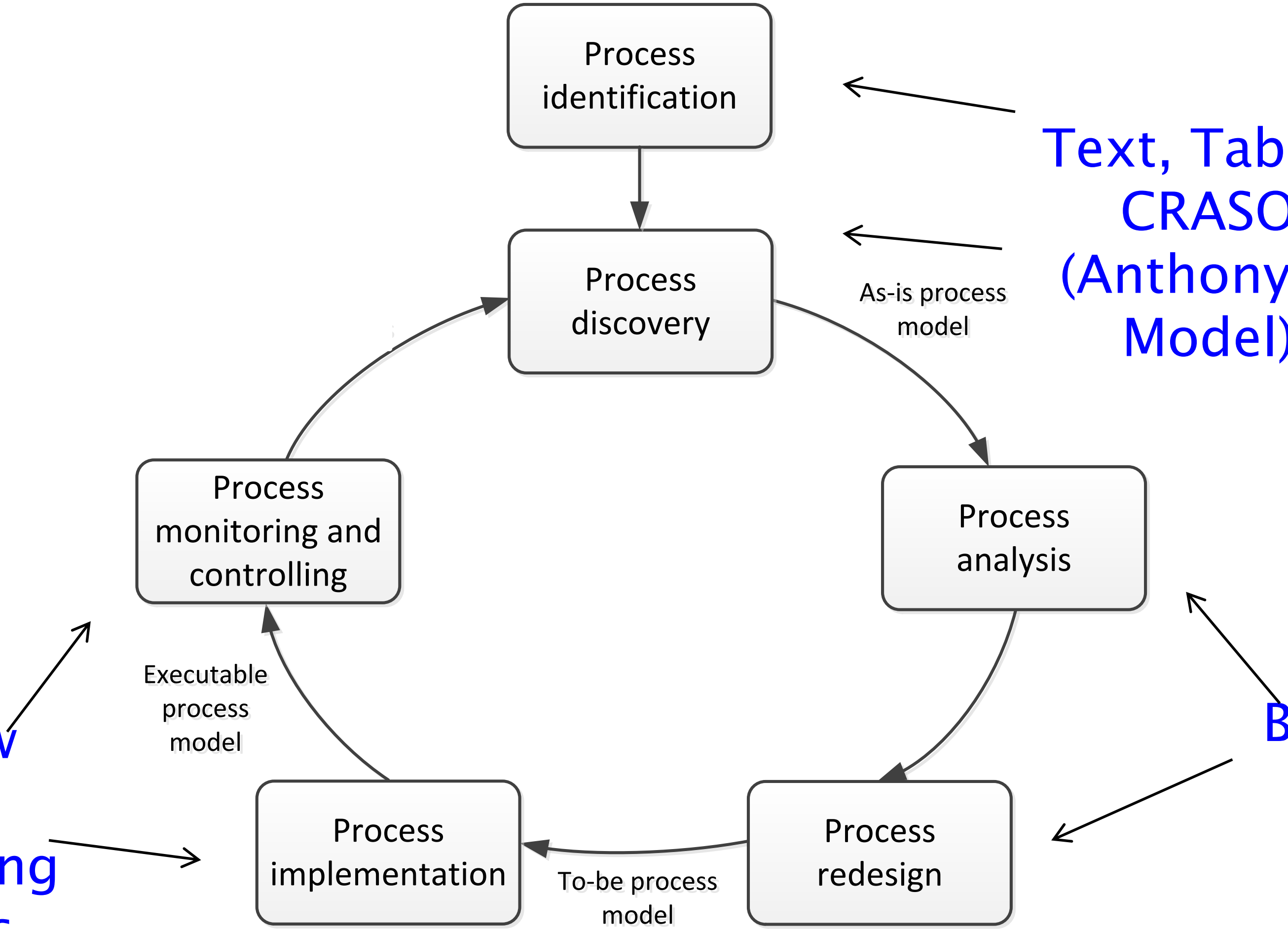


Conceptual

Executable

Conceptual

Levels of process description



Process identification

Process discovery

As-is process model

Process analysis

Process redesign

To-be process model

Process implementation

Executable process model

Process monitoring and controlling

Text, Tables, CRASO (Anthony, T Model)

BPMN, hig level

BPMN, low level, programming languages

Processes, conceptual level

- Free text
    - Receive order
    - Manage production of order
    - Manage warehouse
    - Record order payment and issue receipt
    - Hire new employee
    - ..

Processes, conceptual level

- T model
  - Support
    - Hire new employee (HR)
    - Record order payment and issue receipt (accounting)
  - Primary
    - Receive order
    - Manage production of order
    - Manage warehouse

Processes

- Conceptual level
  - BPMN, high level
- Executable level
  - BPMN, low level (BPMN + programming language aka Java)

BPMN

- **BPMN** Business Process Modeling Notation: A graphical representation for specifying business processes

- Dimensions
  - Functional (what is done and when)
  - Organizational (who does what)
  - Data

- BPMN is meant to represent processes at
  - Medium level of detail
    - process analysis phase
  - High level of detail
    - Process implementation phase

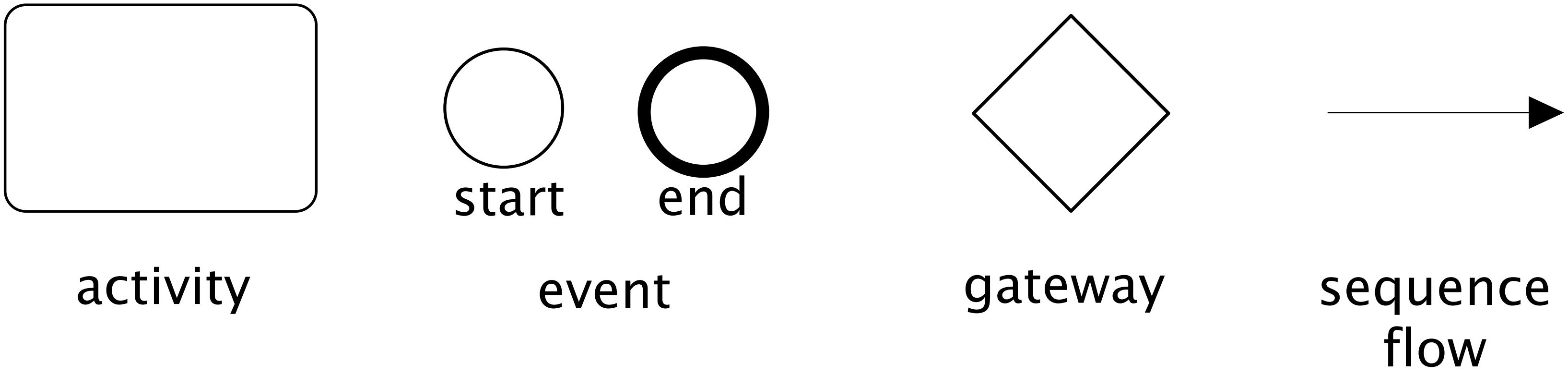
BPMN

Business Process Model and Notation

- OMG standard (nowadays BPMN 2.0)
- Both for conceptual and executable models
- Supported by numerous tools: bpmn.org lists over 70 tools, incl.
    - **Signavio**
    - Bizagi Process Modeler
    - Cameo Business Analyst
    - Camunda

BPMN from 10,000 miles…

A BPMN process model is a graph consisting of four types of **core elements**:



activity

start end

event

gateway

sequence flow

Let’s start modeling

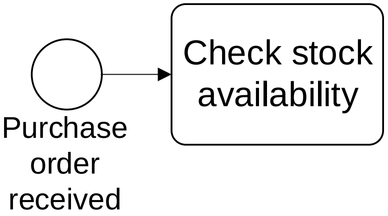
Order-to-cash

An order-to-cash process is triggered by the receipt of a purchase order from a customer. Upon receipt, the purchase order has to be checked against the stock to determine if the the requested item(s) are available. Depending on stock availability the purchase order may be confirmed or rejected.

If the purchase order is confirmed, an invoice is emitted and the goods requested are shipped. The process completes by archiving the order.

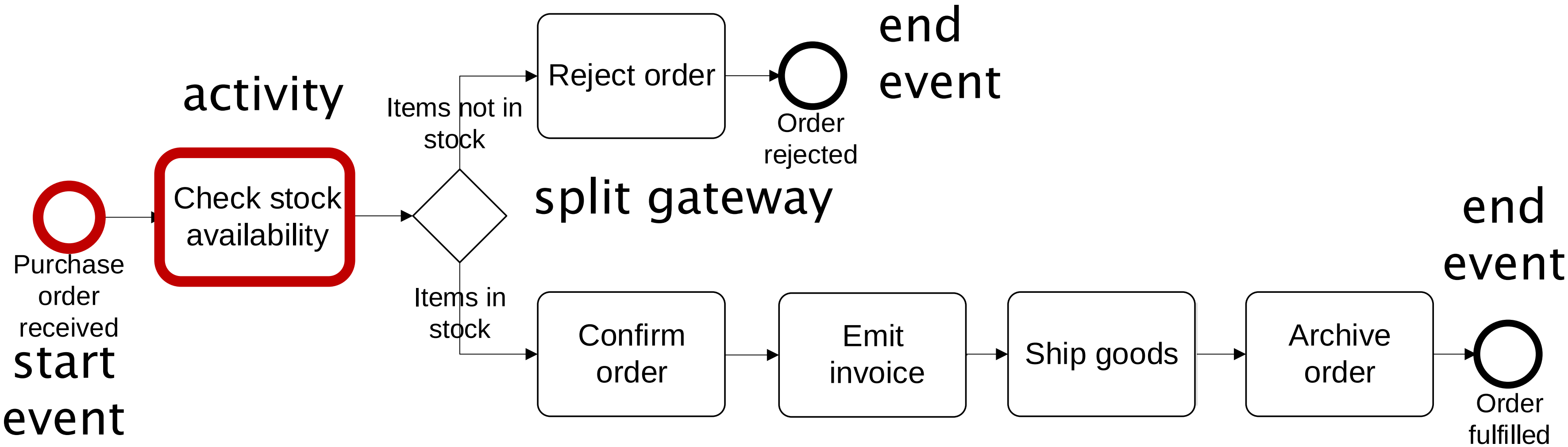
BPMN Model

Order-to-cash



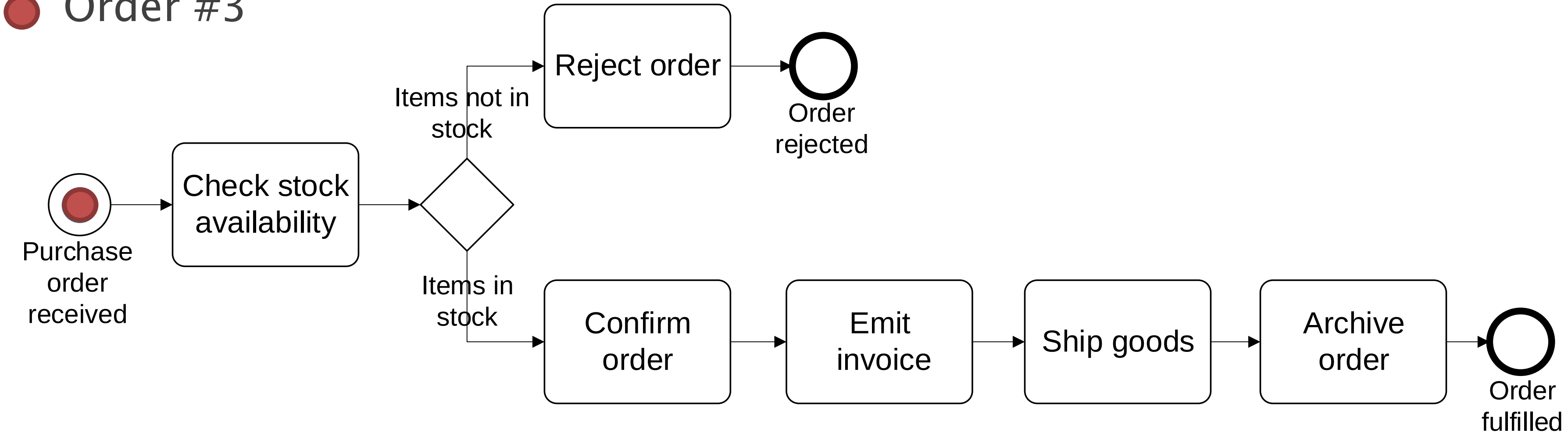
BPMN Model

Order-to-cash



Execution of a process model: tokens

- ● Order #1
- ● Order #2
- ● Order #3

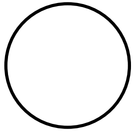


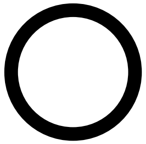
Process model vs process instances

- Process model
  - Aka UML class diagram
  - Aka class X {} in Java
- Process instance, tokens
  - Aka UML instance diagram, instances
  - Aka new X; in Java

A little bit more on events…

A *start event* triggers a new process instance
by generating a token that traverses the sequence flow (“tokens source”)

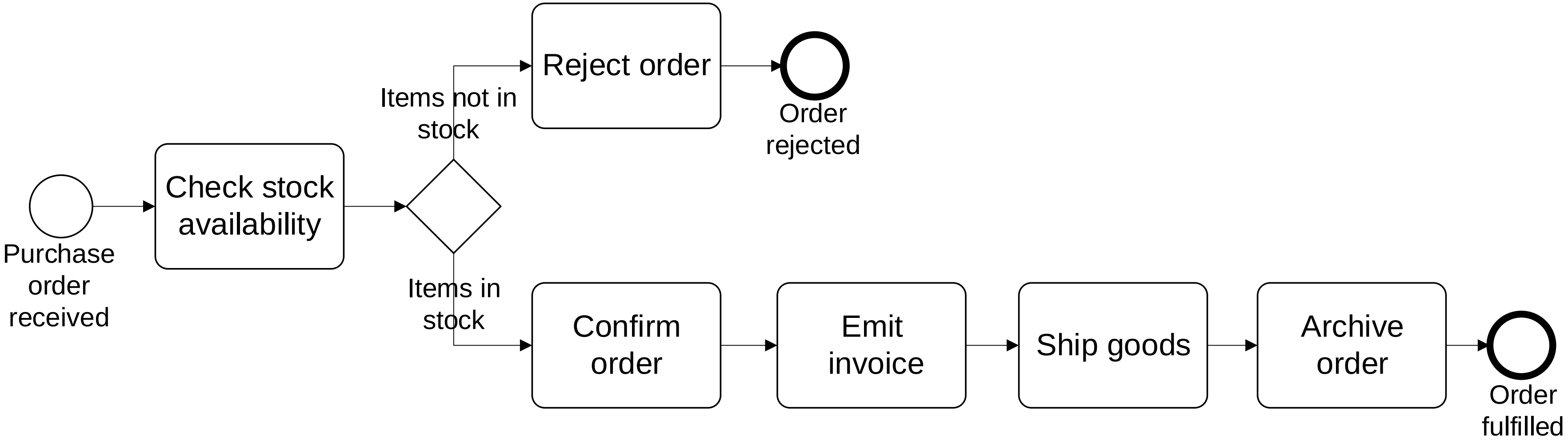




An *end event* signals that a process instance has completed with a given outcome by consuming a token (“tokens sink”)

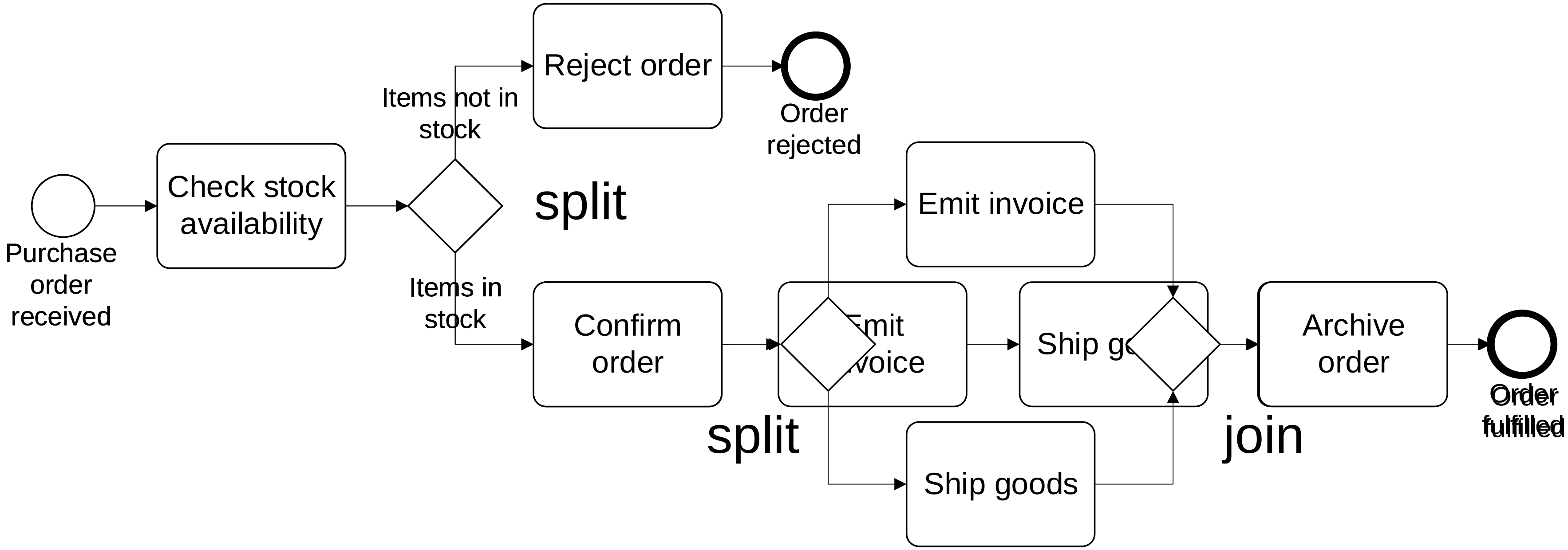
Order-to-cash example revisited…

[…] If the purchase order is confirmed, **an invoice is emitted and the goods requested are shipped (in any order)**. The process completes by archiving the order. […]

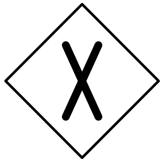


First try

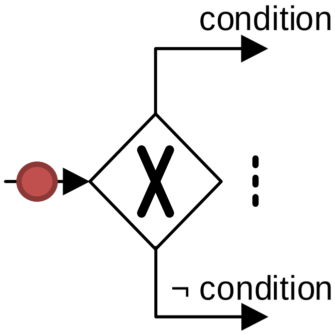
Order-to-cash



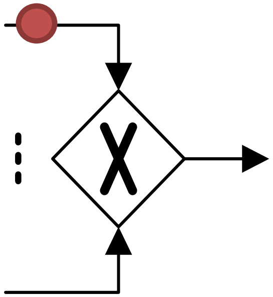
A little more on gateways: XOR Gateway



An *XOR Gateway* captures decision points (XOR-split) and points where alternative flows are merged (XOR-join)



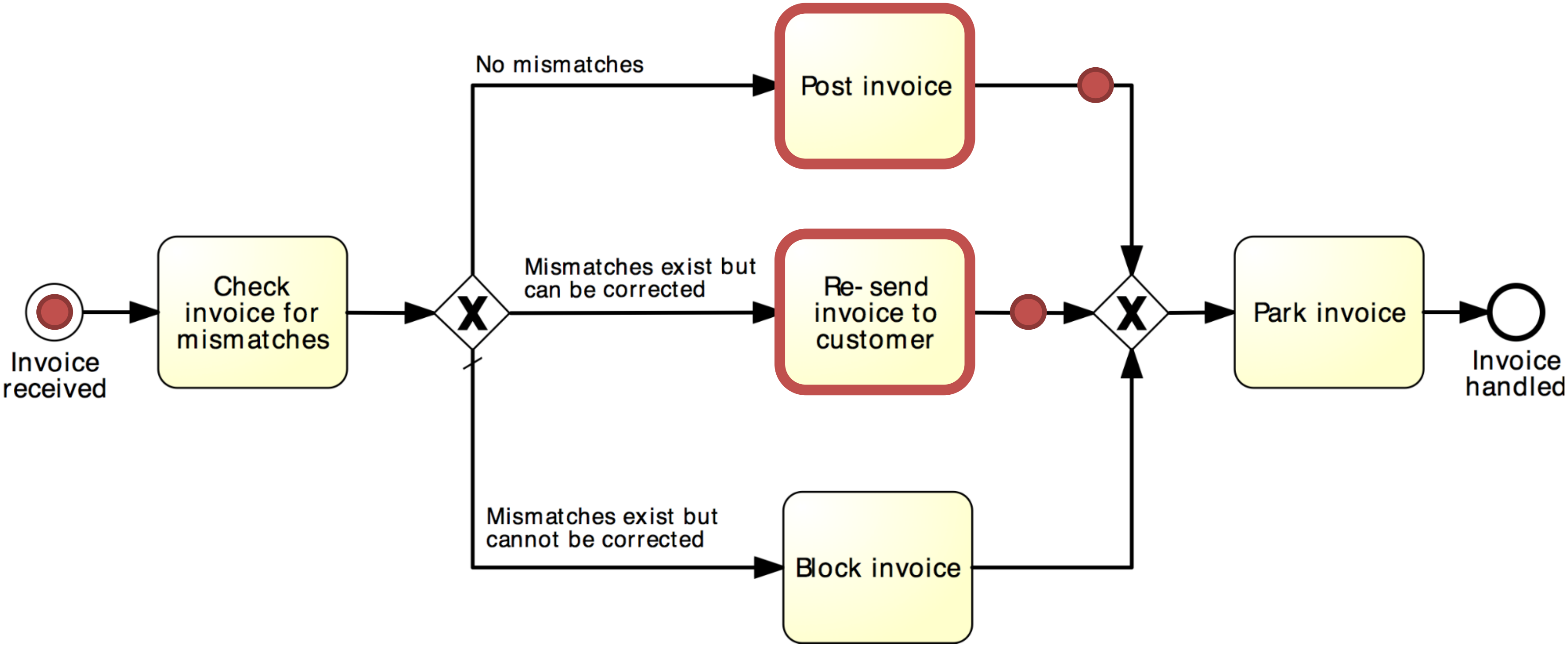
XOR-split ➔ takes **one** outgoing branch



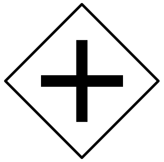
XOR-join ➔ proceeds when **one** incoming branch has completed

Example: XOR Gateway

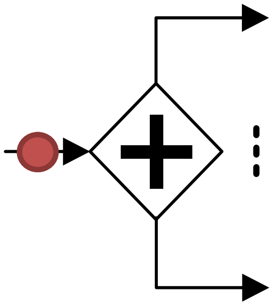
Invoice checking process


No mismatches
Post invoice
Mismatches exist but can be corrected
Re- send invoice to customer
Invoice received
Check invoice for mismatches
Park invoice
Invoice handled
Mismatches exist but cannot be corrected
Block invoice

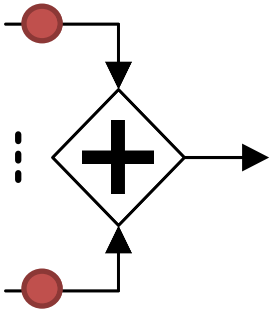
A little more on gateways: AND Gateway



An *AND Gateway* provides a mechanism to create and synchronize “parallel” flows.



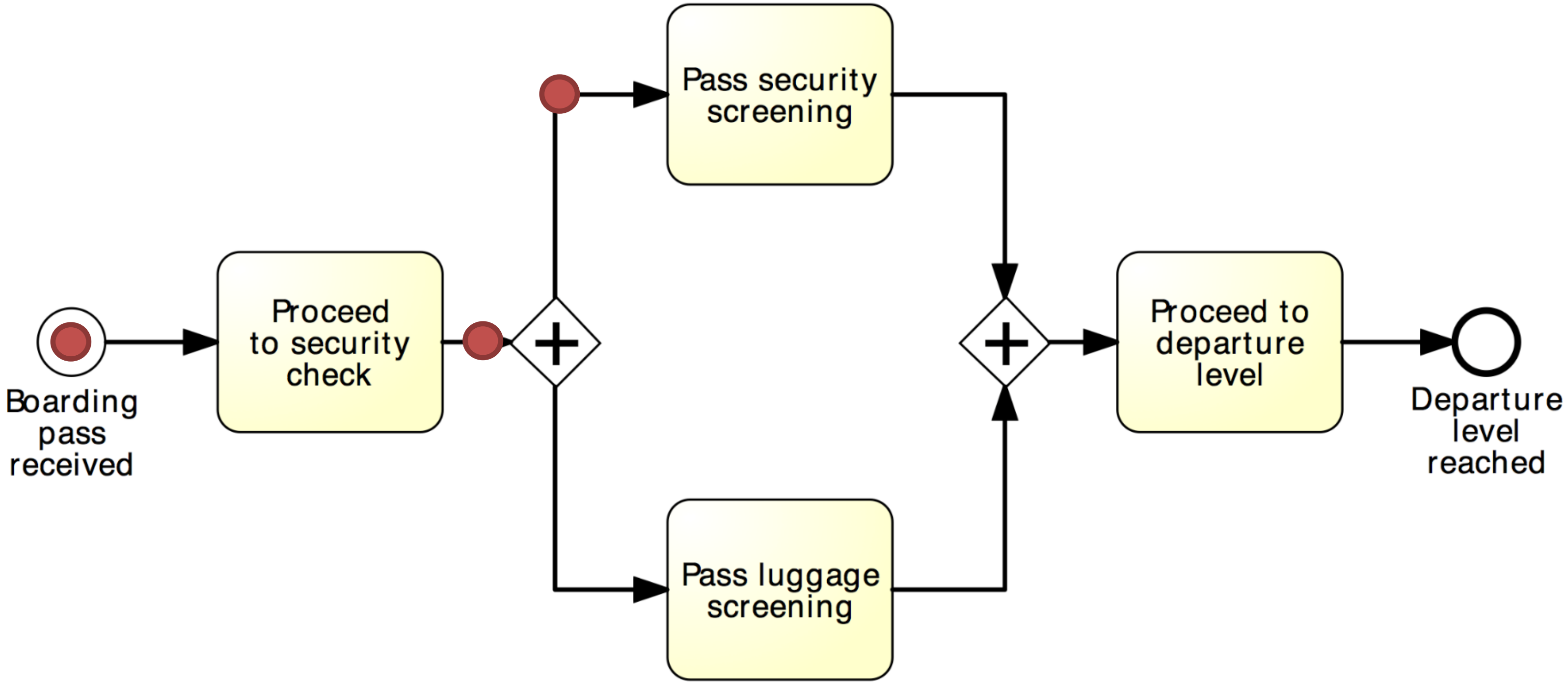
AND-split ➔ takes **all** outgoing branches



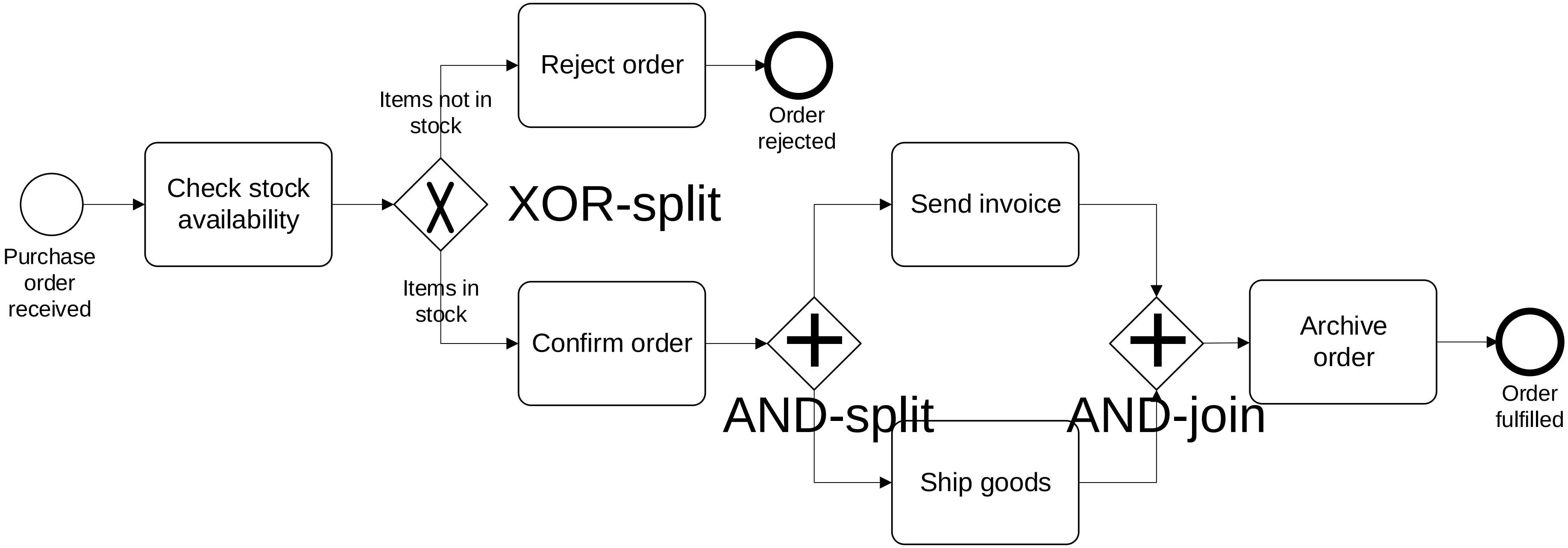
AND-join ➔ proceeds when **all** incoming branches have completed

Example: AND Gateway

Airport security check

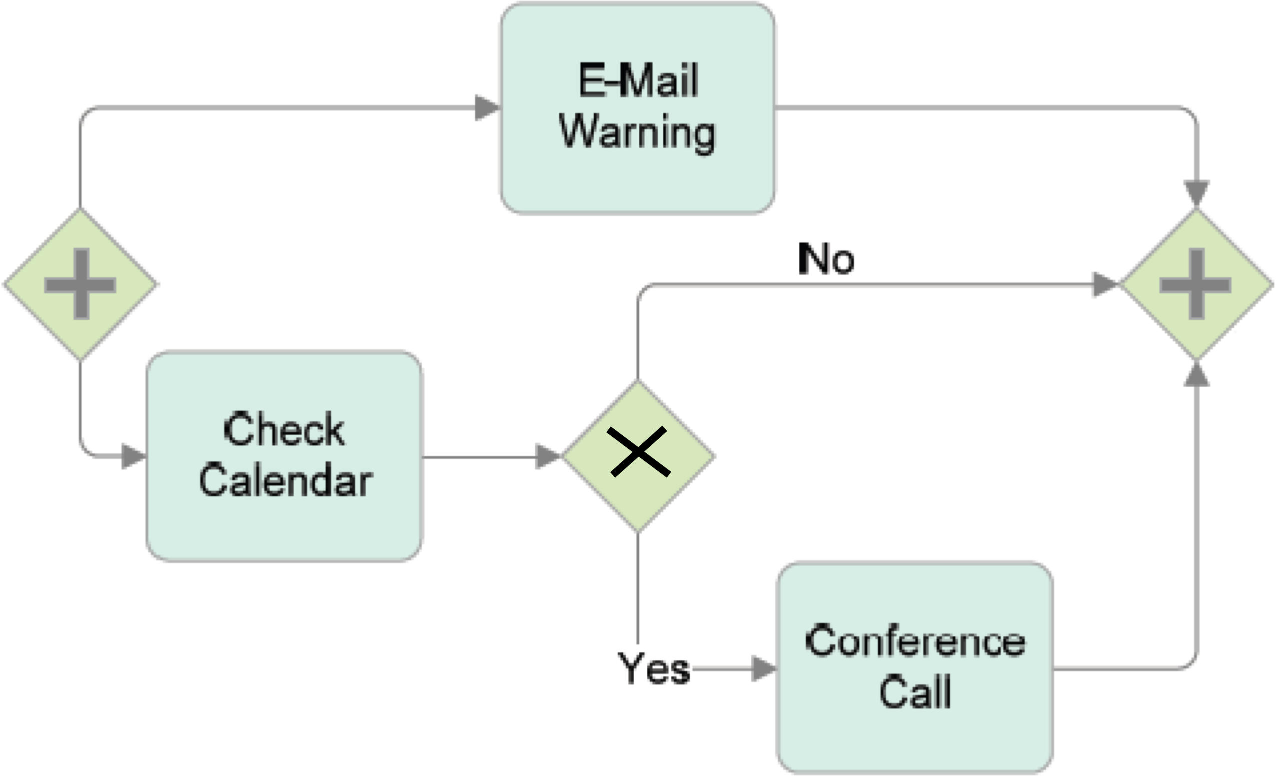


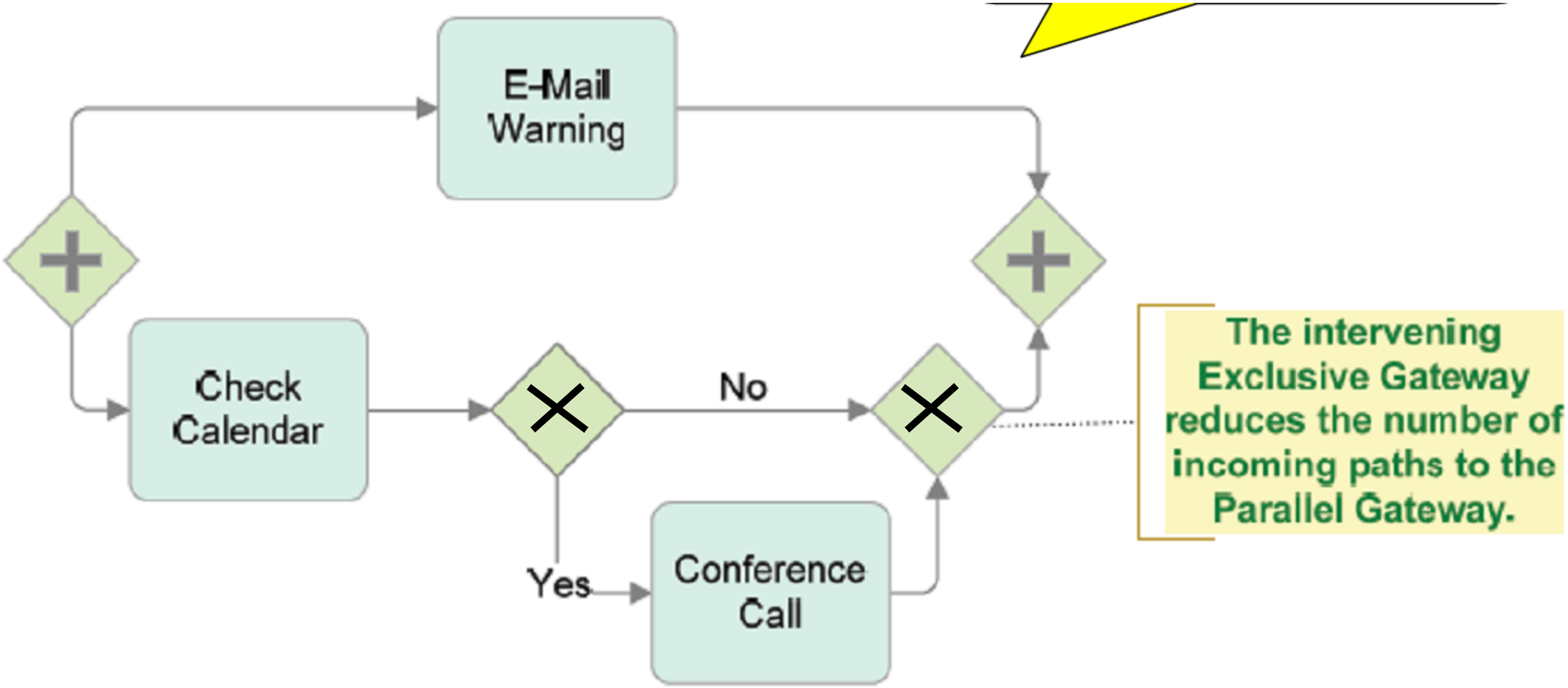
Revised order-to-cash process model



- Remark: order could be
  - Send invoice, then ship goods
  - Ship goods then send invoice
  - Send invoice – ship goods at the same time

Result?





The intervening Exclusive Gateway reduces the number of incoming paths to the Parallel Gateway.

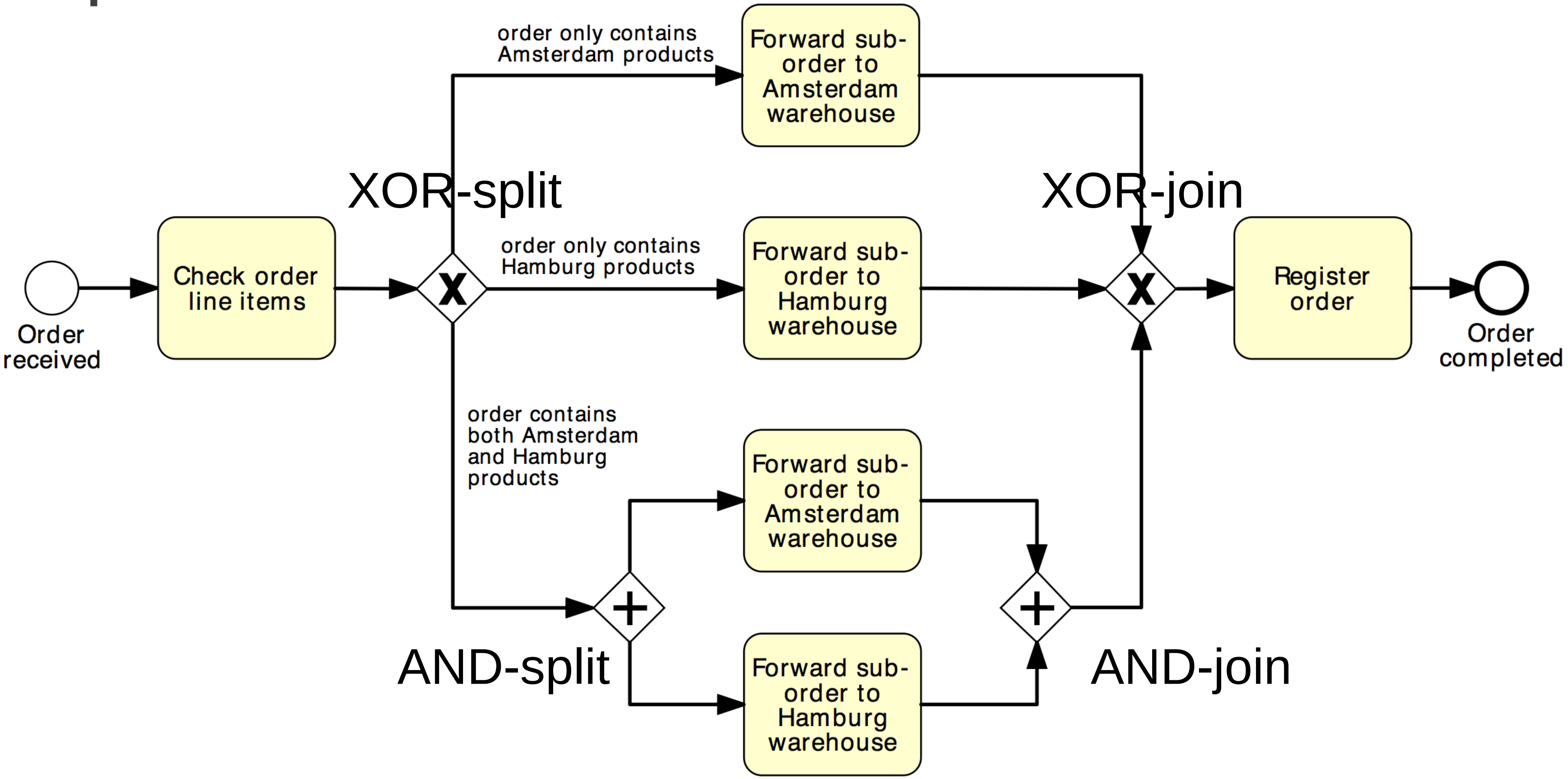
Between XOR and AND

Order distribution process

A company has two warehouses that store different products: Amsterdam and Hamburg. When an order is received, it is distributed across these warehouses: if some of the relevant products are maintained in Amsterdam, a sub-order is sent there; likewise, if some relevant products are maintained in Hamburg, a sub-order is sent there. Afterwards, the order is registered and the process completes.

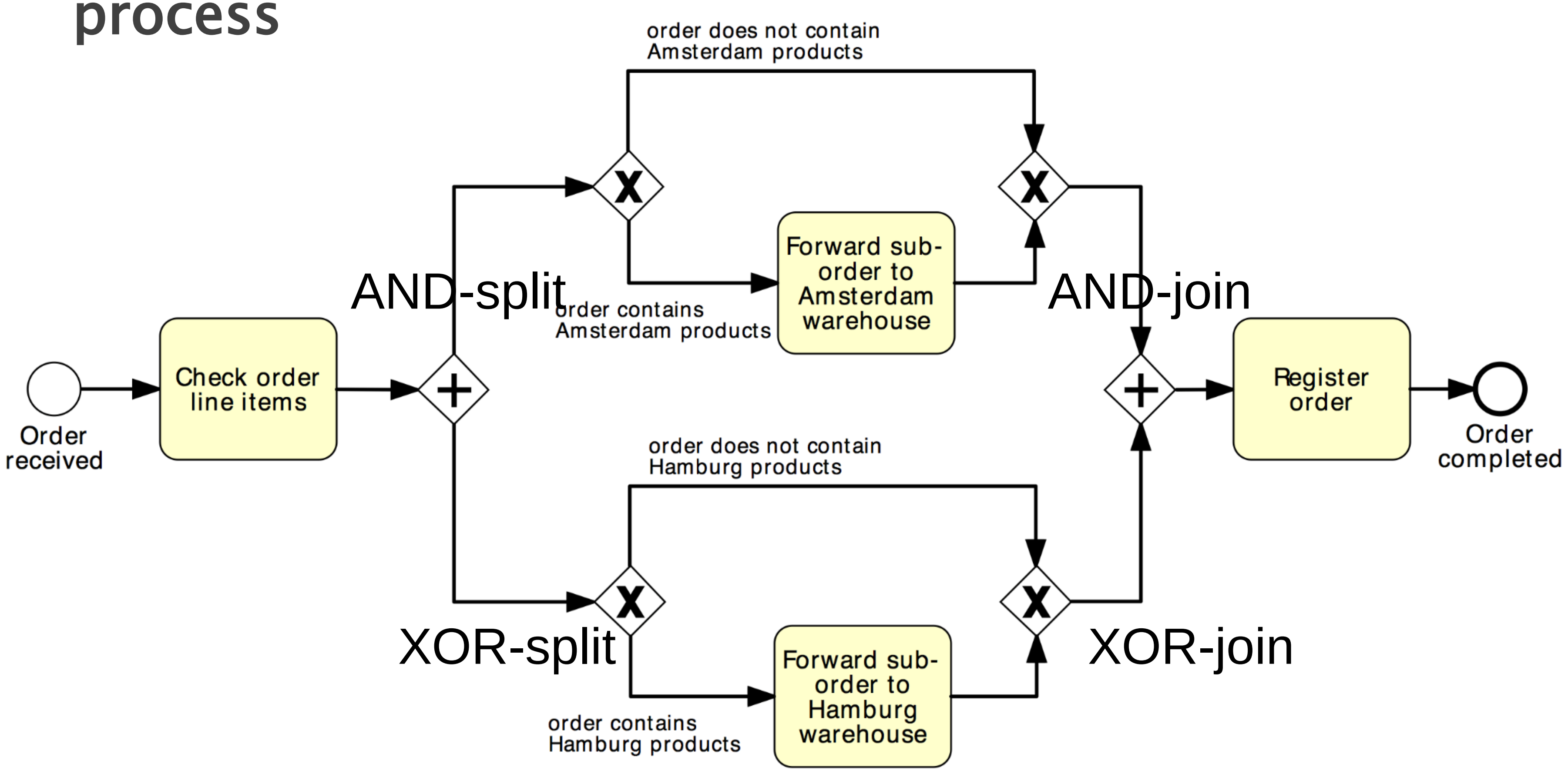
Solution 1

Order distribution process

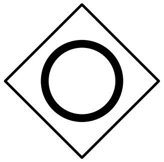

order only contains Amsterdam products
Forward sub-order to Amsterdam warehouse
XOR-split
XOR-join
order only contains Hamburg products
Forward sub-order to Hamburg warehouse
Order received
Check order line items
Register order
Order completed
order contains both Amsterdam and Hamburg products
Forward sub-order to Amsterdam warehouse
AND-split
Forward sub-order to Hamburg warehouse
AND-join

Solution 2

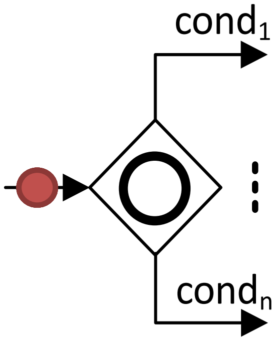
Order distribution process


order does not contain Amsterdam products
AND-split
order contains Amsterdam products
Forward sub-order to Amsterdam warehouse
AND-join
Order received
Check order line items
Register order
Order completed
order does not contain Hamburg products
XOR-split
order contains Hamburg products
Forward sub-order to Hamburg warehouse
XOR-join

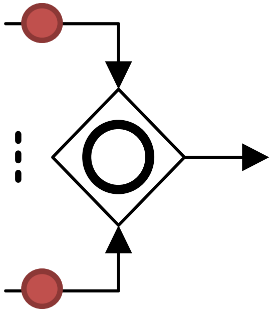
OR Gateway



An *OR Gateway* provides a mechanism to create and synchronize n out of m parallel flows.



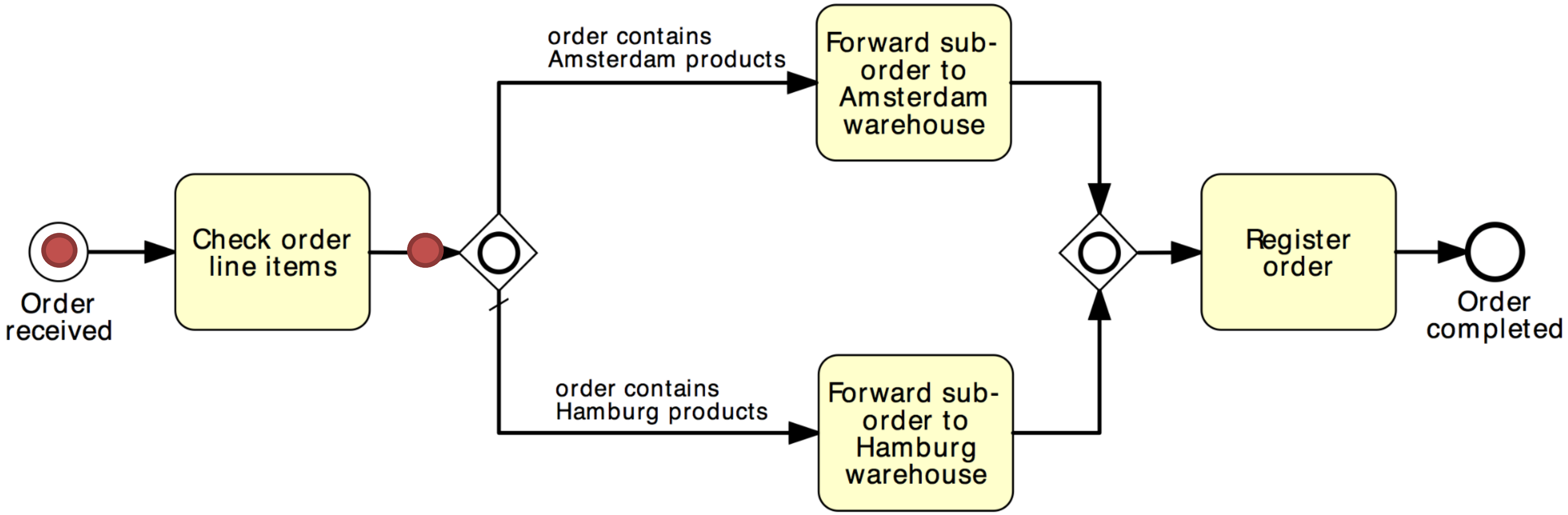
OR-split ➔ takes one or more branches depending on conditions



OR-join ➔ proceeds when all **active** incoming branches have completed

Solution using OR Gateway

Order distribution process


Order received
Check order line items
order contains Amsterdam products
Forward sub-order to Amsterdam warehouse
order contains Hamburg products
Forward sub-order to Hamburg warehouse
Register order
Order completed

BPMN Gateways: sum up

Exclusive (XOR)

- Exclusive decision
  take one branch
- Exclusive merge
  Proceed when one branch has completed

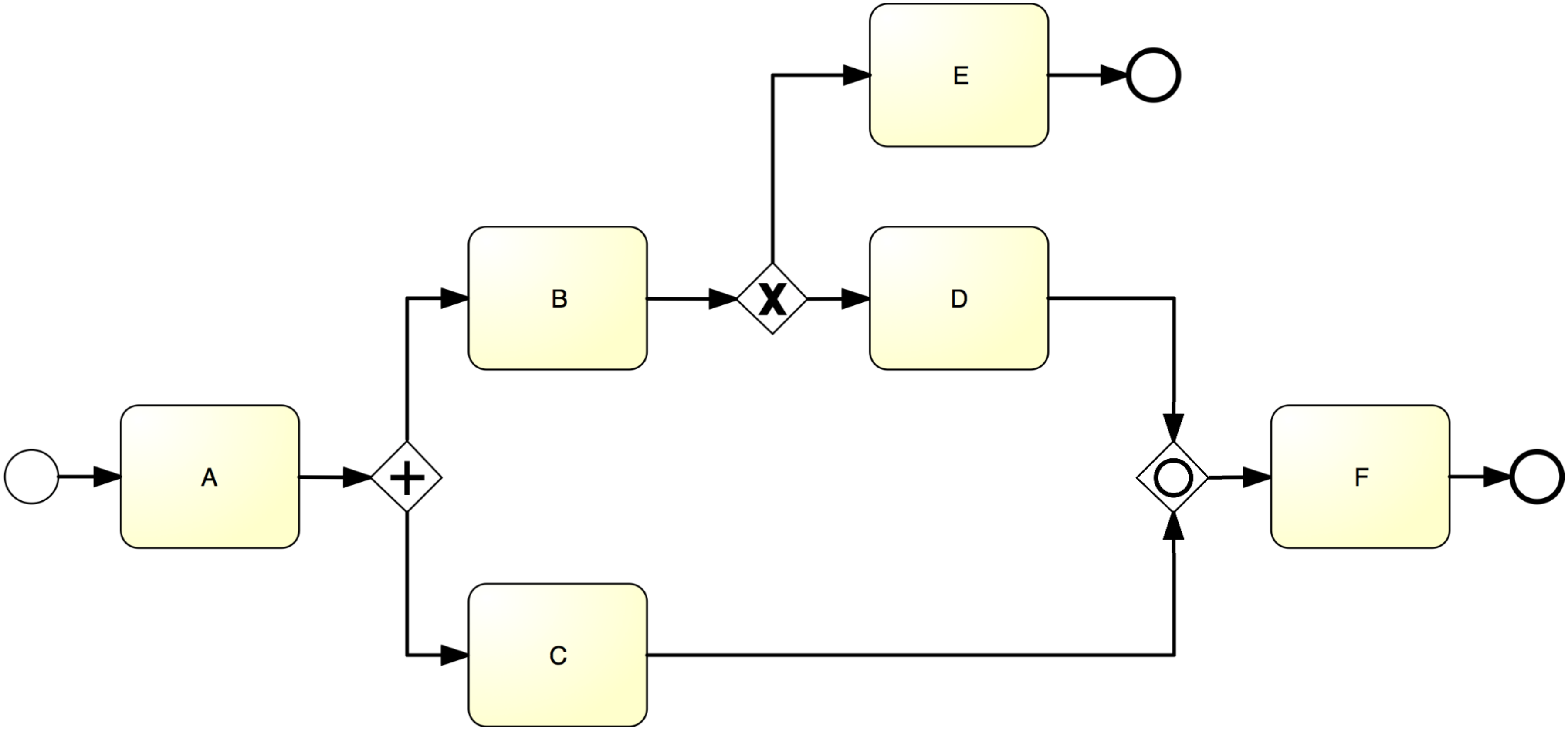
Parallel (AND)

- Parallel split
  take all branches
- Parallel join
  proceed when all incoming branches have completed

Inclusive (OR)

- Inclusive decision
  take one or several branches depending on conditions
- Inclusive merge
  proceed when all active incoming branches have completed

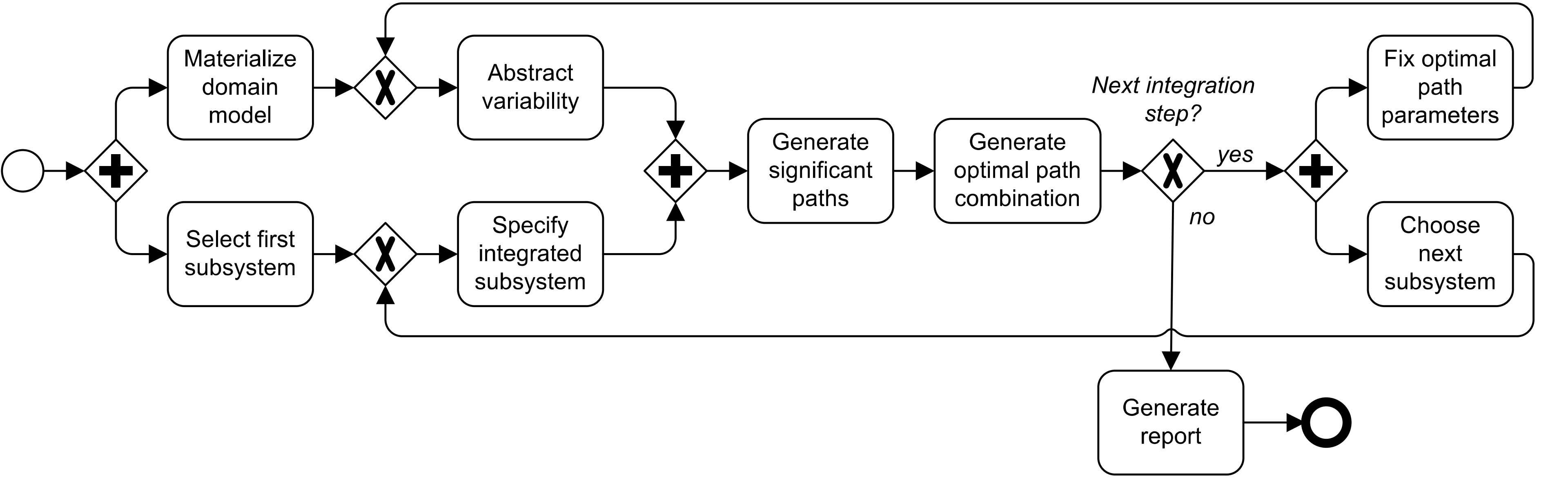
What join type do we need here?

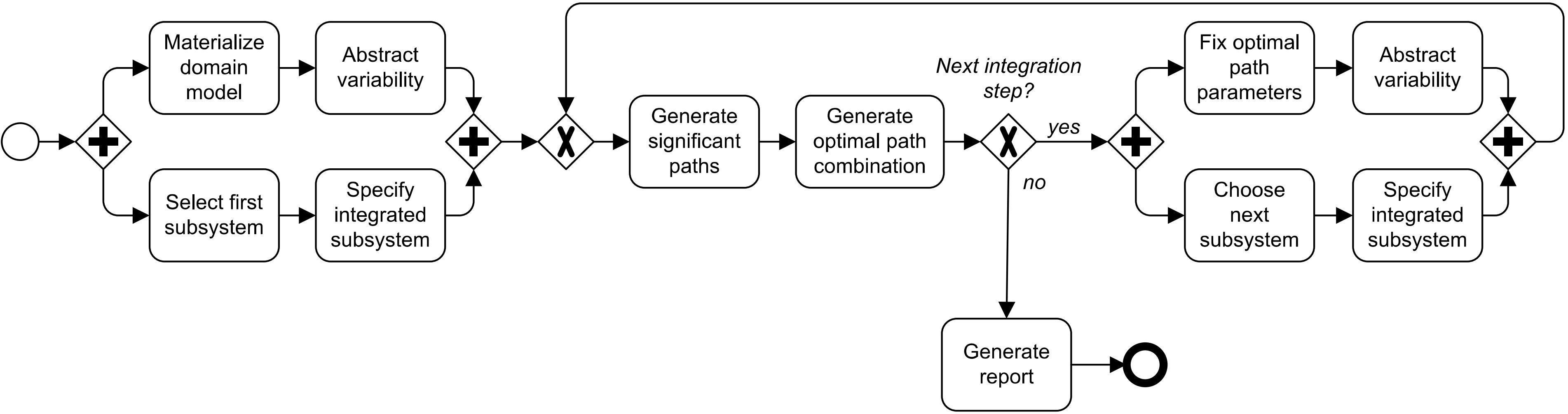


Guidelines: Naming Conventions

1. Give a name to every event and task
2. For tasks: verb followed by business object name and possibly complement
   - Issue Driver Licence, Renew Licence via Agency
3. For message events: object + past participle
   - Invoice received, Claim settled
4. Avoid generic verbs such as Handle, Record…
5. Label each XOR-split with a condition
   - Policy is invalid, Claim is inadmissible

Poll: Which model do you prefer?





One more guideline…

- Model in blocks
  - Pair up each AND-split with an AND-join and each XOR-split with a XOR-join, whenever possible
  - Exception: sometimes a XOR-split leads to two end events – different outcomes (cf. order management example)

Rework and repetition

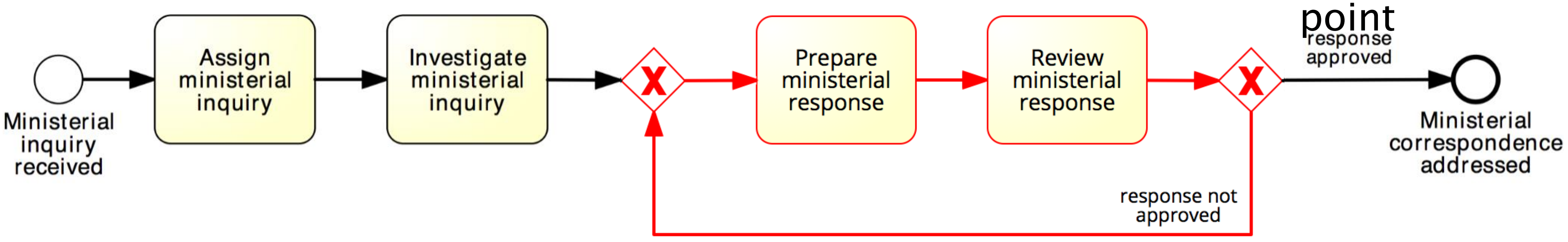
Address ministerial correspondence

In the minister’s office, when a ministerial inquiry has been received, it is registered into the system. Then the inquiry is investigated so that a ministerial response can be prepared.

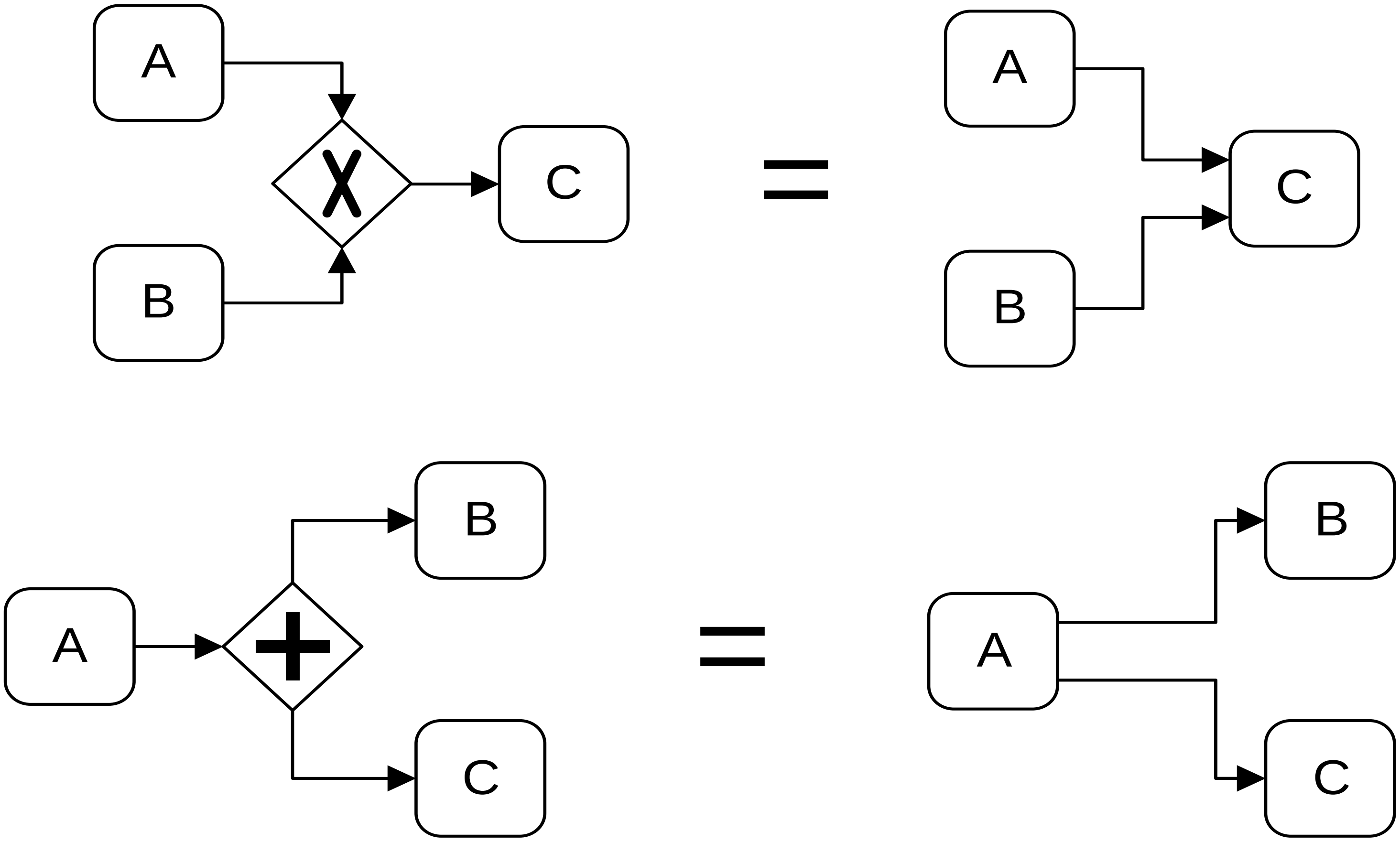
The finalization of a response includes the preparation of the response itself by the cabinet officer and the review of the response by the principal registrar. If the registrar does not approve the response, the latter needs to be prepared again by the cabinet officer for review. The process finishes only once the response has been approved.

XOR-join: entry point

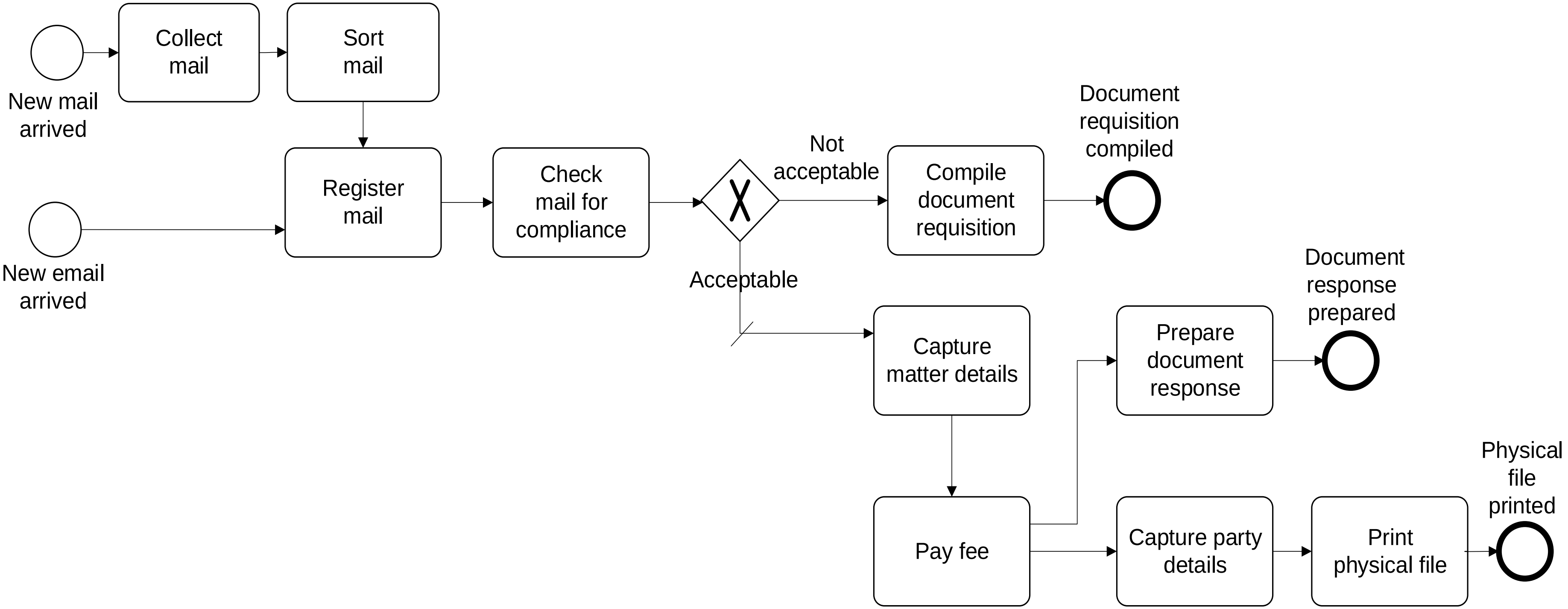
XOR-split: exit point


Ministerial inquiry received
Assign ministerial inquiry
Investigate ministerial inquiry
X
Prepare ministerial response
Review ministerial response
X
response approved
response not approved
Ministerial correspondence addressed

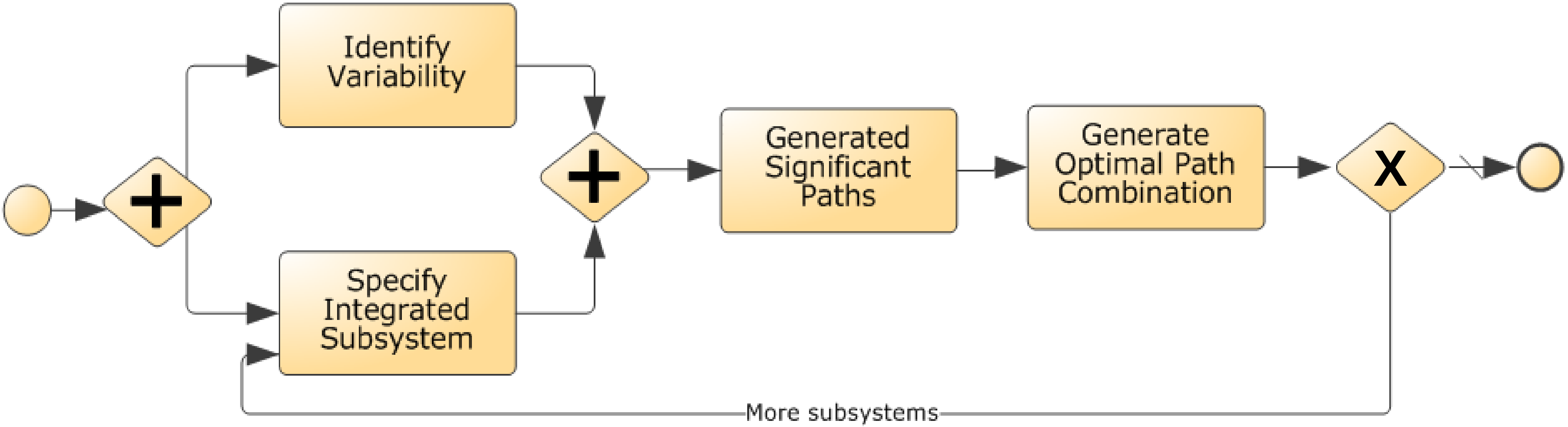
Quick Note: Implicit vs. explicit gateways



How this process starts? How it ends?



What’s wrong with this model? How to fix it?

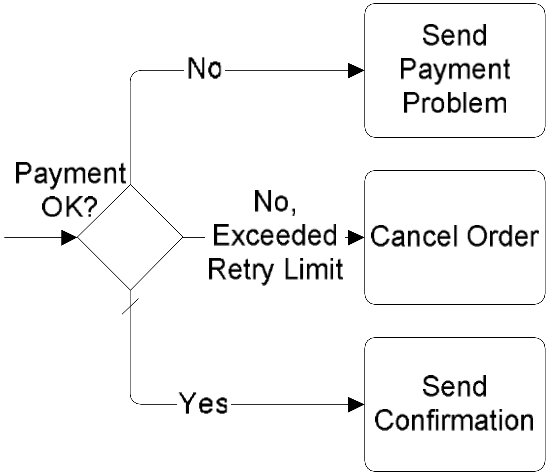

Identify Variability
Specify Integrated Subsystem
Generated Significant Paths
Generate Optimal Path Combination
X
More subsystems

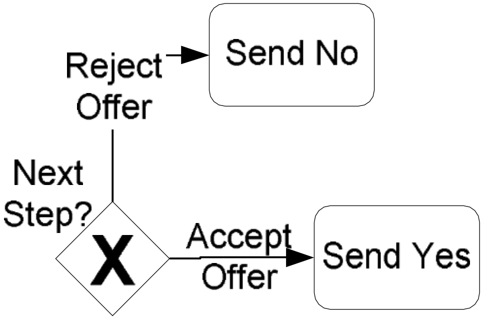
More on Exclusive Gateway

- The flow, after split, takes only one flow (aka XOR)
  - Based on data
  - Based on event

Exclusive, based on data

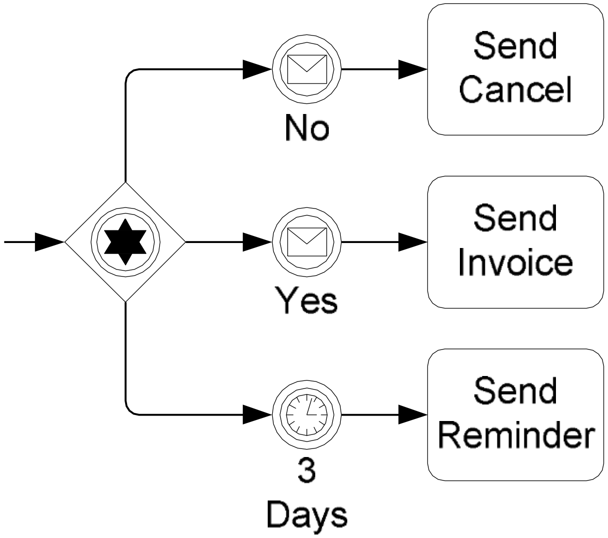
- Dash indicates default flow
- Diamond can also be drawn without X





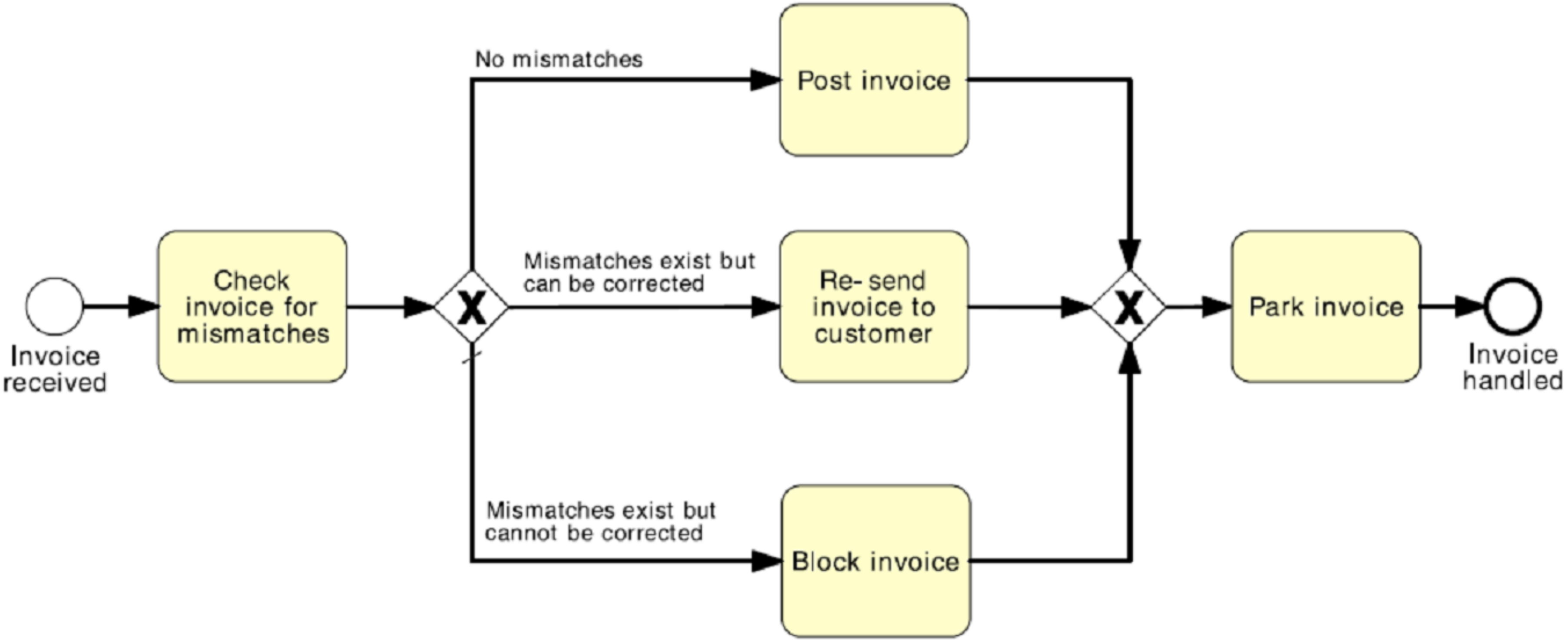
Exclusive, based on event

- First event that happens decides the next flow



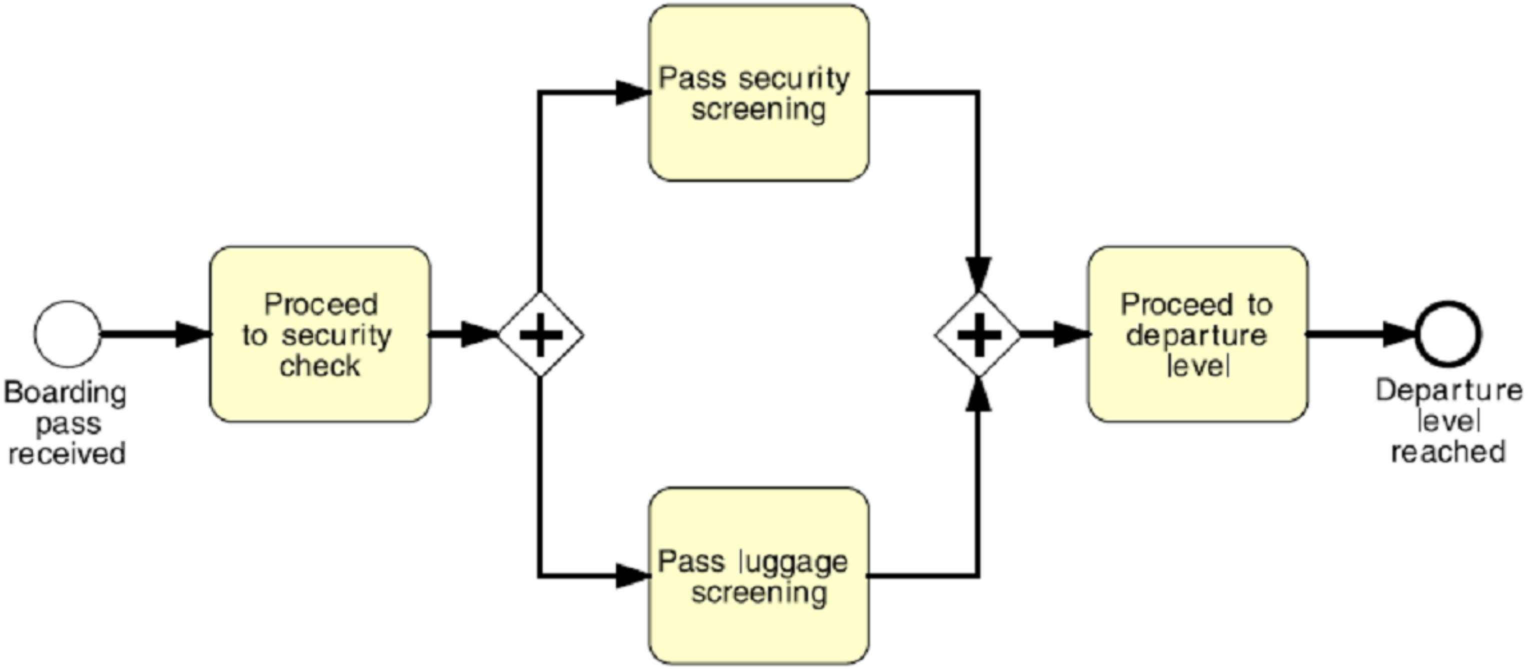
Exercice

- As soon as an invoice is received from a customer, it needs to be checked for mismatches.
- The check may result in either of these three options:
  - there are no mismatches, in which case the invoice is posted;
  - there are mismatches but these can be corrected, in which case the invoice is re-sent to the customer;
  - there are mismatches but these cannot be corrected, in which case the invoice is blocked.
- Once one of these three activities is performed the invoice is parked and the process completes.


No mismatches
Post invoice
Invoice received
Check invoice for mismatches
Mismatches exist but can be corrected
Re- send invoice to customer
Park invoice
Invoice handled
Mismatches exist but cannot be corrected
Block invoice

Exercice

- Once the boarding pass has been received, passengers proceed to the security check. Here they need to pass the personal security screening and the luggage screening. Afterwards, they can proceed to the departure level.


Boarding pass received
Proceed to security check
Pass security screening
Pass luggage screening
Proceed to departure level
Departure level reached

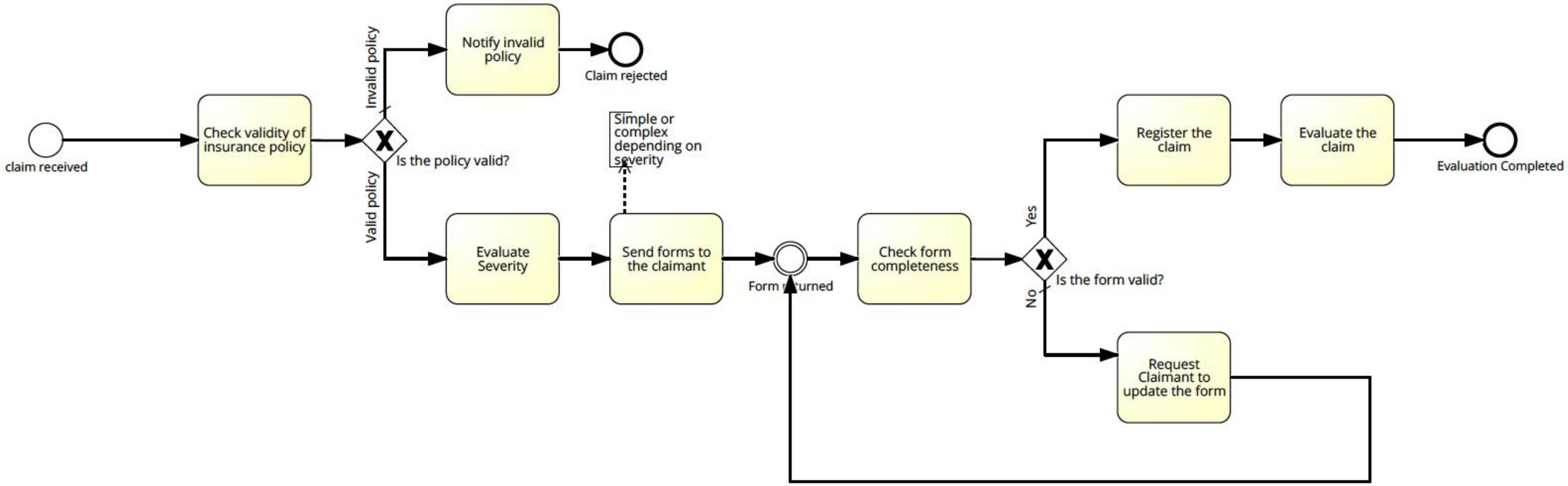
Exercise

When a claim is received, it is first checked whether the claimant has a valid insurance policy. If not, the claimant is informed that the claim is rejected due to an invalid policy.

Otherwise, the severity of the claim is evaluated. Based on the outcome (simple or complex claims), relevant forms are sent to the claimant. Once the forms are returned, they are checked for completeness.

If the forms are complete, the claim is registered in the Claims Management system and the evaluation of the claim may start. Otherwise, the claimant is asked to update the forms. Upon reception of the updated forms, they are checked again.

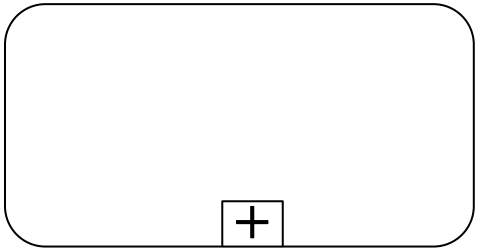
Solution


claim received
Check validity of insurance policy
Invalid policy
Is the policy valid?
Valid policy
Notify invalid policy
Claim rejected
Evaluate Severity
Send forms to the claimant
Simple or complex depending on severity
Form returned
Check form completeness
Is the form valid?
Yes
No
Register the claim
Evaluate the claim
Evaluation Completed
Request Claimant to update the form

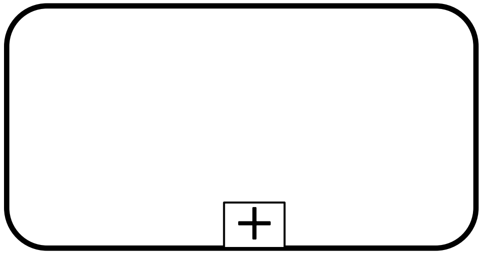
Sub processes

Sub process

- An activity can invoke a sub process. Should be used to
  - Decompose large models
  - Share subprocesses

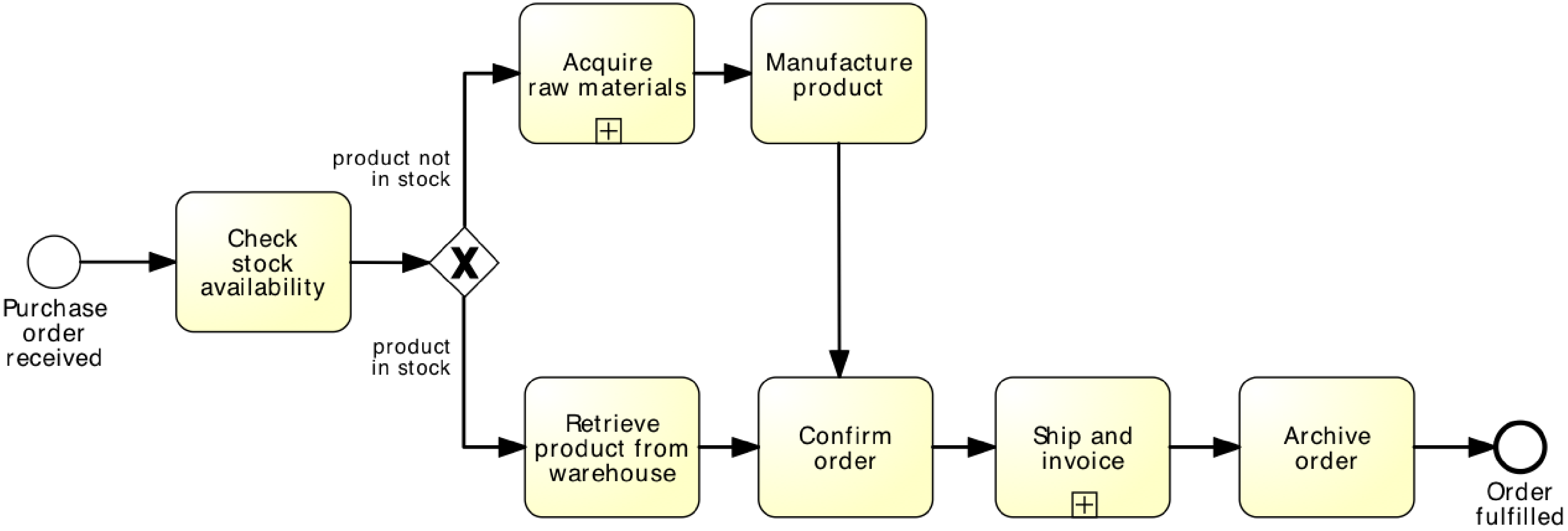


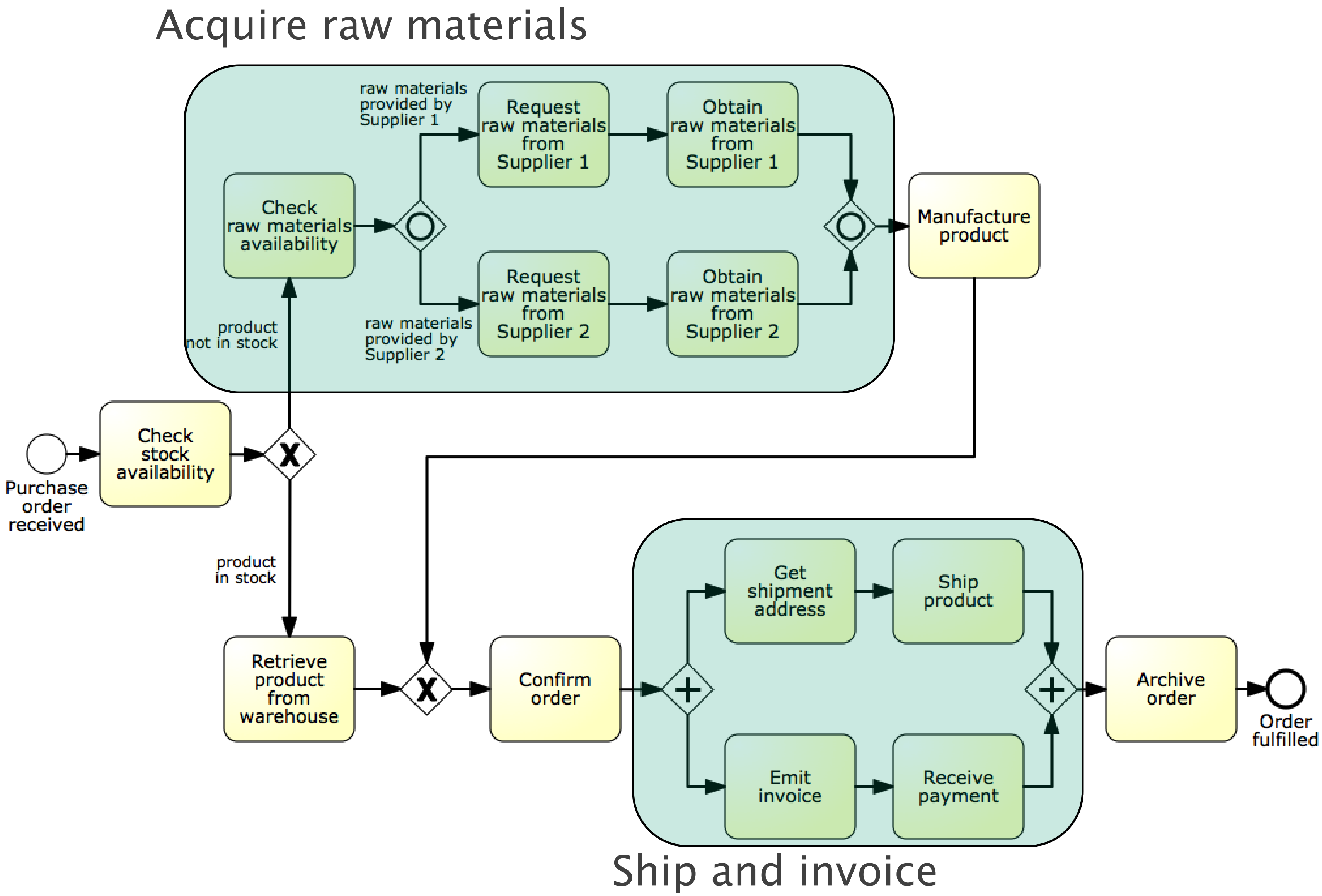
decomposition



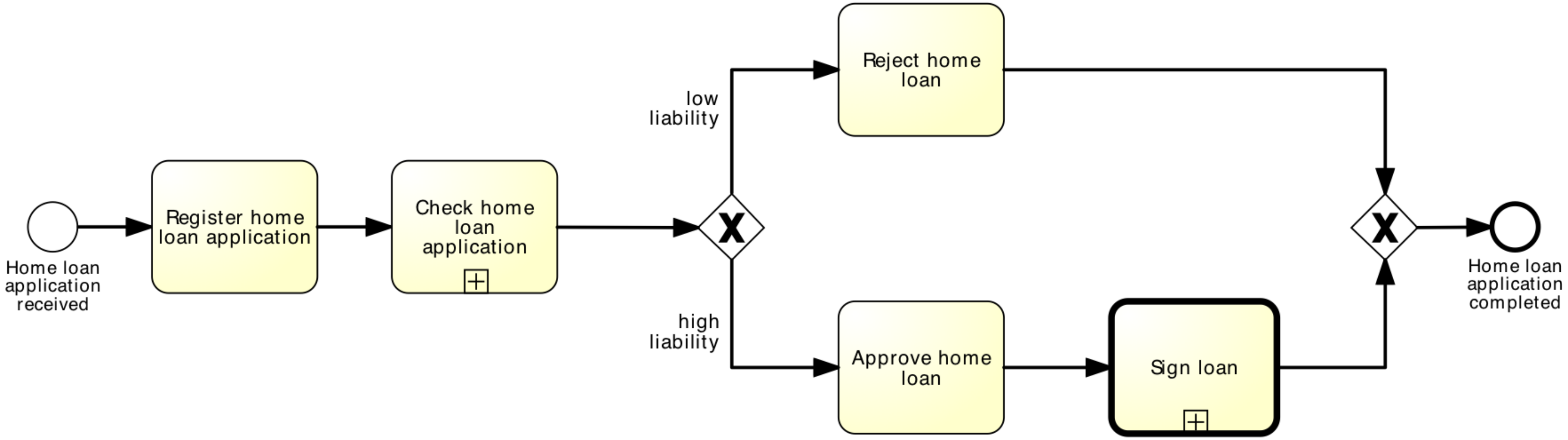
shared process (call activity)
(note bold border)

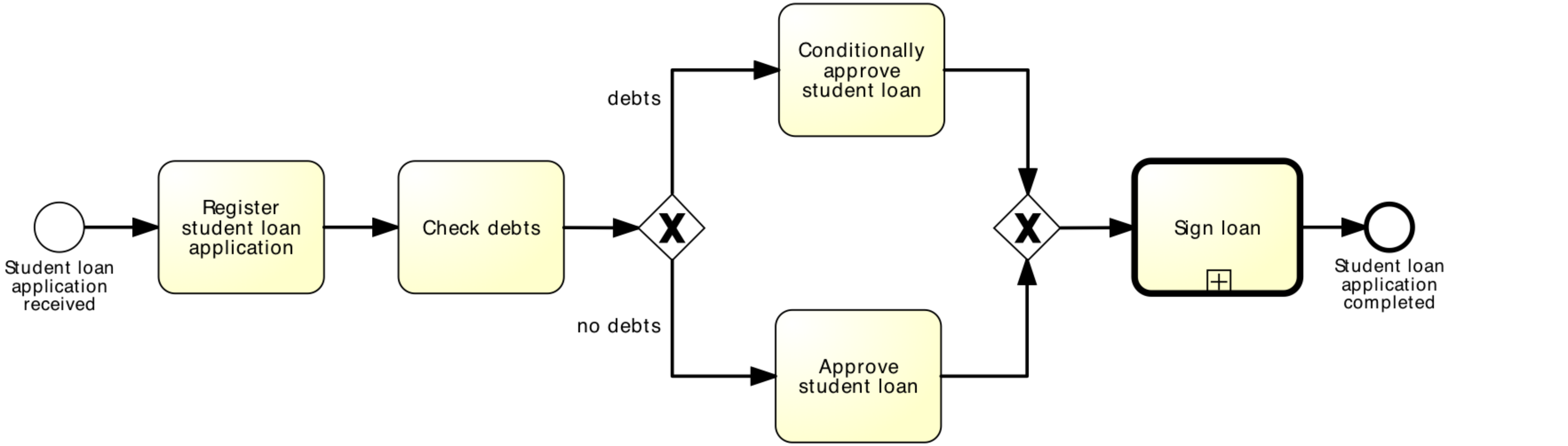
Ex: organize model




Acquire raw materials
raw materials provided by Supplier 1
Request raw materials from Supplier 1
Obtain raw materials from Supplier 1
Check raw materials availability
Manufacture product
Request raw materials from Supplier 2
Obtain raw materials from Supplier 2
product not in stock
raw materials provided by Supplier 2
Check stock availability
Purchase order received
product in stock
Get shipment address
Ship product
Retrieve product from warehouse
Confirm order
Archive order
Order fulfilled
Emit invoice
Receive payment
Ship and invoice

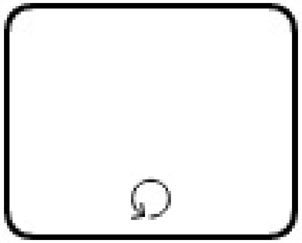
Ex: shared subprocesses


Home loan application received
Register home loan application
Check home loan application
low liability
high liability
Reject home loan
Approve home loan
Sign loan
Home loan application completed

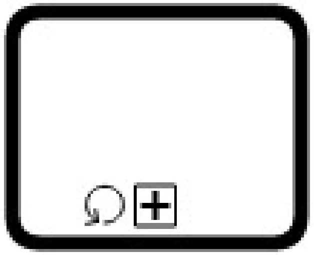

Student loan application received
Register student loan application
Check debts
debts
no debts
Conditionally approve student loan
Approve student loan
Sign loan
Student loan application completed

Repetitions

- Either a task, or a subprocess, can be repeated



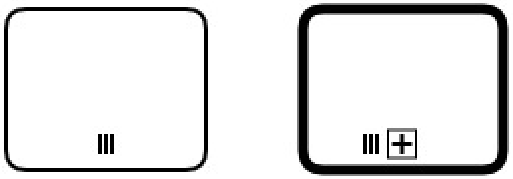
repeated task



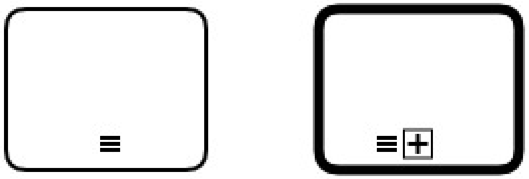
repeated sub process

- Futher, it is possible to state if the repetition is made in sequence or in parallel

In parallel:

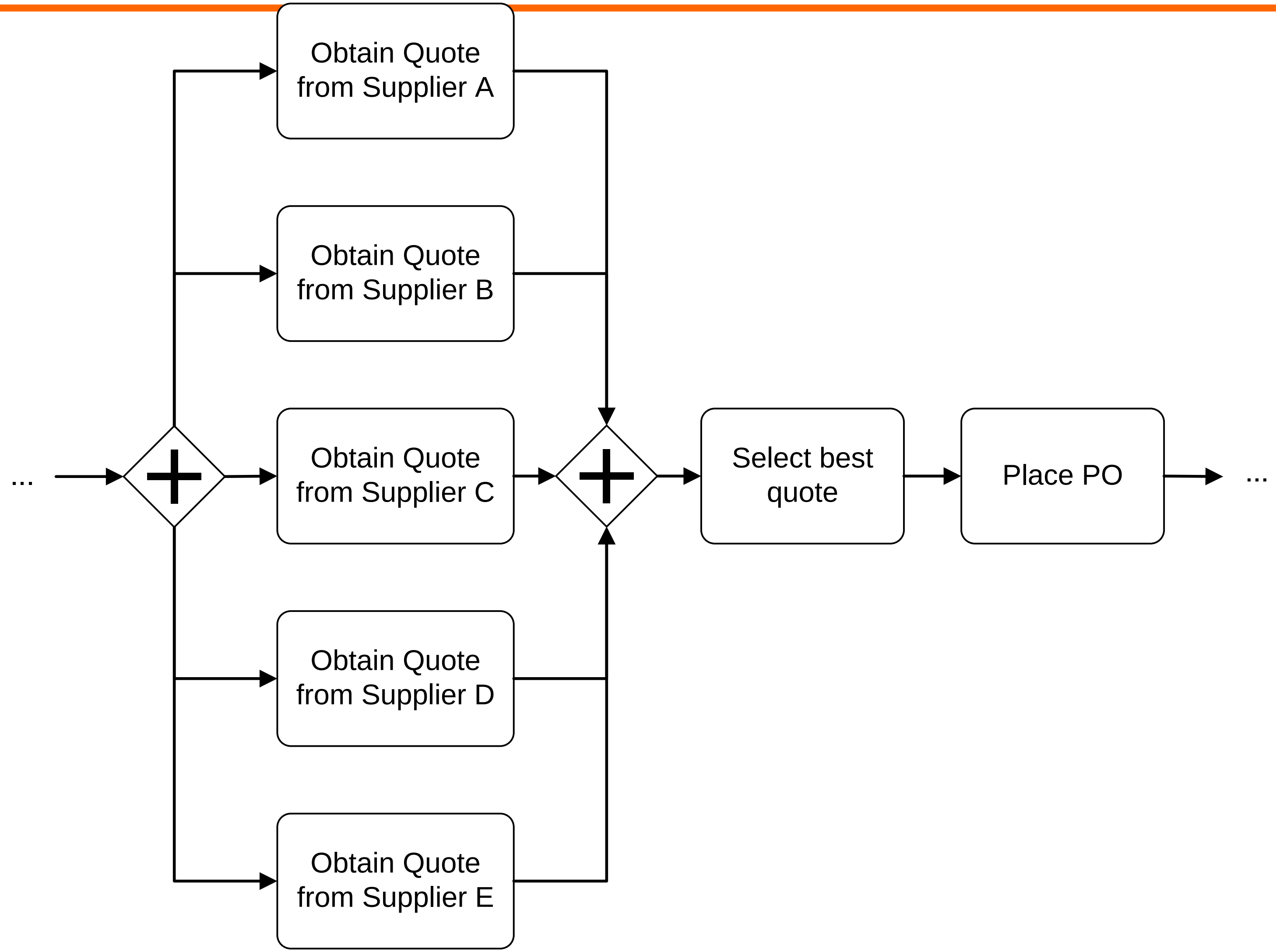


In sequence:

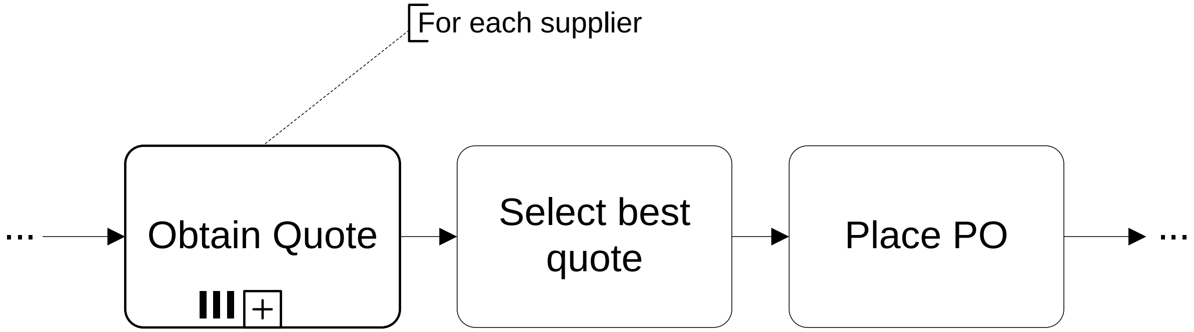


Ex.:

In procurement, typically a quote is to be obtained from all preferred suppliers (assumption: five preferred suppliers exist). After all quotes are received, they are evaluated and the best quote is selected. A corresponding purchase order is then placed.


...
Obtain Quote from Supplier A
Obtain Quote from Supplier B
Obtain Quote from Supplier C
Obtain Quote from Supplier D
Obtain Quote from Supplier E
Select best quote
Place PO
...

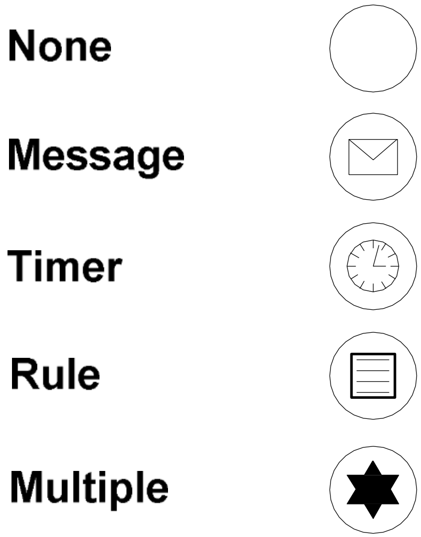
With parallel subtask



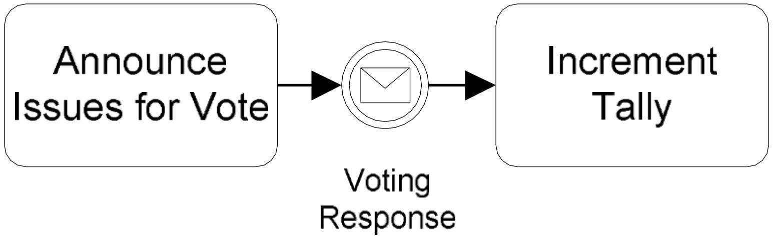
Events – advanced

Start events

- Have an associated trigger (condition that describes when event happens)
  - None has no trigger (for subprocesses)
  - Multiple has more than one



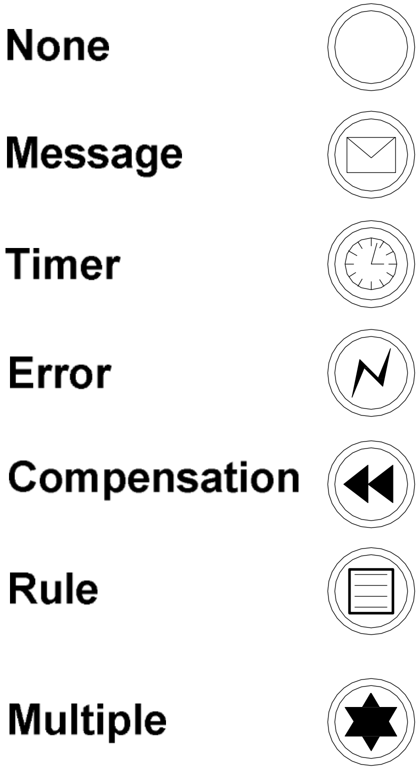
Intermediate events



Announce Issues for Vote

Voting Response

Increment Tally



None

Message

Timer

Error

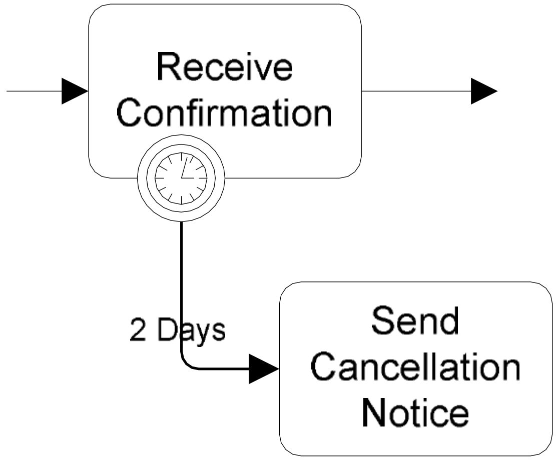
Compensation

Rule

Multiple

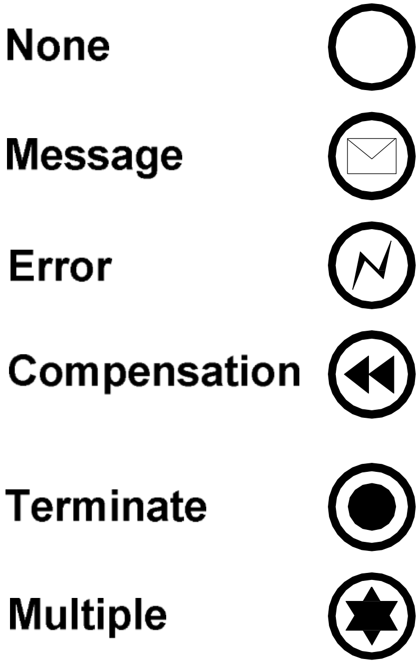
Intermediate events

- If attached to boundary of activity interrupt the activity



End events

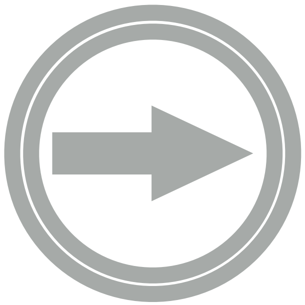
- Indicate end of process
- None for subprocesses

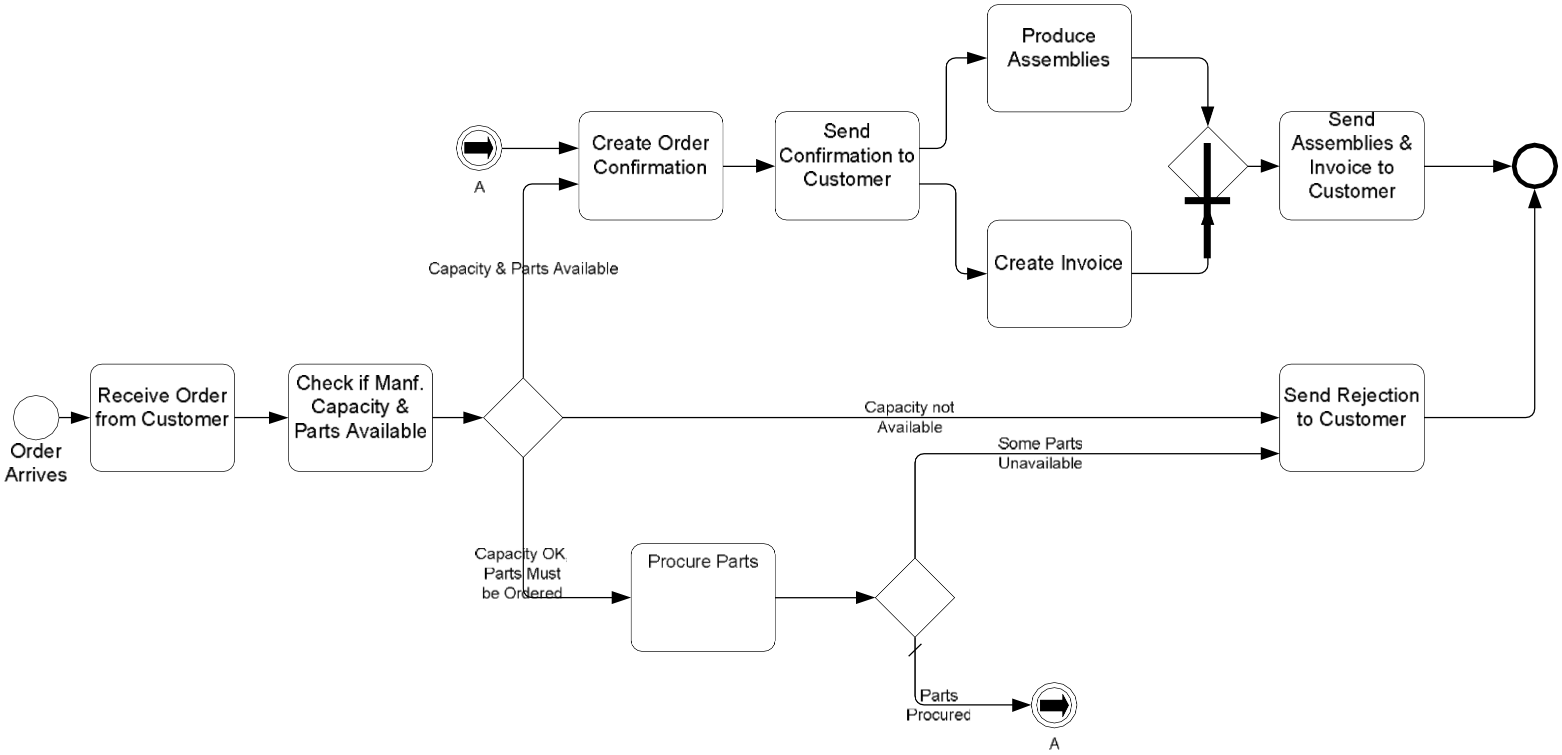


Link events

Used to cut a process in parts

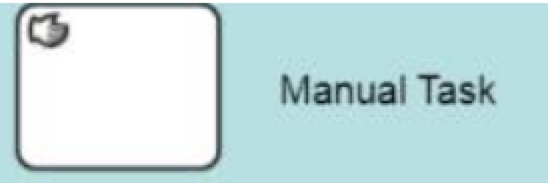
Deprecated



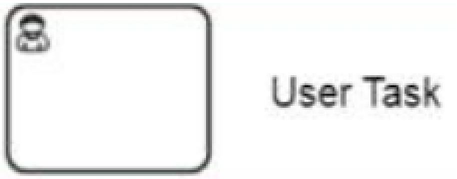

A
Create Order Confirmation
Send Confirmation to Customer
Produce Assemblies
Create Invoice
Send Assemblies & Invoice to Customer
Capacity & Parts Available
Order Arrives
Receive Order from Customer
Check if Manf. Capacity & Parts Available
Capacity not Available
Send Rejection to Customer
Some Parts Unavailable
Capacity OK, Parts Must be Ordered
Procure Parts
Parts Procured
A

Tasks, advanced

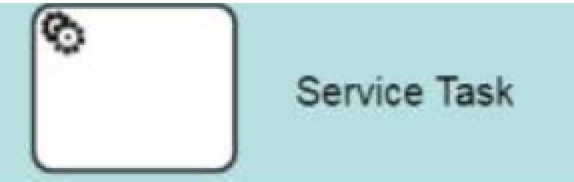
User, manual, service



- Manual task: executed by person, with no software tool

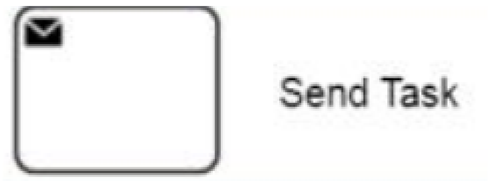


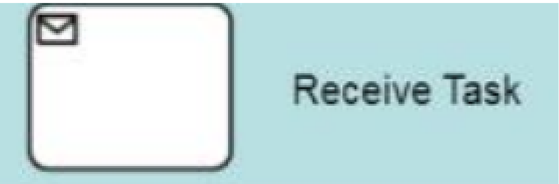
- User task: executed by person, with software tool



- Service task: completely automated

Message task





- Message send / receive activities that can be interrupted
- (if not interruptable use message events)

BPMN, Organizational perspective

Elements

Resource

- Human actor or equipment (e.g. printer) that is required to perform an activity

Resource class: Set of resources with shared characteristics, e.g. Clerk, Manager, Insurance Officer

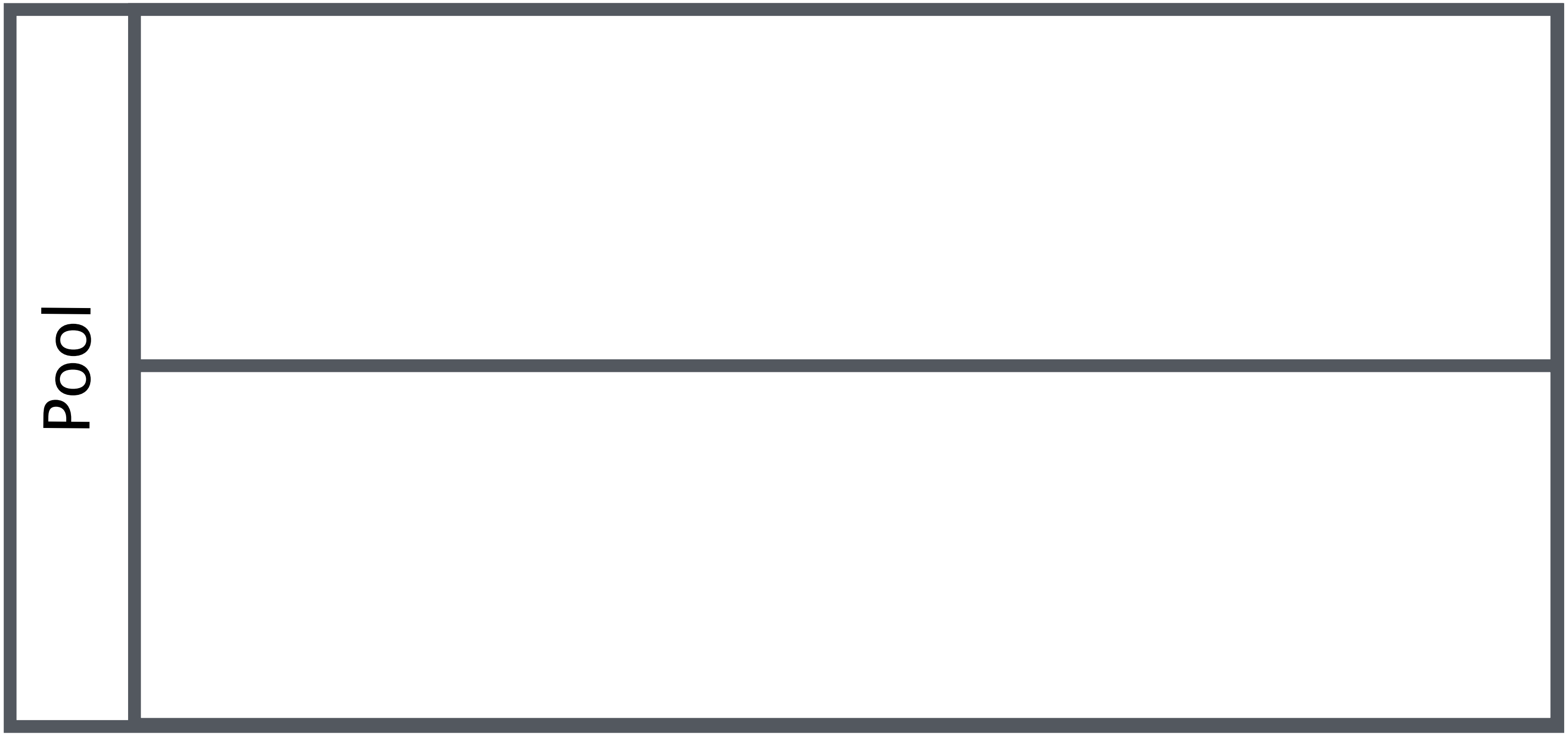
- **Role** (skill, competence, qualification)
  This classification is based on what a resource can do or is expected to do.
- **Group** (department, team, office, organizational unit). This classification is based on the organization’s structure.

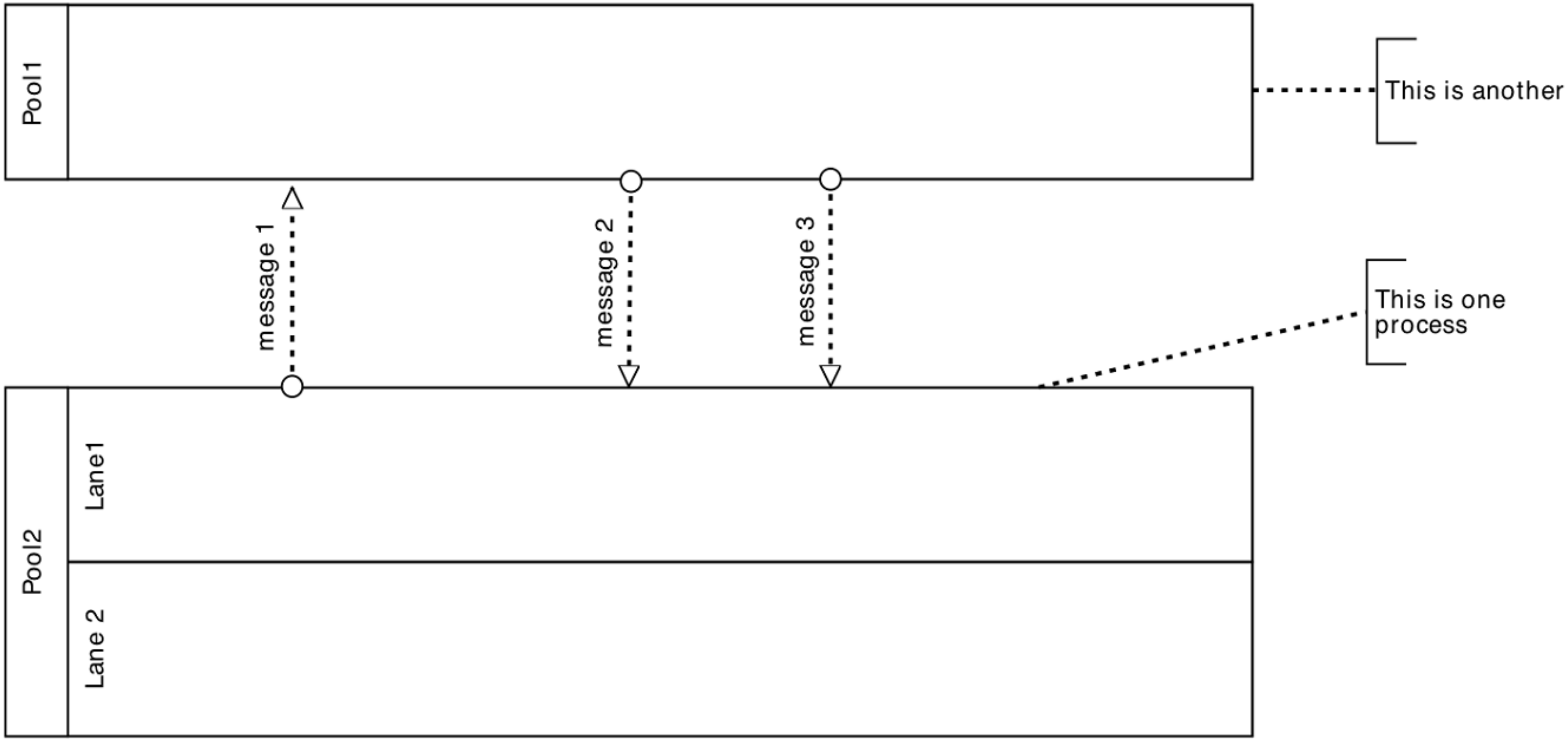
Organisational Elements

- Pool;
- Swimlane;
- Group.

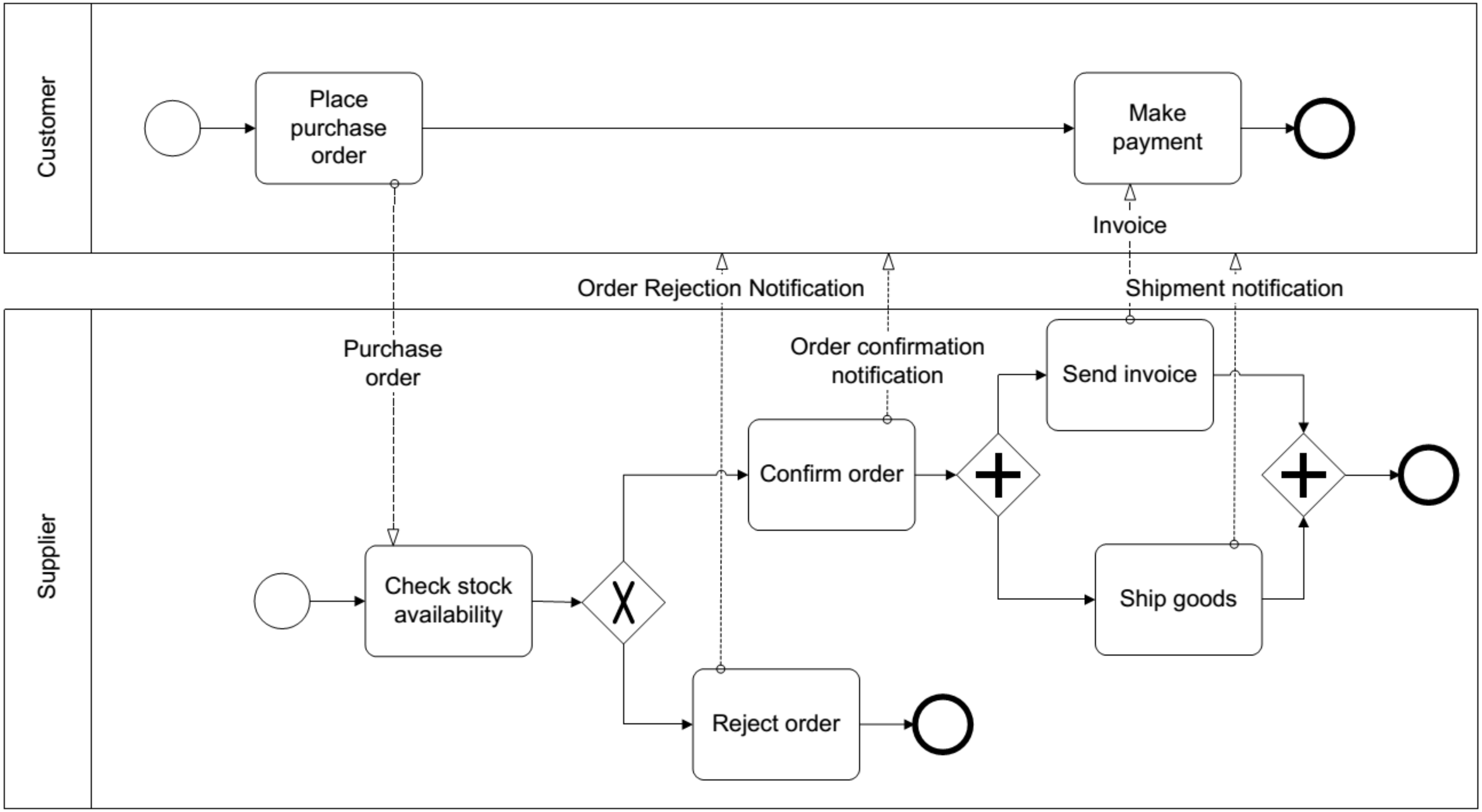
Pool

- Contains a single, complete process.
- Flow cannot cross pool
- Processes in different pools communicate via messages




Pool1
This is another
message 1
message 2
message 3
This is one process
Pool2
Lane1
Lane 2

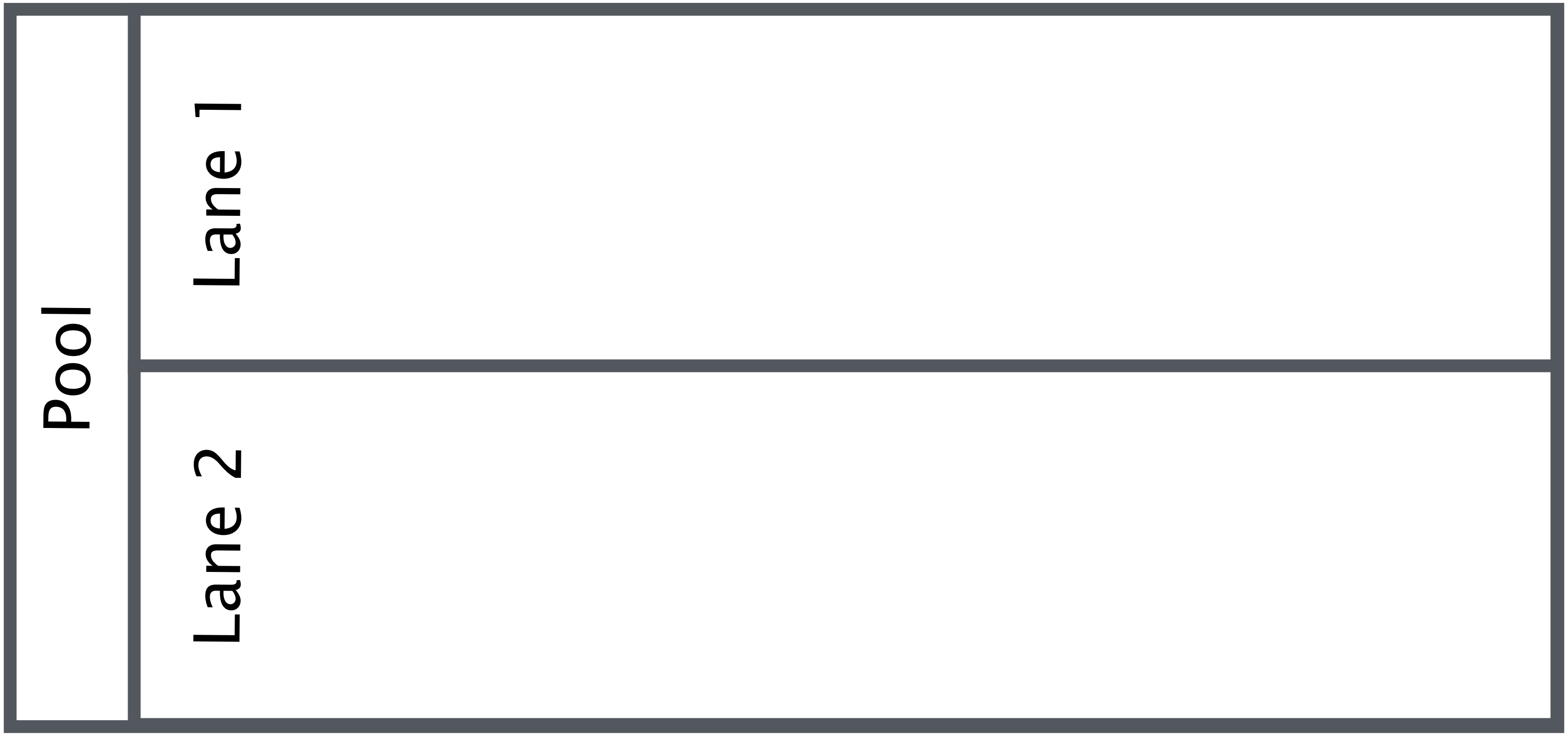
Order management w pools



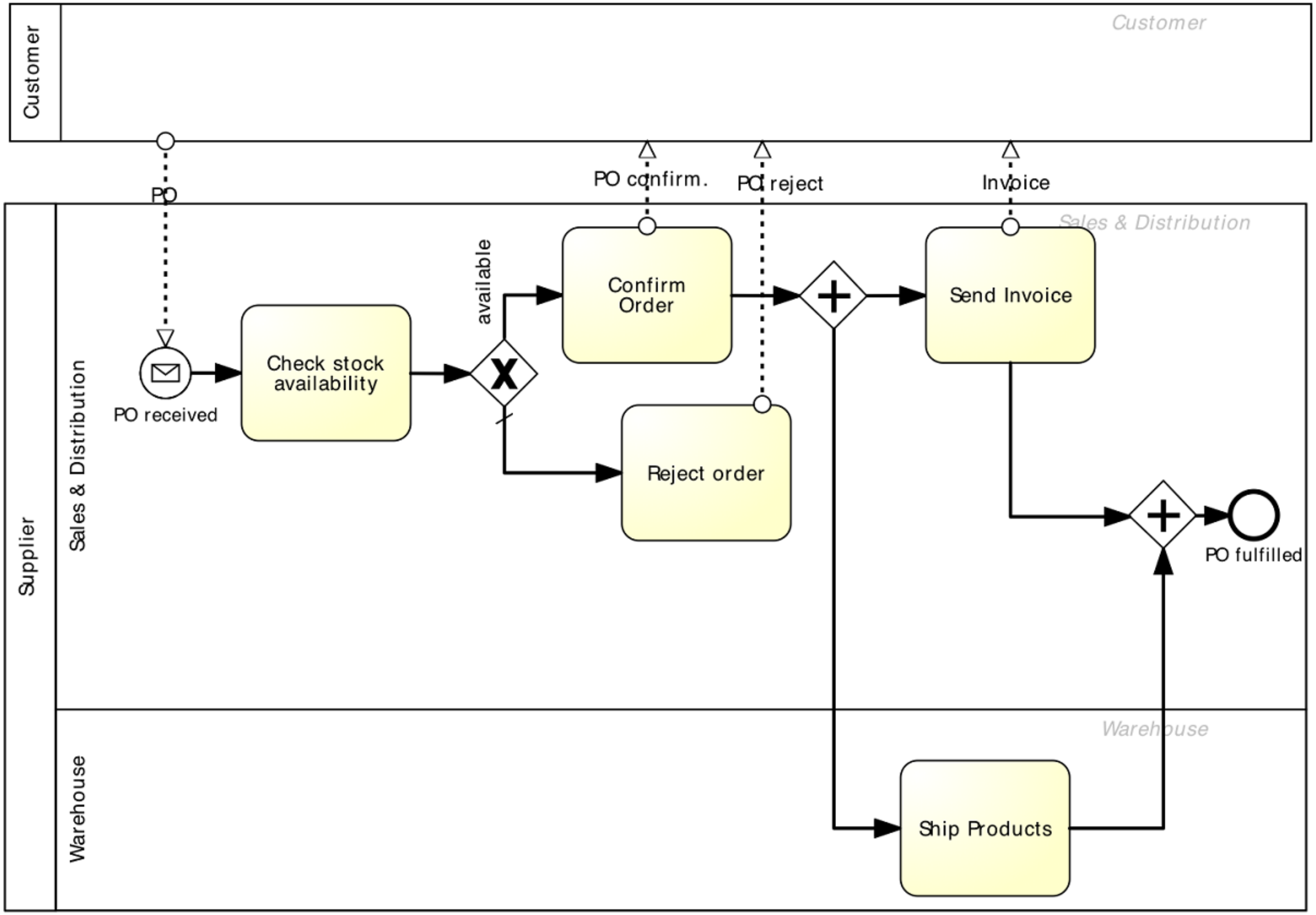
Lane

Inside a process (Pool) shows who does what.

Flow can cross lanes



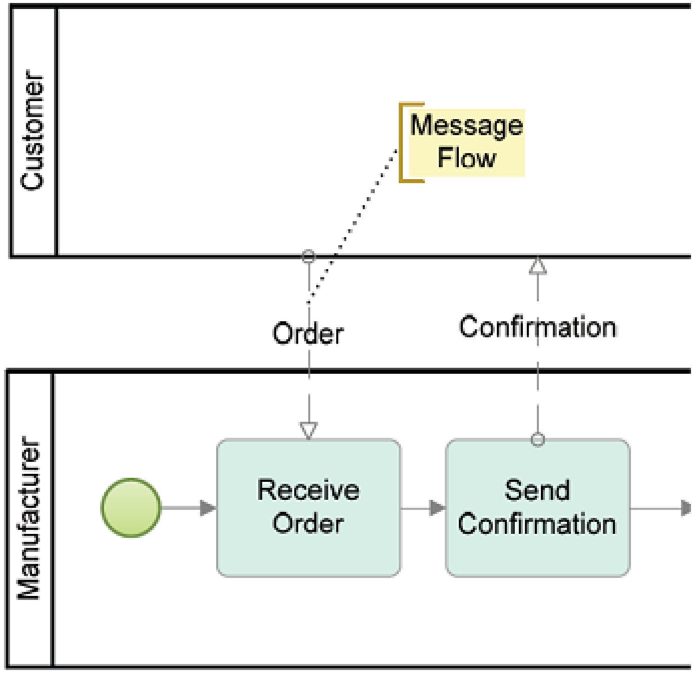
Order management w lanes



Pools and lanes

- Pool: independent organisational entity
    - Ex: customer == pool
      supplier == pool
- Two pools do not share a common system to communicate
    - Must use messages /events to communicate
- Lane
    - Units of same organisational entity
    - Share a common system to communicate

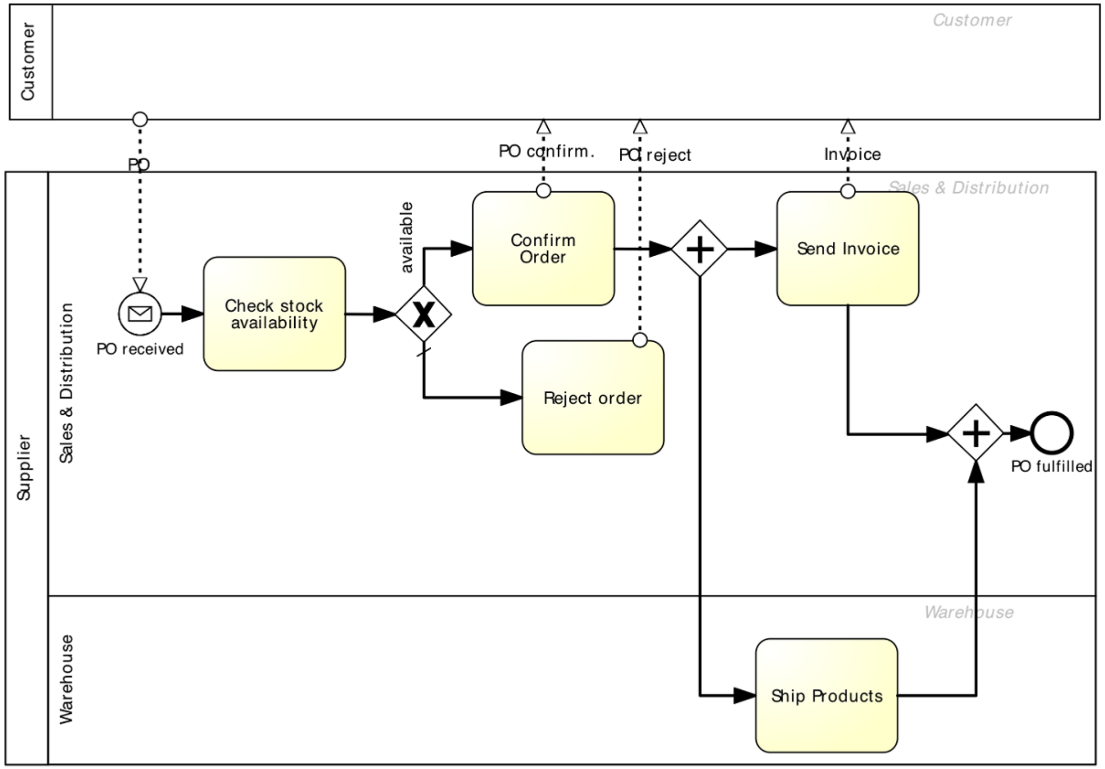
Message flow



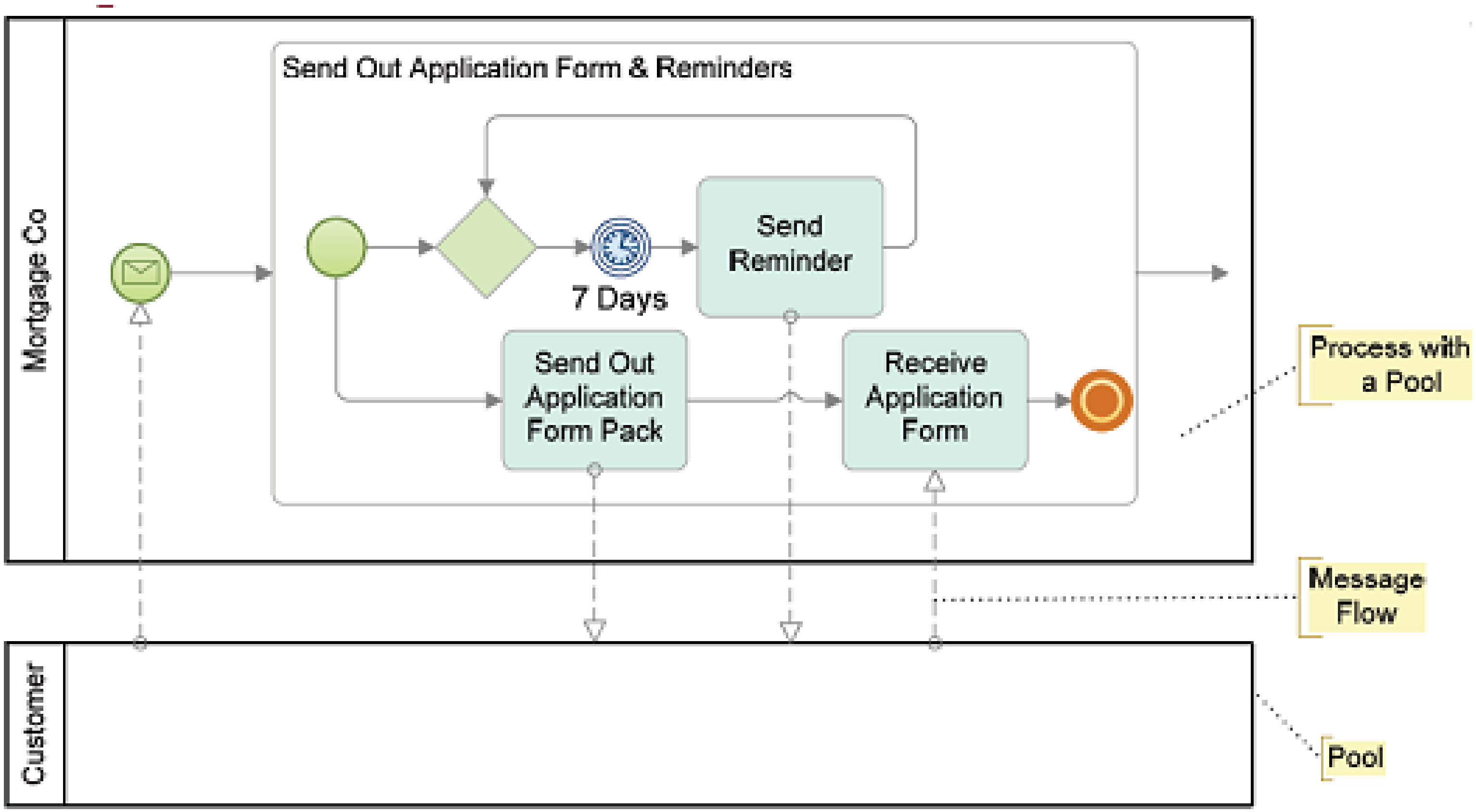
- Represents asynchronous communication between two pools (two processes)
  - Cannot happen within a pool / process

Black box pool

- Process in the pool is undefined
- Message flows connect to/from pool boundary

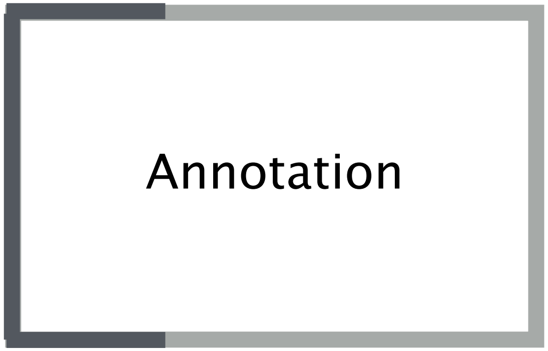


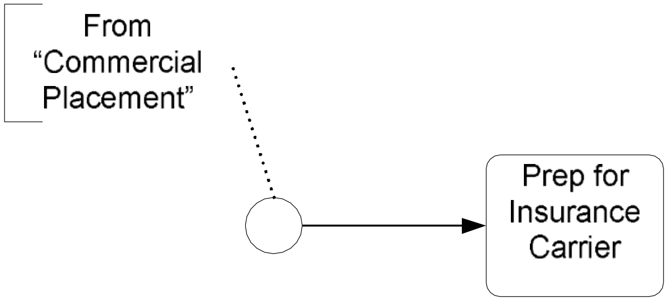
Black box pool



Text Annotation

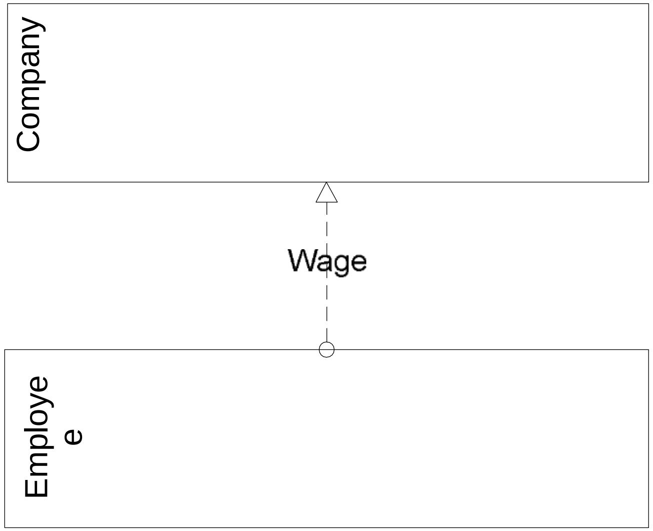
To attach notes to a model entity with explanations for clarity.



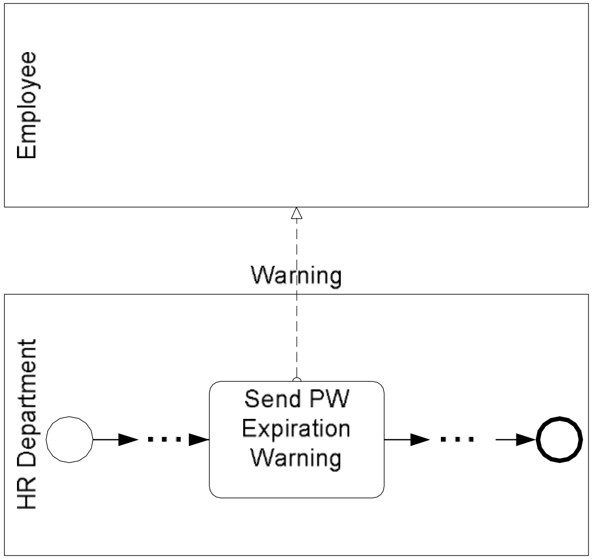


Message flow

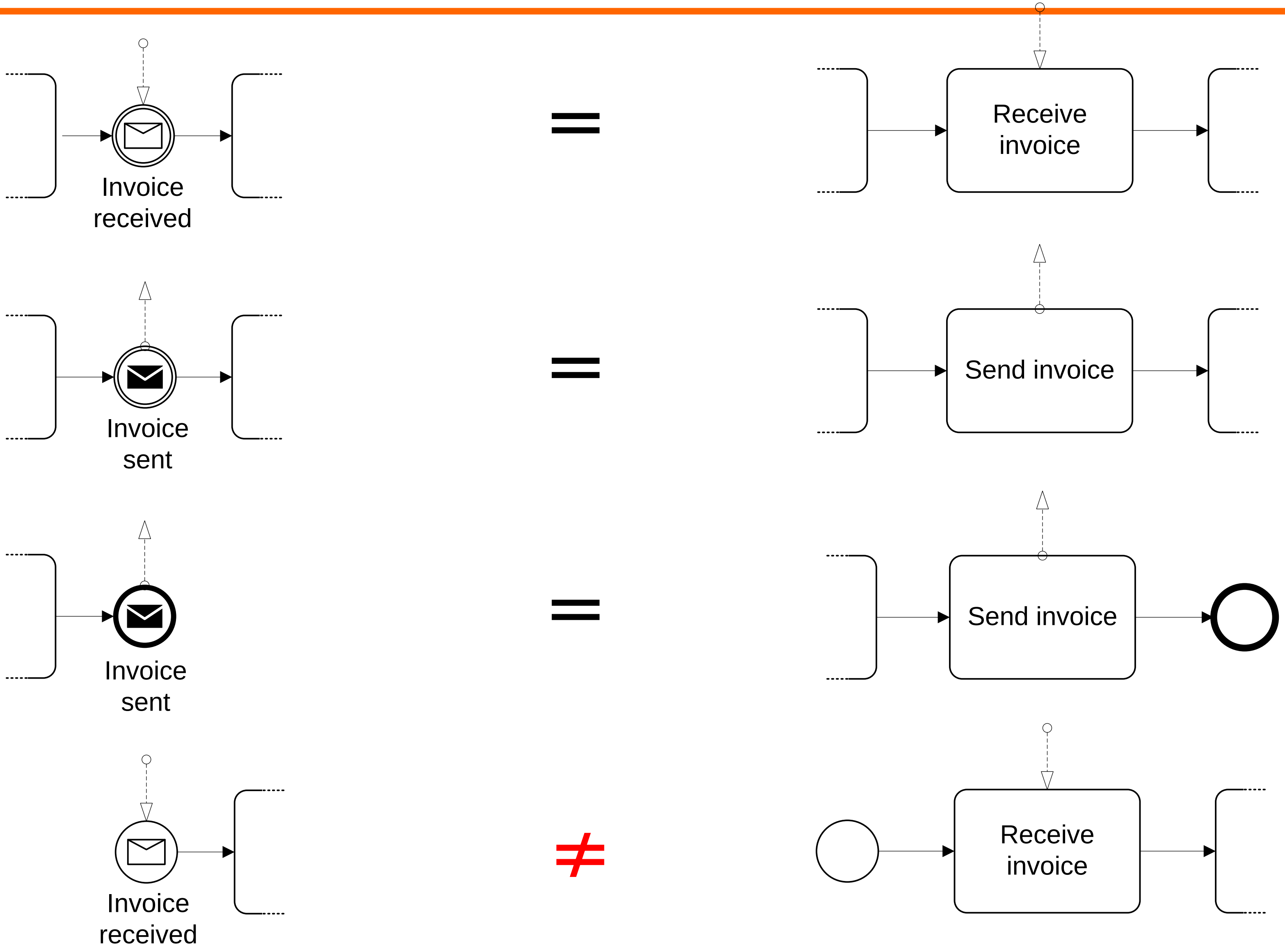
- Shows flow of message
    - Between pools



    - Between activities in different pools



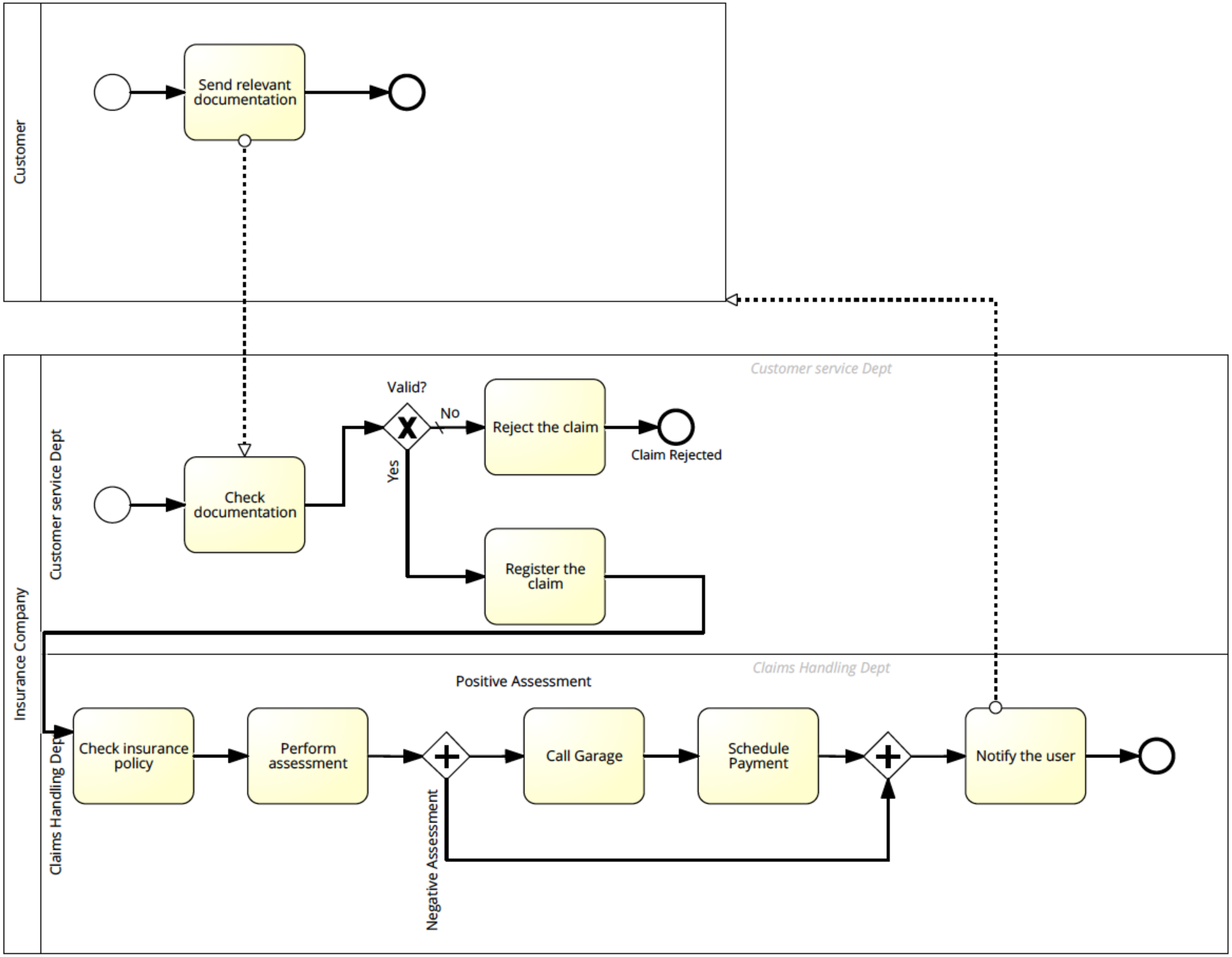
Equivalences



Exercise: Lanes, Pools

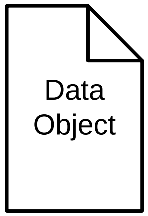
- Claims Handling process at a car insurer
- A customer submits a claim by sending in relevant documentation. The Customer Service department checks the documents for completeness and registers the claim. The Claims Handling department picks up the claim and first checks the insurance policy. Then, an assessment is performed. If the assessment is positive, a garage is phoned to authorise the repairs and the payment is scheduled (in this order). In any case (whether the outcome is positive or negative), an e-mail is sent to the customer to notify the outcome

Solution


Customer
Send relevant documentation
Insurance Company
Customer service Dept
Customer service Dept
Check documentation
Valid?
No
Yes
Reject the claim
Claim Rejected
Register the claim
Claims Handling Dept
Claims Handling Dept
Positive Assessment
Check insurance policy
Perform assessment
Negative Assessment
Call Garage
Schedule Payment
Notify the user

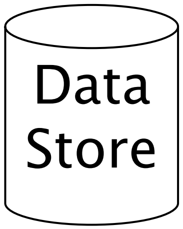
Data perspective

Information Artifacts

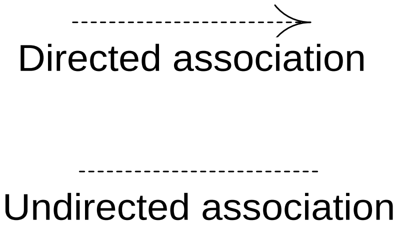


Data Objects show volatile data required or produced by activities.

They represent input and output of a process activity.

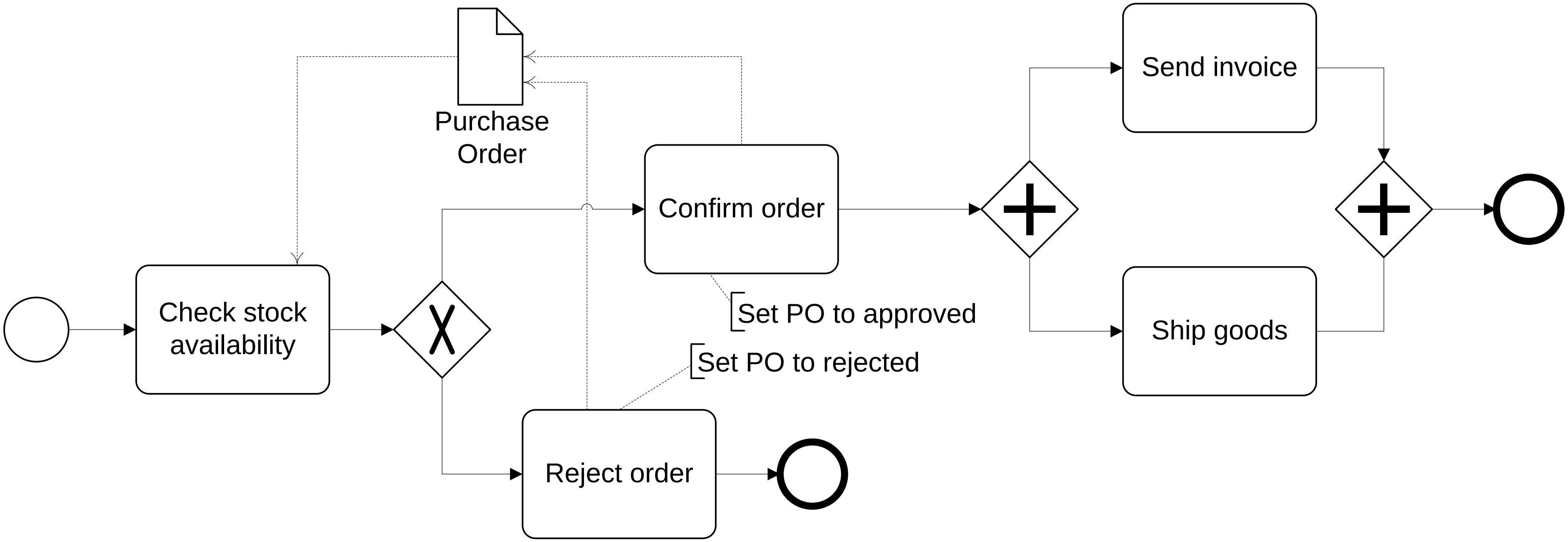


Data stores are containers of data objects that need be persisted beyond the duration of a process instance



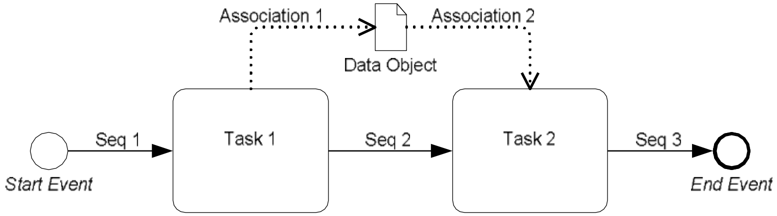
Associations are used to link artifacts such as data objects and data stores with flow objects (e.g. activities).

Order Processing Model with Artifacts

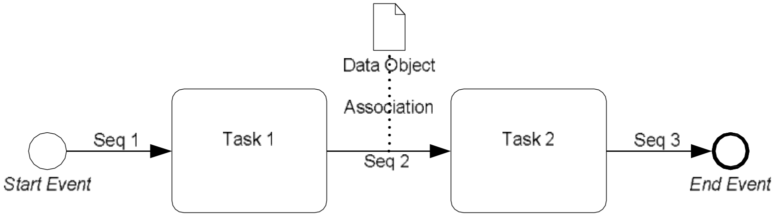


Artifacts – 2

- Data flow decoupled from sequence flow

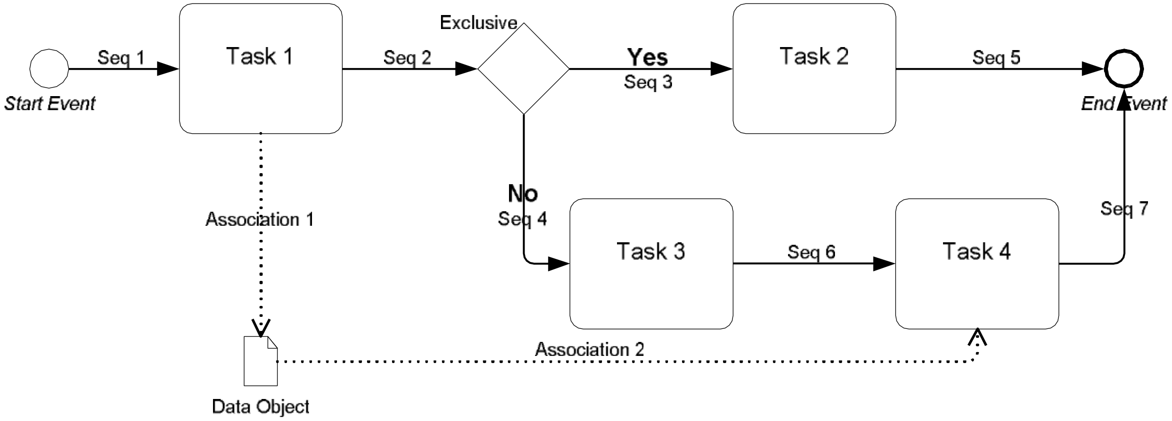


- coupled



Artifacts 3

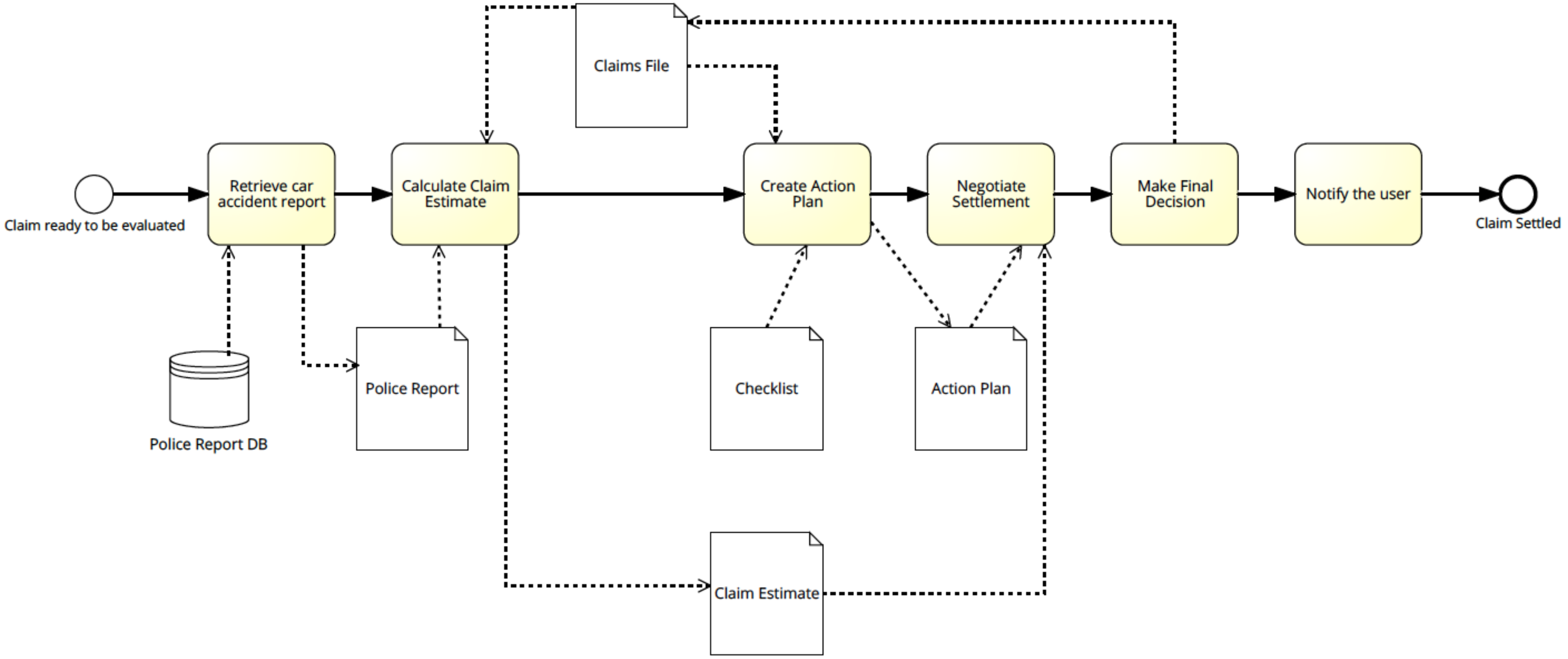
- decoupled



Exercise: Artifacts

When a claim related to a major car accident is evaluated, a clerk first retrieves the corresponding car accident report in the Police Reports database. If the report is retrieved, it is attached to the claim file. The claim file and the police report serve as input to a claims handler who calculates an initial claim estimate. Then, an “action plan” is created based on a “checklist”. Based on the action plan and the initial claims estimate, a claims manager negotiates a settlement with the customer. After this negotiation, the claims manager makes a final decision, updates the claim file to record this decision, and sends a letter to the claimant to inform him/her of the decision.

Solution

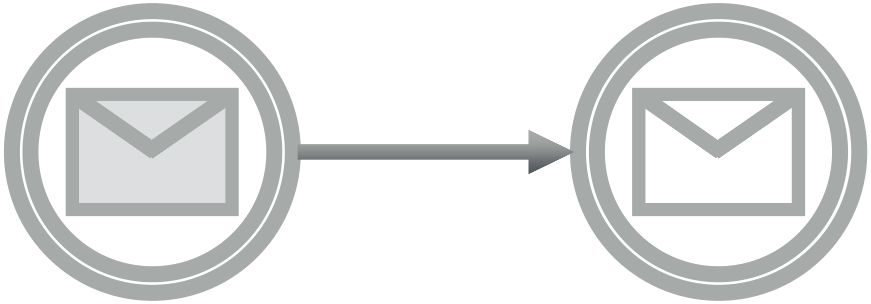

Claims File
Claim ready to be evaluated
Retrieve car accident report
Calculate Claim Estimate
Create Action Plan
Negotiate Settlement
Make Final Decision
Notify the user
Claim Settled
Police Report DB
Police Report
Checklist
Action Plan
Claim Estimate

Special Behaviour Elements

- **Message and Message Flows;**
- Signals;
- Correlation;
- Timers;
- Errors;

Messages and Message Flow

Used to transfer actions or data from one pool or process to another and to correlate related processes.

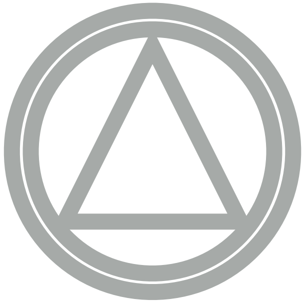


Throw Message   Catch Message

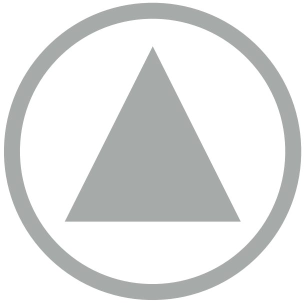
A message is a direct communication between two business participants. These participants must be in separate Pools (they cannot be sent from another Lane inside a single Pool)

Signals

Used to send data to multiple activities simultaneously.



Throw Signal

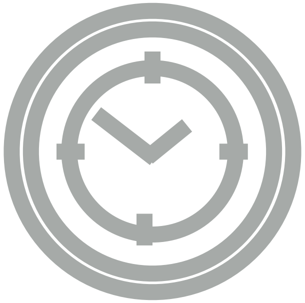


Catch Signal

Signals are broadcast communications from a business participant or another Process. Signals have no specific target or recipient – i.e. all Processes and participants can see the signal and it is up to each of them to decide whether or not to react.

Timers

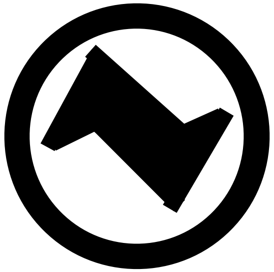
Used to launch periodic activities, or to ensure that an activity happens within a specified deadline



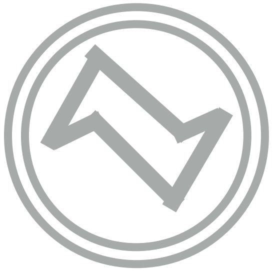
Timer

Error Events

- The Error Intermediate Event is used to handle the occurrence of an *error* that needs the interrupting of an Activity (to which it is attached).



Error end event

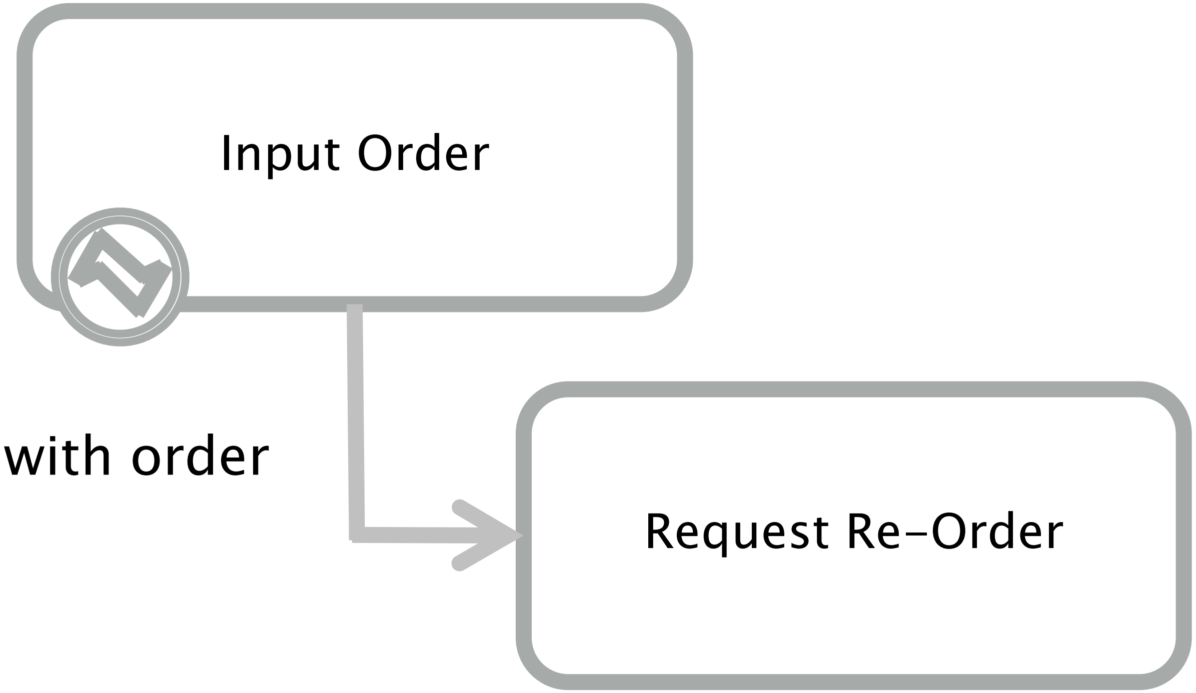


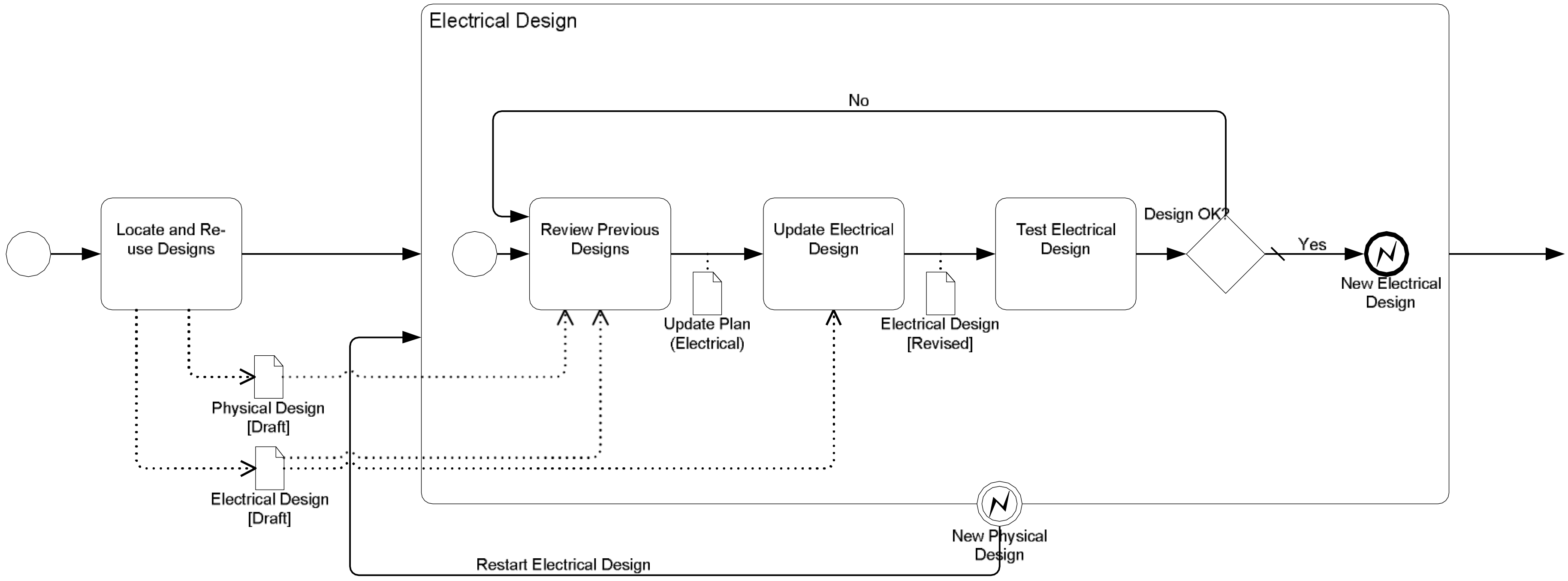
Error Intermediate Event

- The Error End Event is used to throw an error.
- The Error Intermediate Event can only be used when attached to the boundary of an Activity, thus it can only be used to catch an error.
- When an error occurs all work will stop for that Process.

Exception

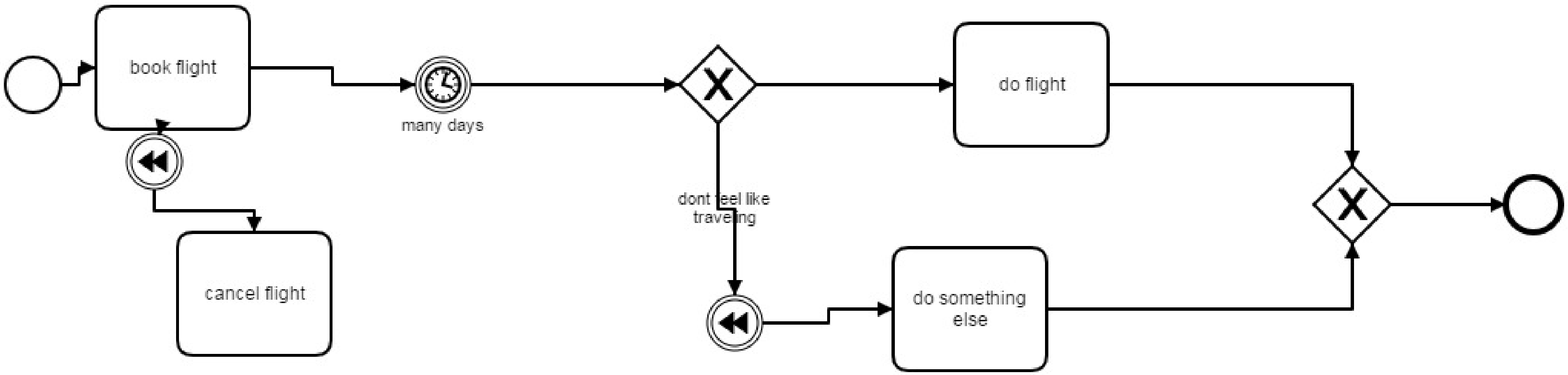
Used to define behaviour when the system encounters a technical error.

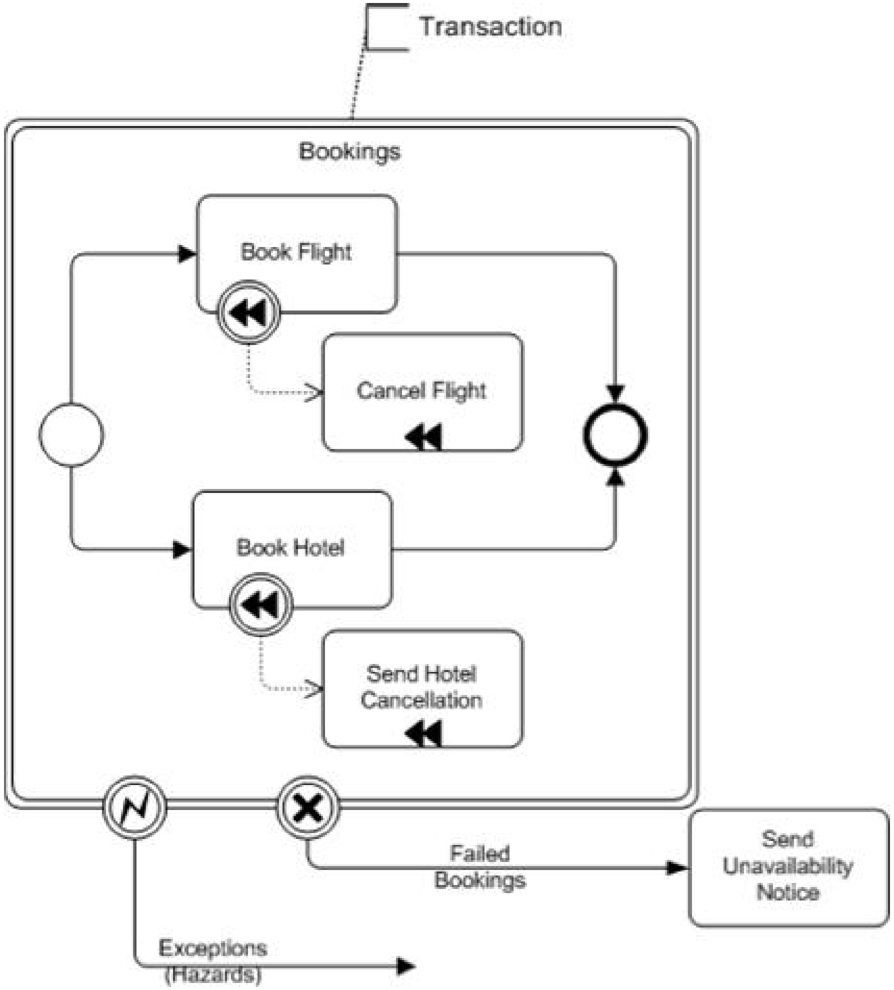



Electrical Design
No
Locate and Re-use Designs
Review Previous Designs
Update Electrical Design
Test Electrical Design
Design OK?
Yes
New Electrical Design
Update Plan (Electrical)
Electrical Design [Revised]
Physical Design [Draft]
Electrical Design [Draft]
New Physical Design
Restart Electrical Design

Compensation

- Event that cancels an activity already terminated in the past, triggering suitable compensation activities



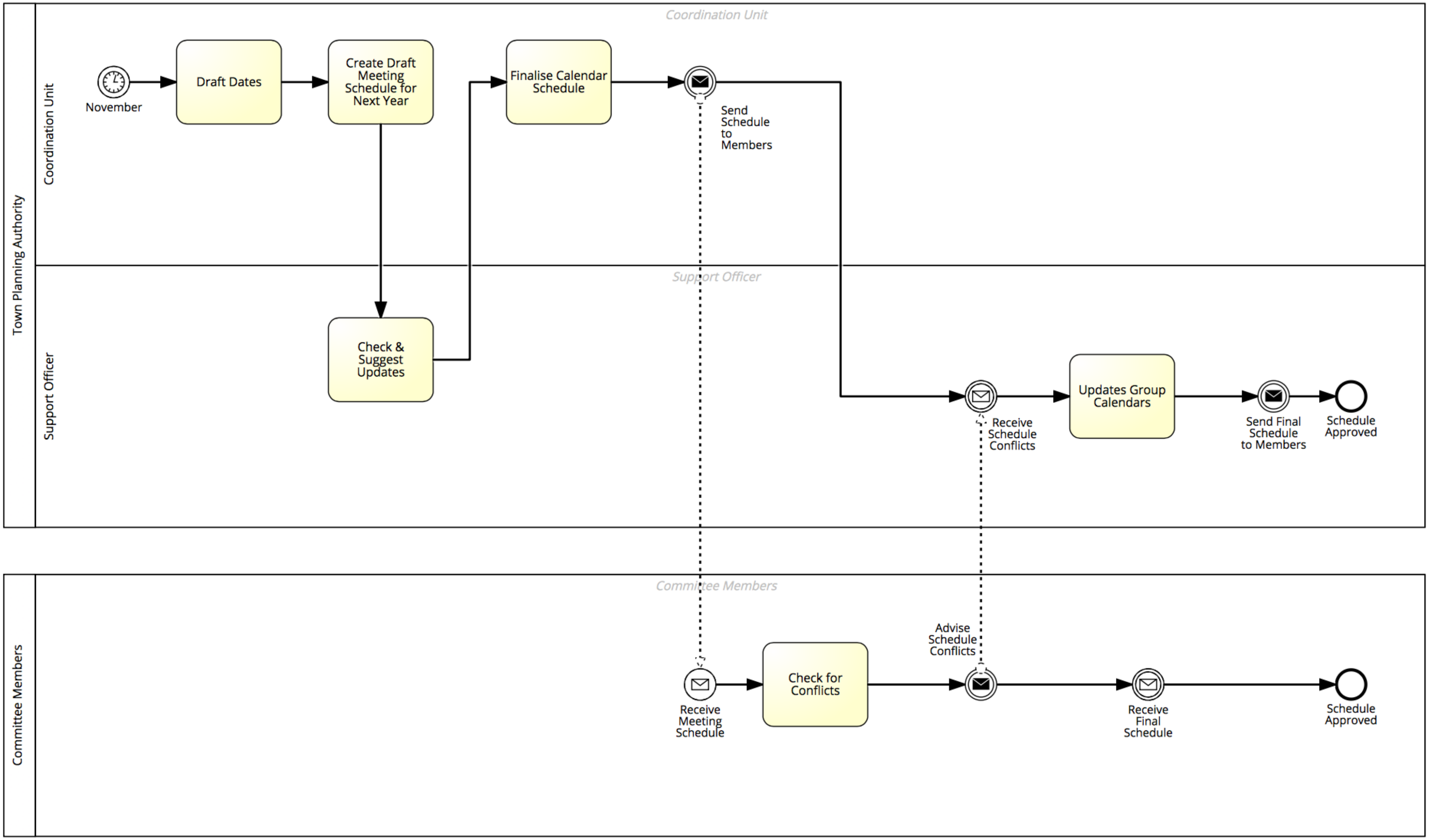

Transaction
Bookings
Book Flight
Cancel Flight
Book Hotel
Send Hotel Cancellation
Failed Bookings
Send Unavailability Notice
Exceptions (Hazards)

Exercise – Part 1

In November of each year, the Coordination Unit at the Town Planning Authority drafts a schedule of meetings for the next calendar year and adds draft dates to all calendars.

The Support Officer then checks the dates and suggests modifications. The Coordination Unit then rechecks all dates and looks for potential conflicts. The final schedule of meeting dates is sent to all the independent Committee Members by email, who then check their diaries and advise the Coordination Unit of any conflicts.

Solution – Part 1

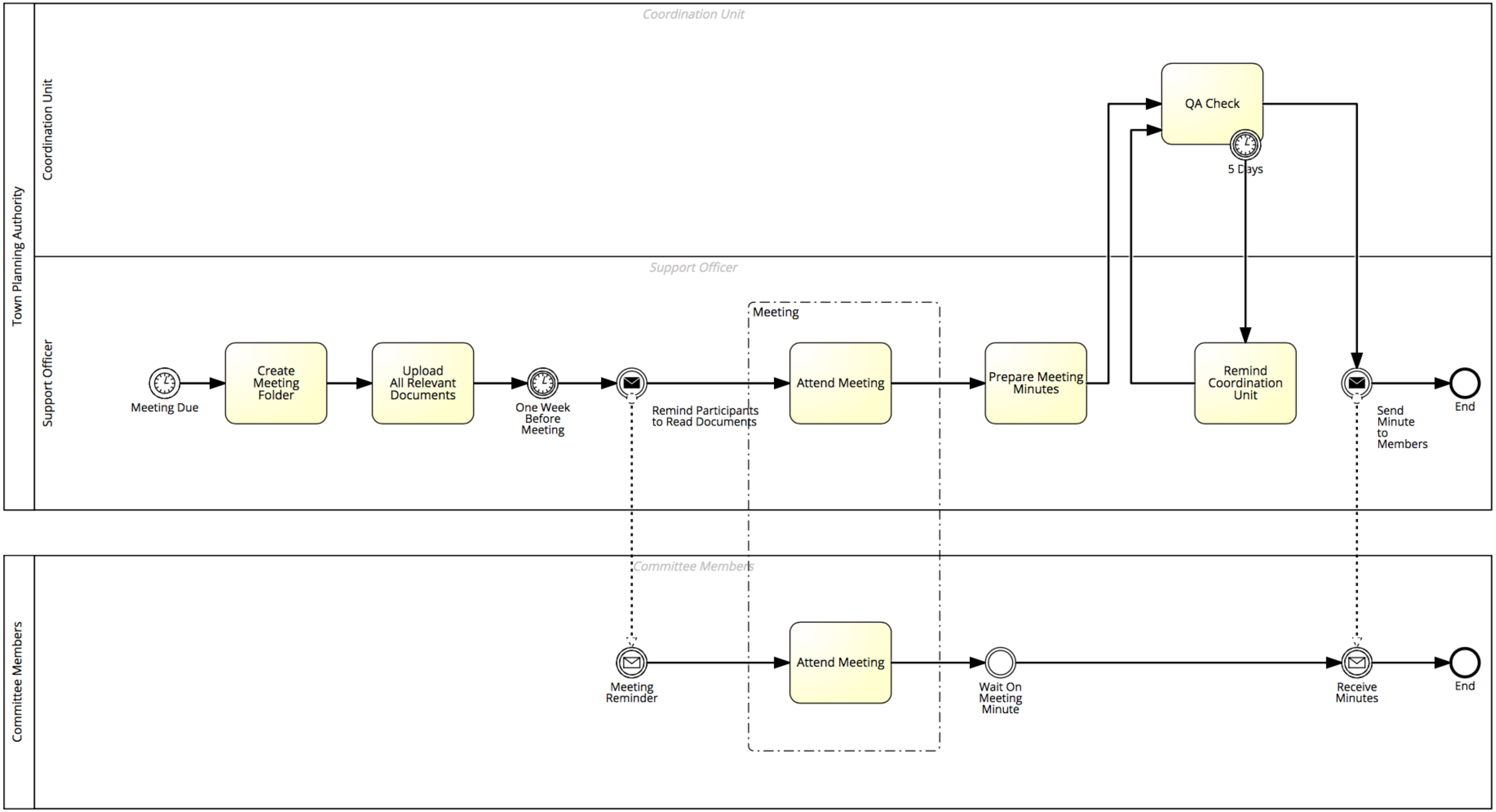


Exercise – Part 2

Once the dates are finalized (by the Coordination Unit), the Support Officer updates all group calendars and creates meeting folders for each meeting and ensures all appropriate documents are uploaded to system.

Committee Members are advised a week before each meeting to read all related documents. The Committee Members hold their meeting, and the Support Office then produces minutes including any Action Points for each Committee Member. Within 5 working days, the Coordination Unit must conduct a QA check on the minutes, which are then sent to all Committee Members. The Support Officer then updates all departmental records.

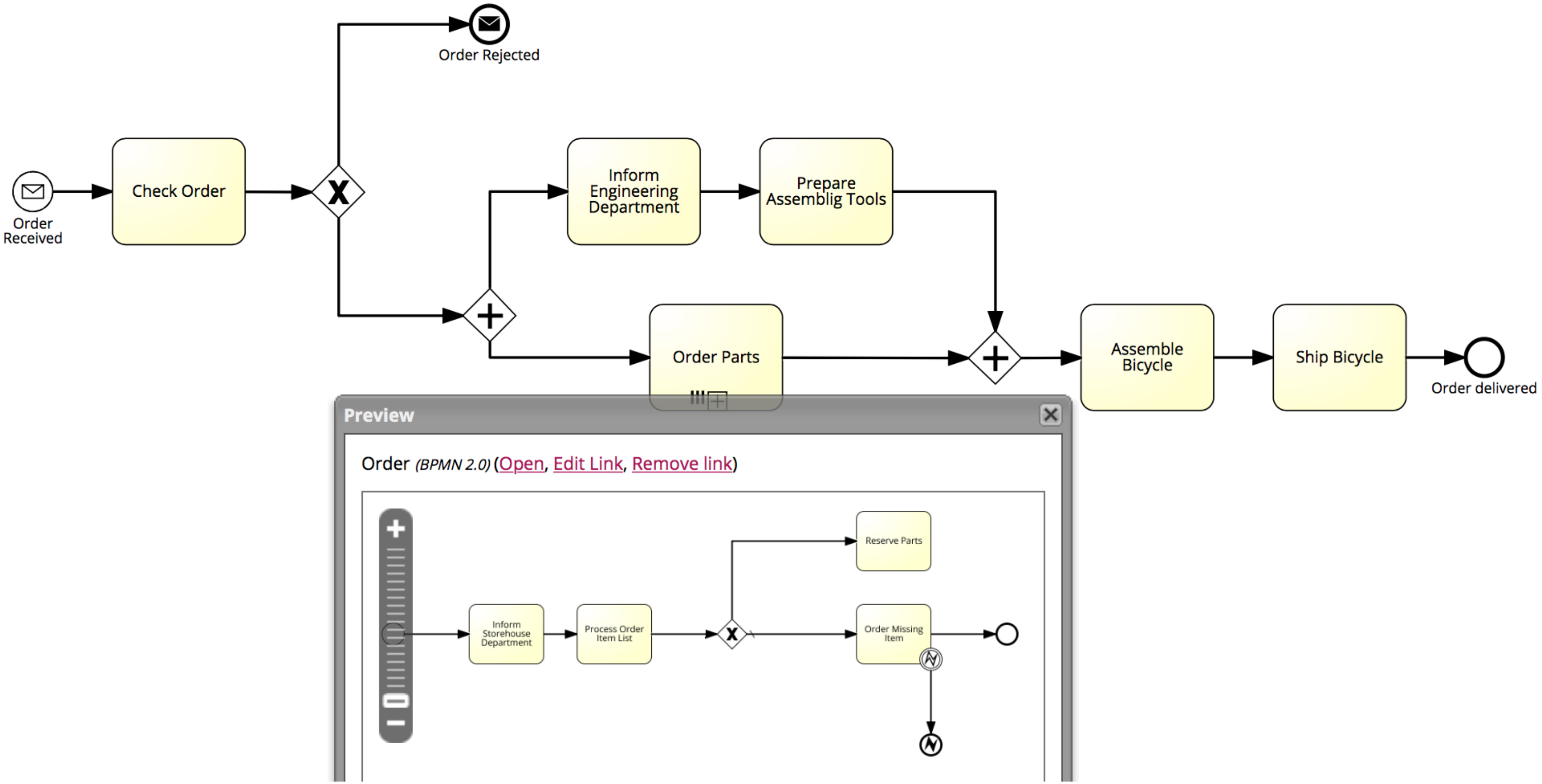
Solution – Part 2



Exercise

A small company manufactures customized bicycles. Whenever the sales department receives an order, a new process instance is created. A member of the sales department can then reject or accept the order for a customized bike. In the former case, the process instance is finished. In the latter case, the storehouse and the engineering department are informed. The storehouse immediately processes the part list of the order and checks the required quantity of each part. If the part is available in-house, it is reserved. If it is not available, it is back-ordered. This procedure is repeated for each item on the part list. In the meantime, the engineering department prepares everything for the assembling of the ordered bicycle. If the storehouse has successfully reserved or back-ordered every item of the part list and the preparation activity has finished, the engineering department assembles the bicycle. Afterwards, the sales department ships the bicycle to the customer and finishes the process instance.

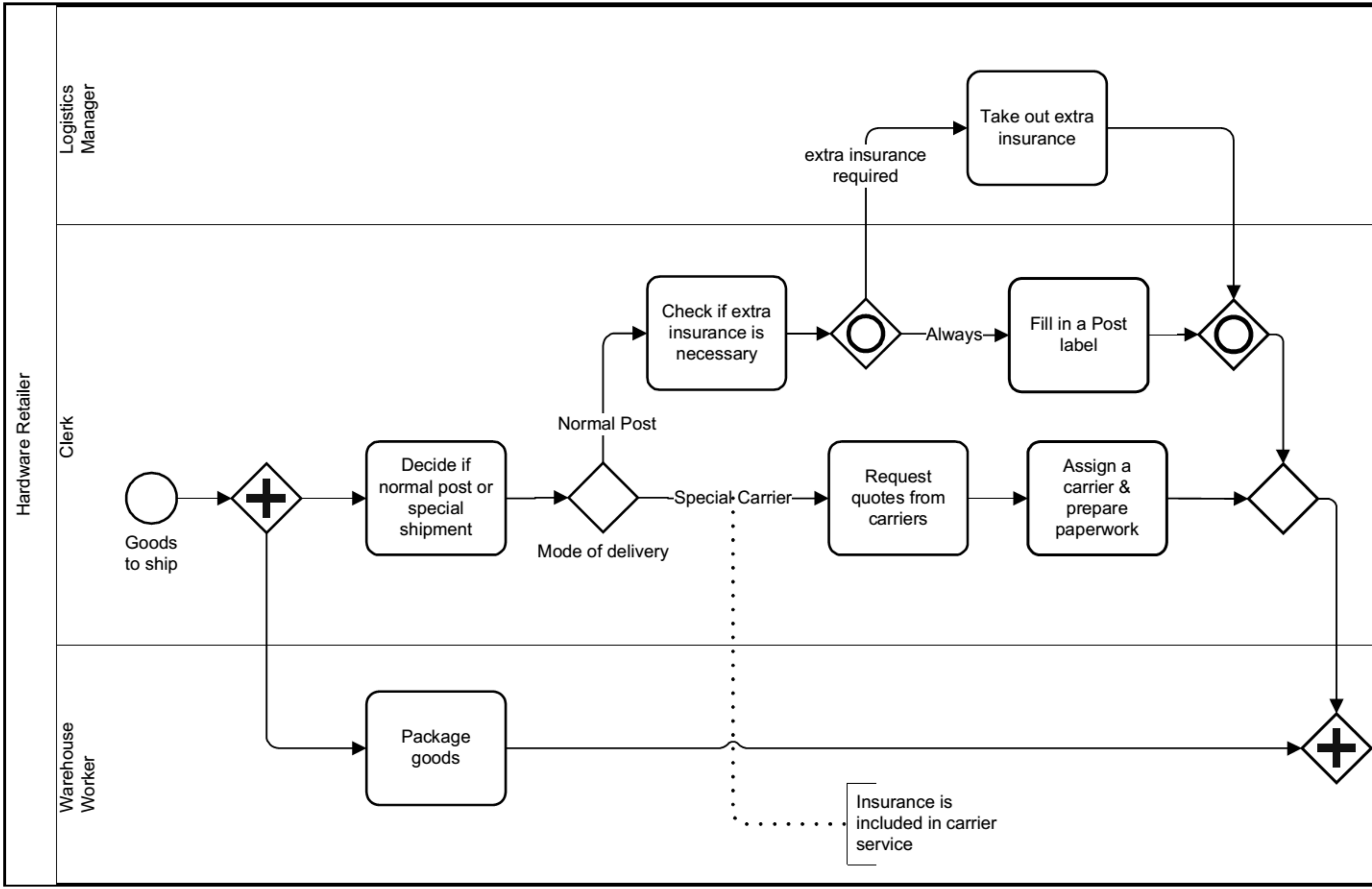
Solution

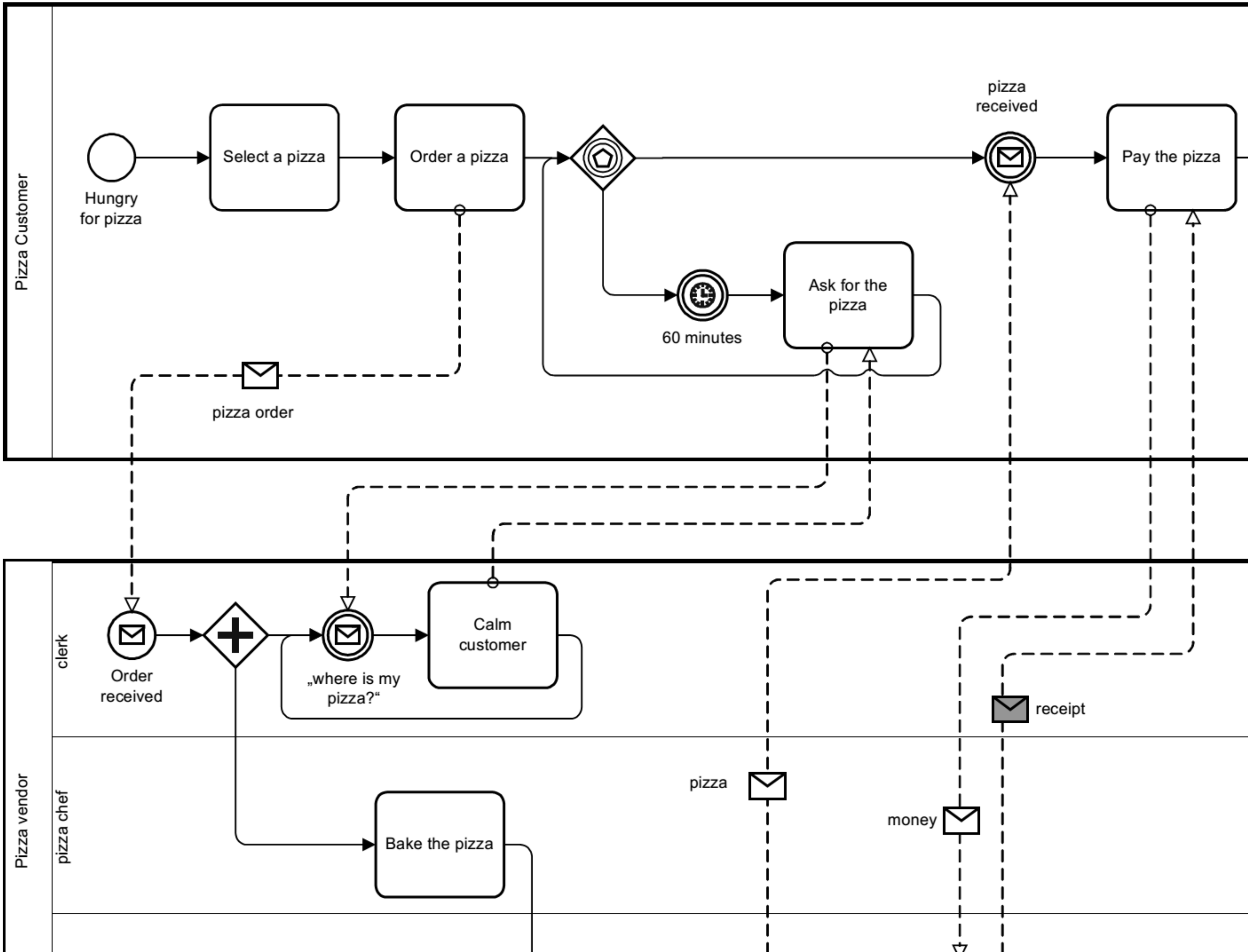


Order *(BPMN 2.0)* (Open, Edit Link, Remove link)

References

- Marlon Dumas – *Fundamentals of BPM*
- **Andrea Marrella – *Modeling Business Processes with BPMN***
- OMG, *BPMN by example*


Hardware Retailer
Logistics Manager
Clerk
Warehouse Worker
Goods to ship
Decide if normal post or special shipment
Mode of delivery
Normal Post
Special Carrier
Check if extra insurance is necessary
extra insurance required
Take out extra insurance
Always
Fill in a Post label
Request quotes from carriers
Assign a carrier & prepare paperwork
Package goods
Insurance is included in carrier service


Pizza Customer
Hungry for pizza
Select a pizza
Order a pizza
pizza received
Pay the pizza
Ask for the pizza
60 minutes
pizza order
Pizza vendor
clerk
Order received
„where is my pizza?“
Calm customer
receipt
pizza chef
Bake the pizza
pizza
money