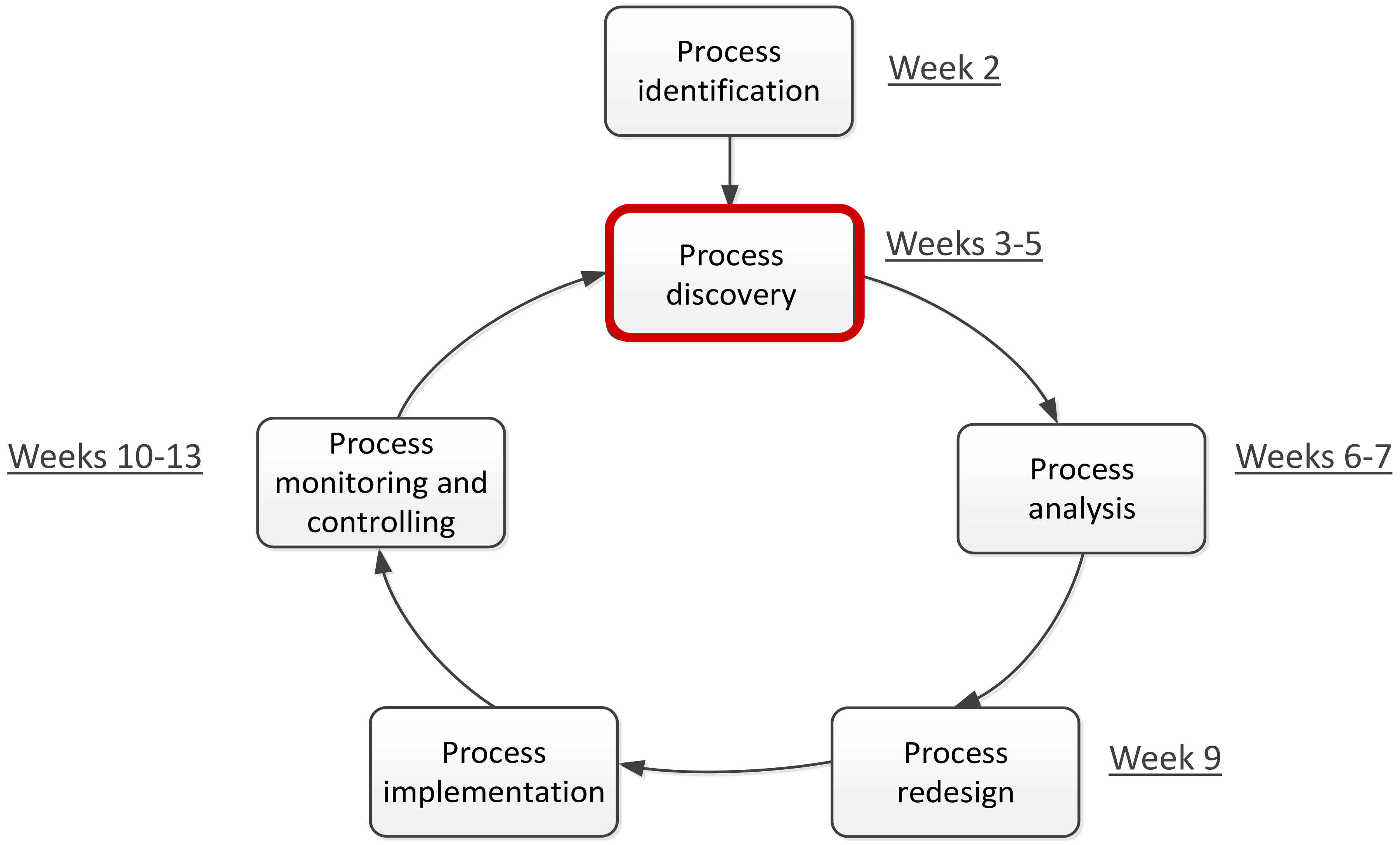
Business Process Management

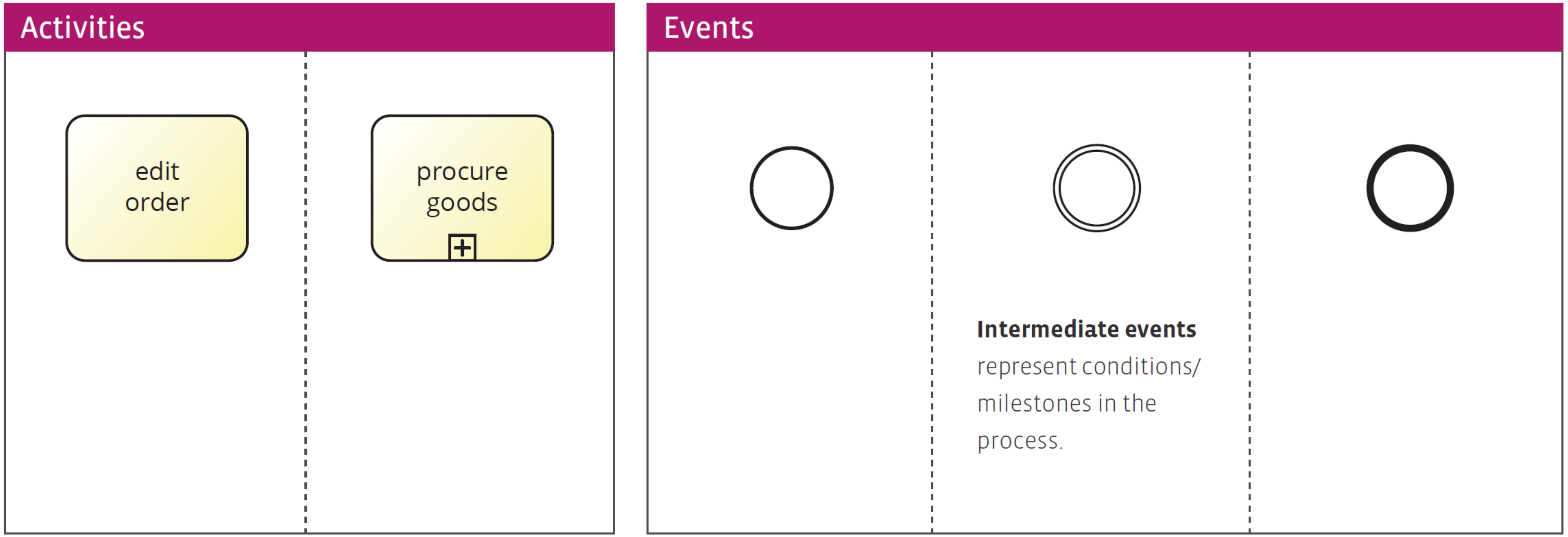
Lecture
Advanced Process Modeling

Prof. Josué Obregón
Department of Industrial Engineering - ITM
Seoul National University of Science and Technology

Course structure



BPMN Recap

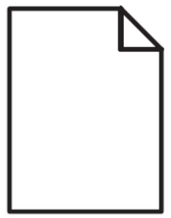


Intermediate events represent conditions/ milestones in the process.

BPMN Recap

Connecting Objects

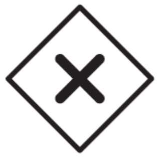
Artifacts

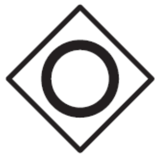


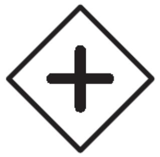
IT Systems represent certain systems, which are utilized to execute activities.

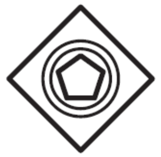
BPMN Recap

Gateways



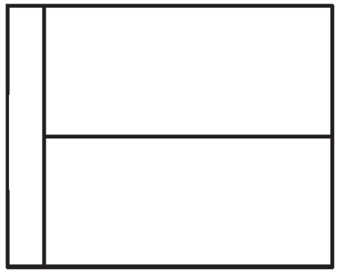






Event-based Gate-ways are used if the ongoing process flow depends on the occurrence of an event. The process will continue with the first occurring event.

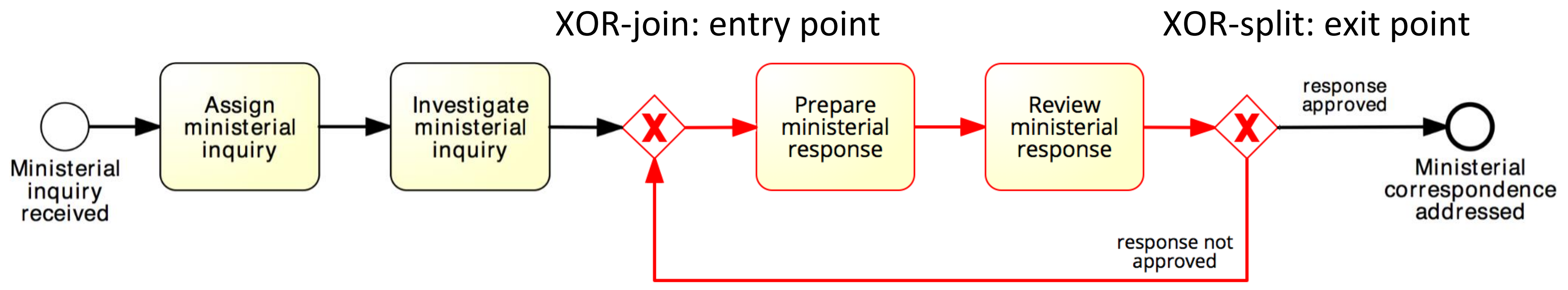
Swimlanes



Contents

- Advanced Process Modeling
  - More on Rework and Repetition
  - Handling Events
  - Handling Exceptions

More on rework and repetition

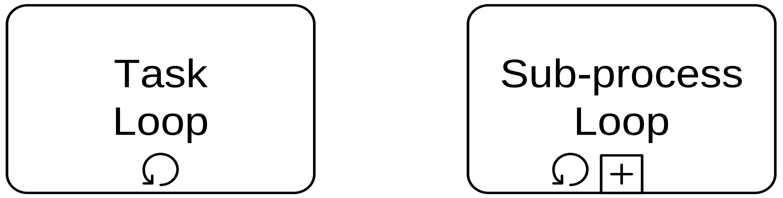


XOR-join: entry point
XOR-split: exit point
Ministerial inquiry received
Assign ministerial inquiry
Investigate ministerial inquiry
Prepare ministerial response
Review ministerial response
response approved
response not approved
Ministerial correspondence addressed

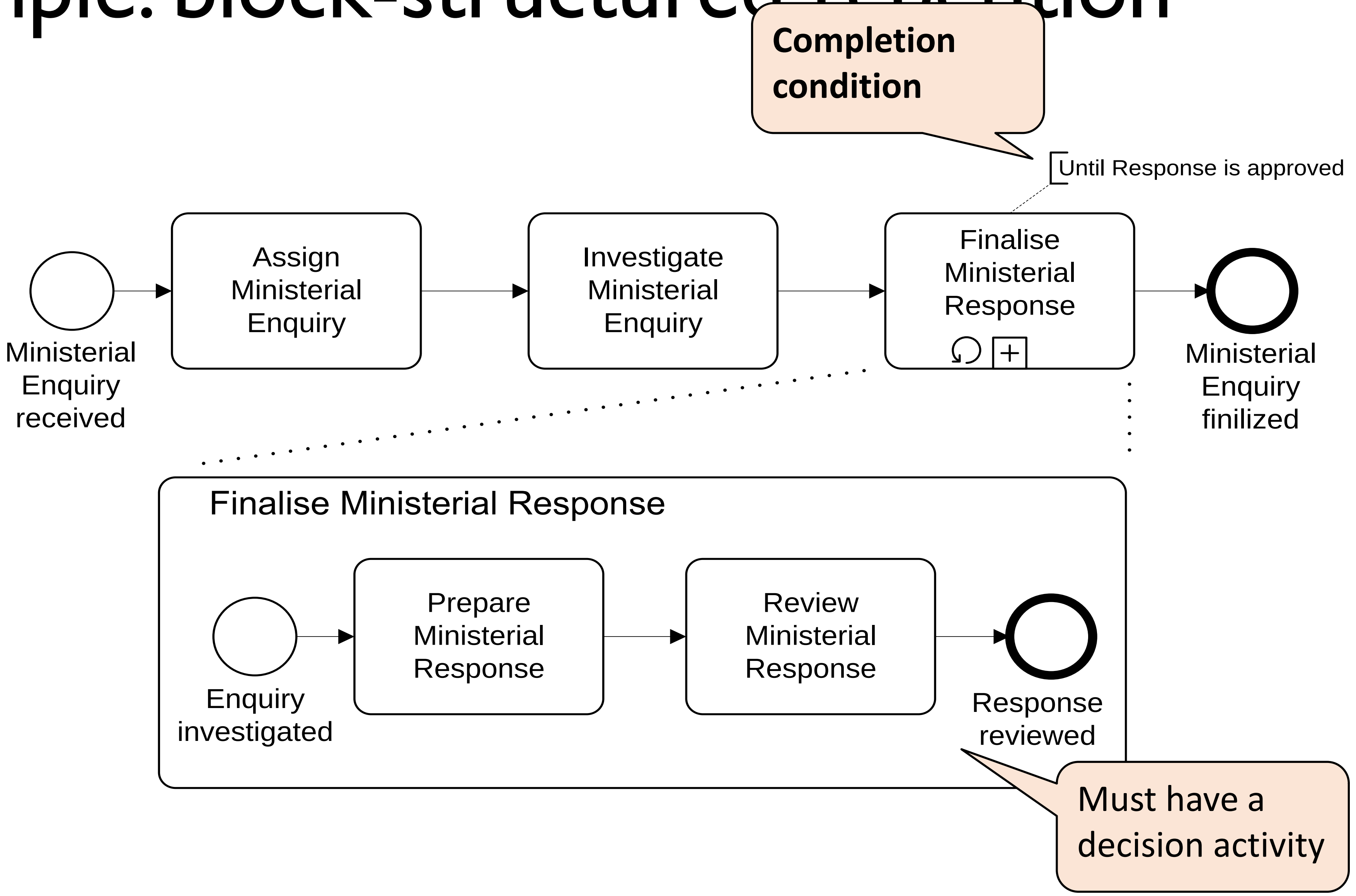
Structured cycle (SESE component)

Block-structured repetition: Activity loop

The *Loop Activity* markers allow us to state that a task or a sub-process may be repeated multiple times



Example: block-structured repetition


Completion condition
Until Response is approved
Ministerial Enquiry received
Assign Ministerial Enquiry
Investigate Ministerial Enquiry
Finalise Ministerial Response
Ministerial Enquiry finilized
Finalise Ministerial Response
Enquiry investigated
Prepare Ministerial Response
Review Ministerial Response
Response reviewed
Must have a decision activity

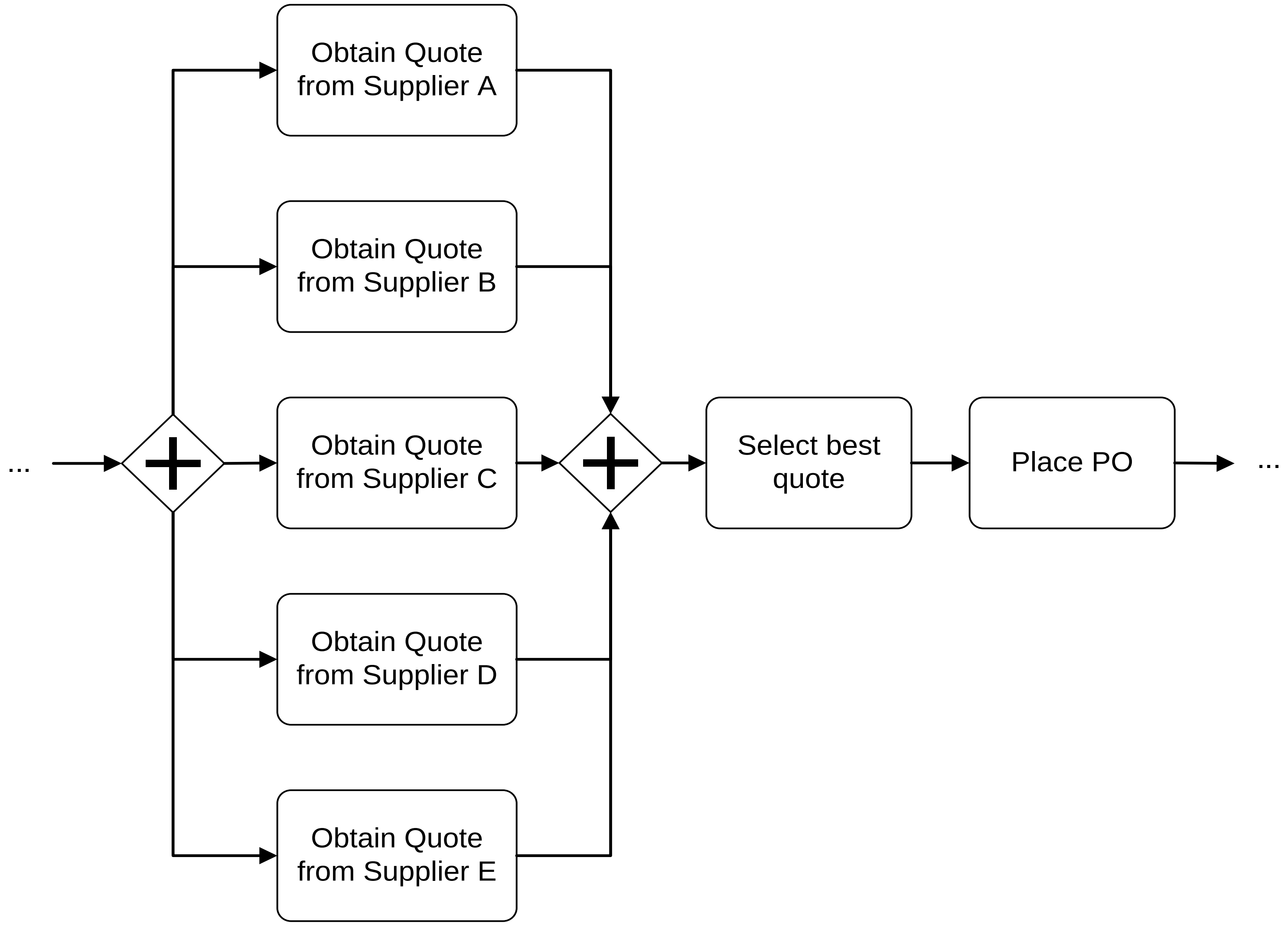
Example: multi-instance activity

Procurement

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

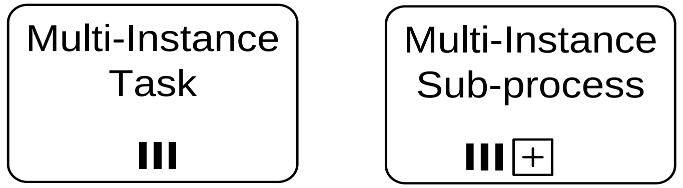
Solution: without multi-instance activity

Procurement


...
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
...

Parallel repetition: multi-instance activity

The multi-instance activity provides a mechanism to indicate that an activity is executed ***multiple times concurrently***

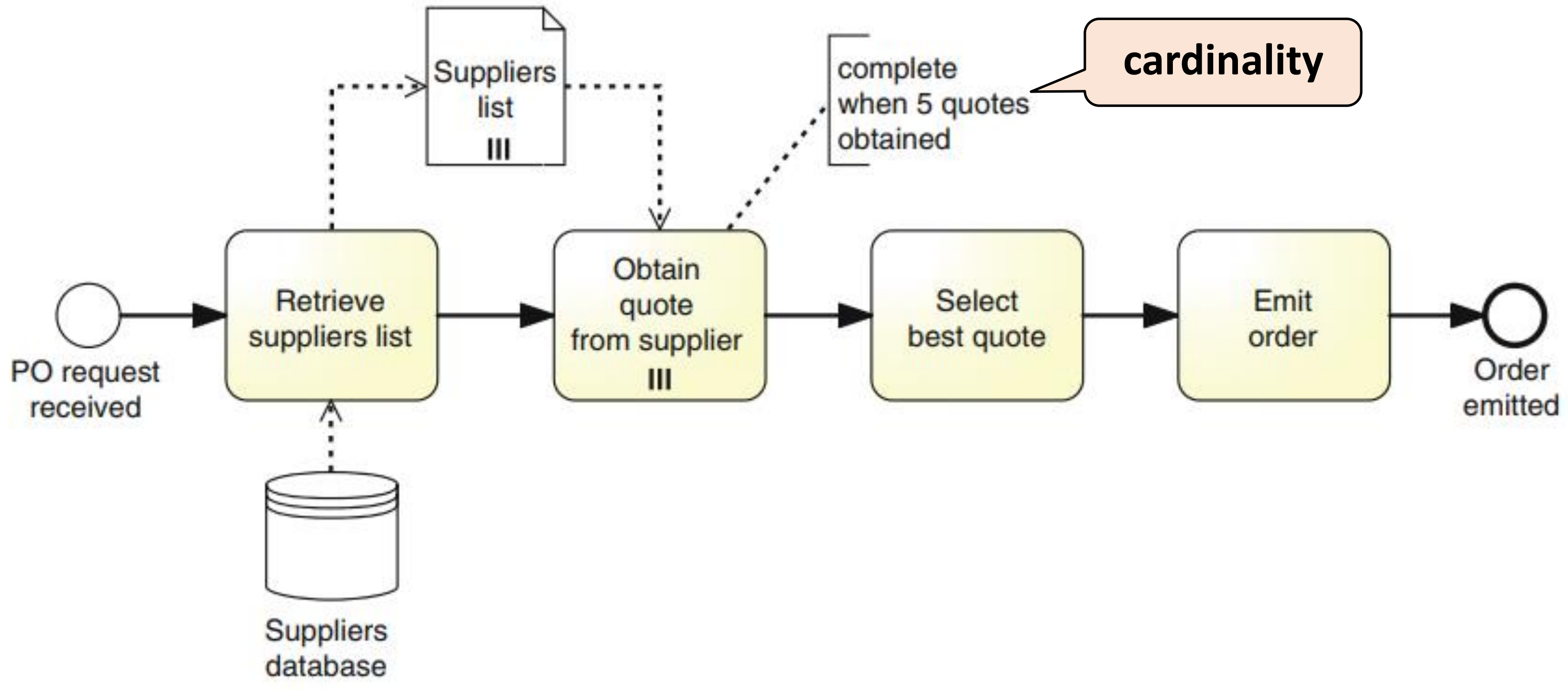


Useful when the same activity needs to be executed for multiple entities or data items, such as:

- Request quotes from multiple suppliers
- Check the availability for each line item in an order separately
- Send and gather questionnaires from multiple witnesses in the context of an insurance claim

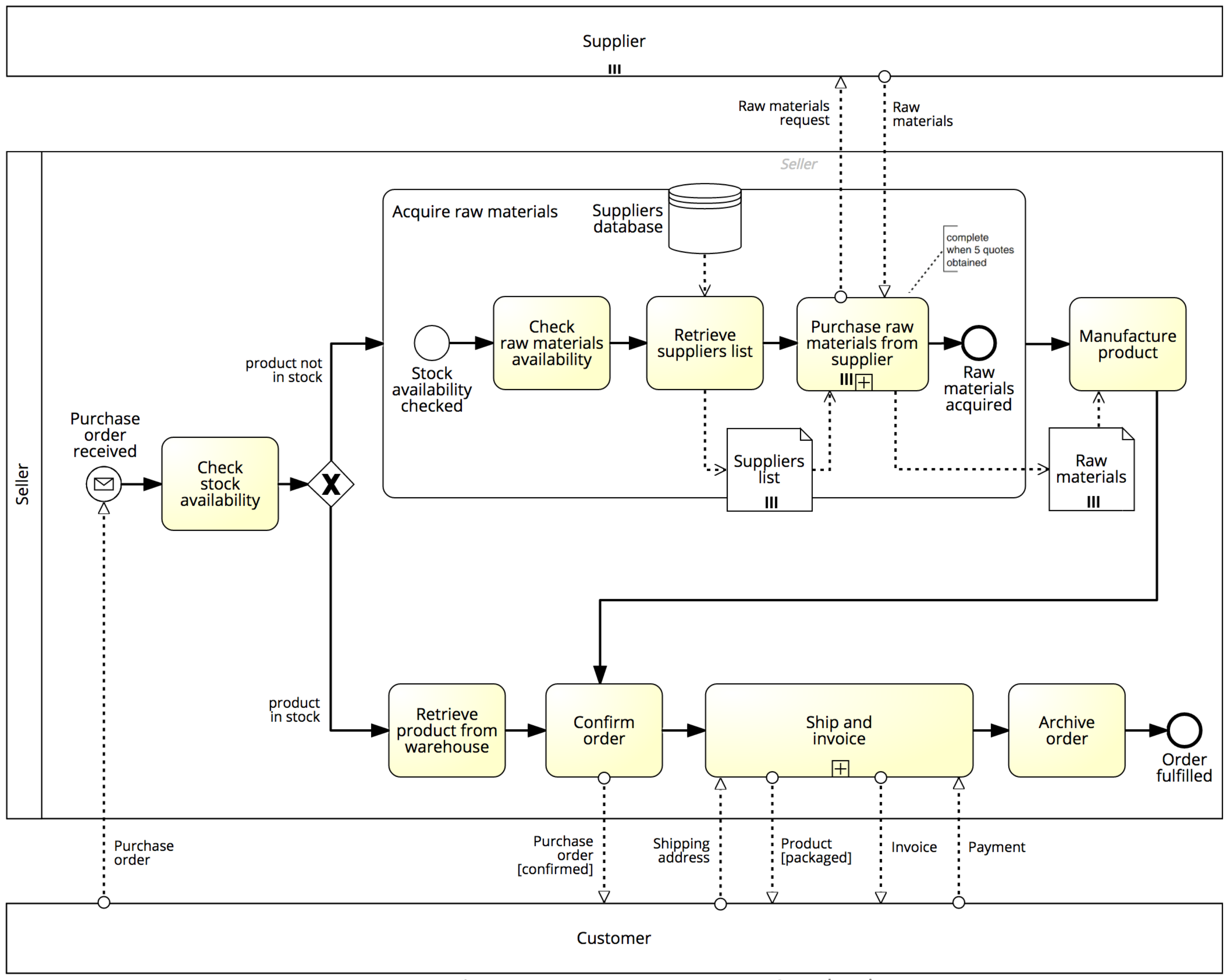
Solution: with multi-instance activity

Procurement



Our order-to-cash example…

now with pools, messages and MI markers



Exercise 4.2

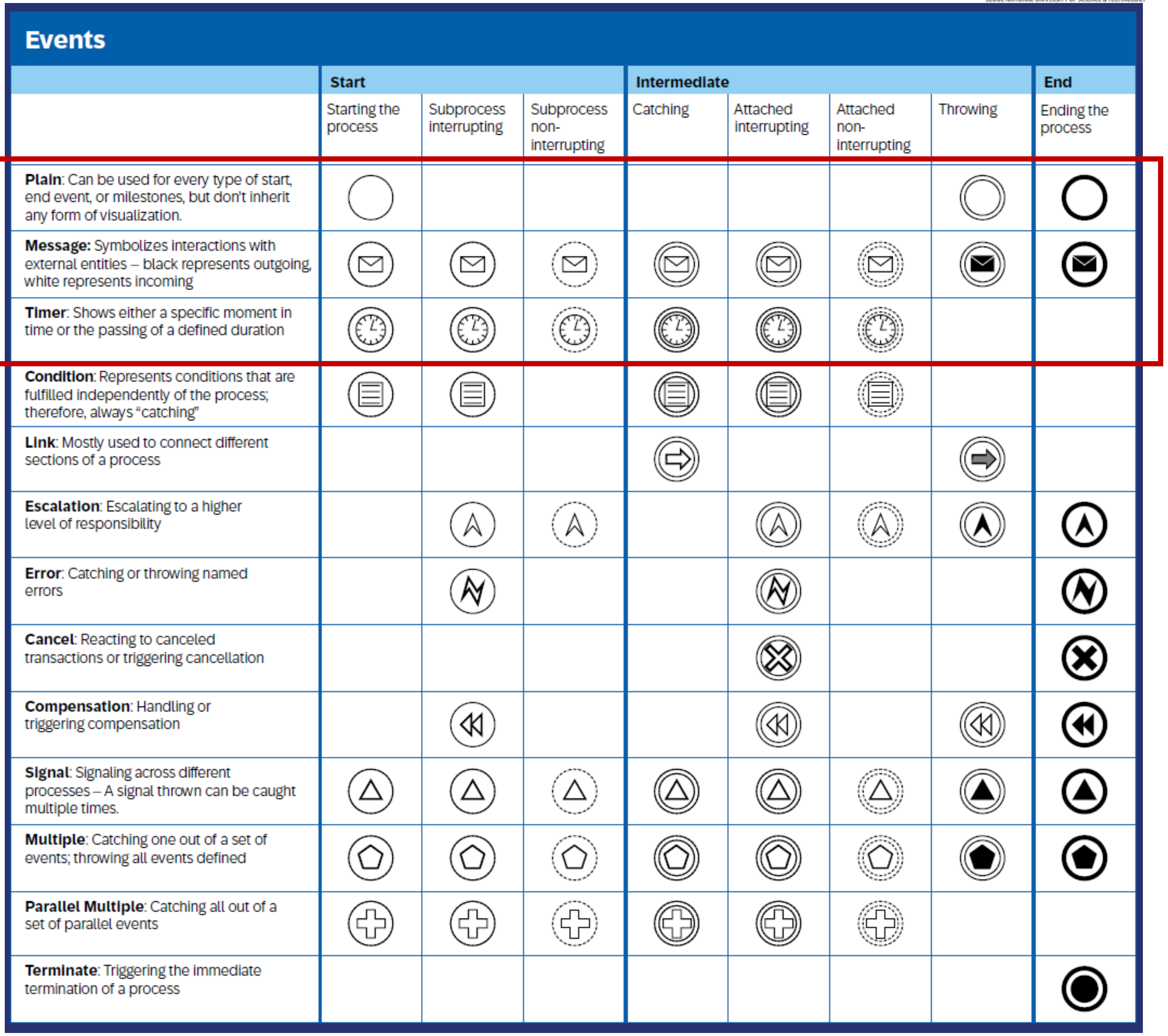
Model the following motor insurance claim lodgement (present or request)

After a car accident, a statement is sought from two witnesses out of the five that were present multi-instance request the insurance claim. As soon as the first two statements are received, the claim can be requested with the insurance company without waiting for the other statements.

Note: Use pools for different participants in the process, multi-instance markers, annotations and data objects if needed.

Handling Events

Message events, temporal events and racing events



Events

	Start			Intermediate				End
	Starting the process	Subprocess interrupting	Subprocess non-interrupting	Catching	Attached interrupting	Attached non-interrupting	Throwing	Ending the process
Plain: Can be used for every type of start, end event, or milestones, but don't inherit any form of visualization.	○						○	○
Message: Symbolizes interactions with external entities – black represents outgoing, white represents incoming	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Shows either a specific moment in time or the passing of a defined duration	⏲	⏲	⏲	⏲	⏲	⏲		
Condition: Represents conditions that are fulfilled independently of the process; therefore, always "catching"	▤	▤		▤	▤	▤		
Link: Mostly used to connect different sections of a process				⇨			➡	
Escalation: Escalating to a higher level of responsibility		⮝	⮝		⮝	⮝	⮝	⮝
Error: Catching or throwing named errors		ϟ			ϟ			ϟ
Cancel: Reacting to canceled transactions or triggering cancellation					⊗			⊗
Compensation: Handling or triggering compensation		⏪			⏪		⏪	⏪
Signal: Signaling across different processes – A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events; throwing all events defined	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events	✚	✚	✚	✚	✚	✚		
Terminate: Triggering the immediate termination of a process								◉

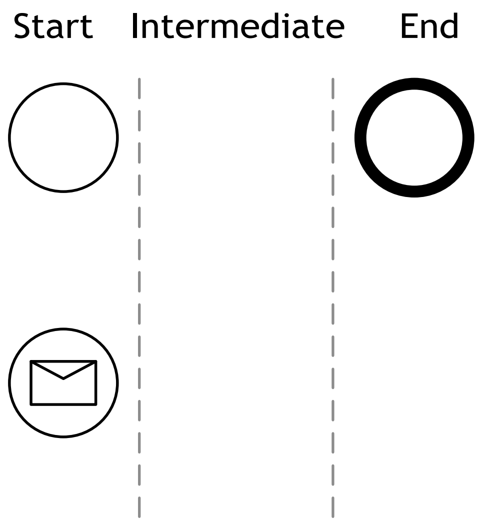
Events

In BPMN, events model something instantaneous happening during the execution of a process

Types of event:

- Start
  - Signal how process instances start (tokens are created)
- Intermediate
  - Occurs during a process
- End
  - Signal when process instances complete (tokens are destroyed)

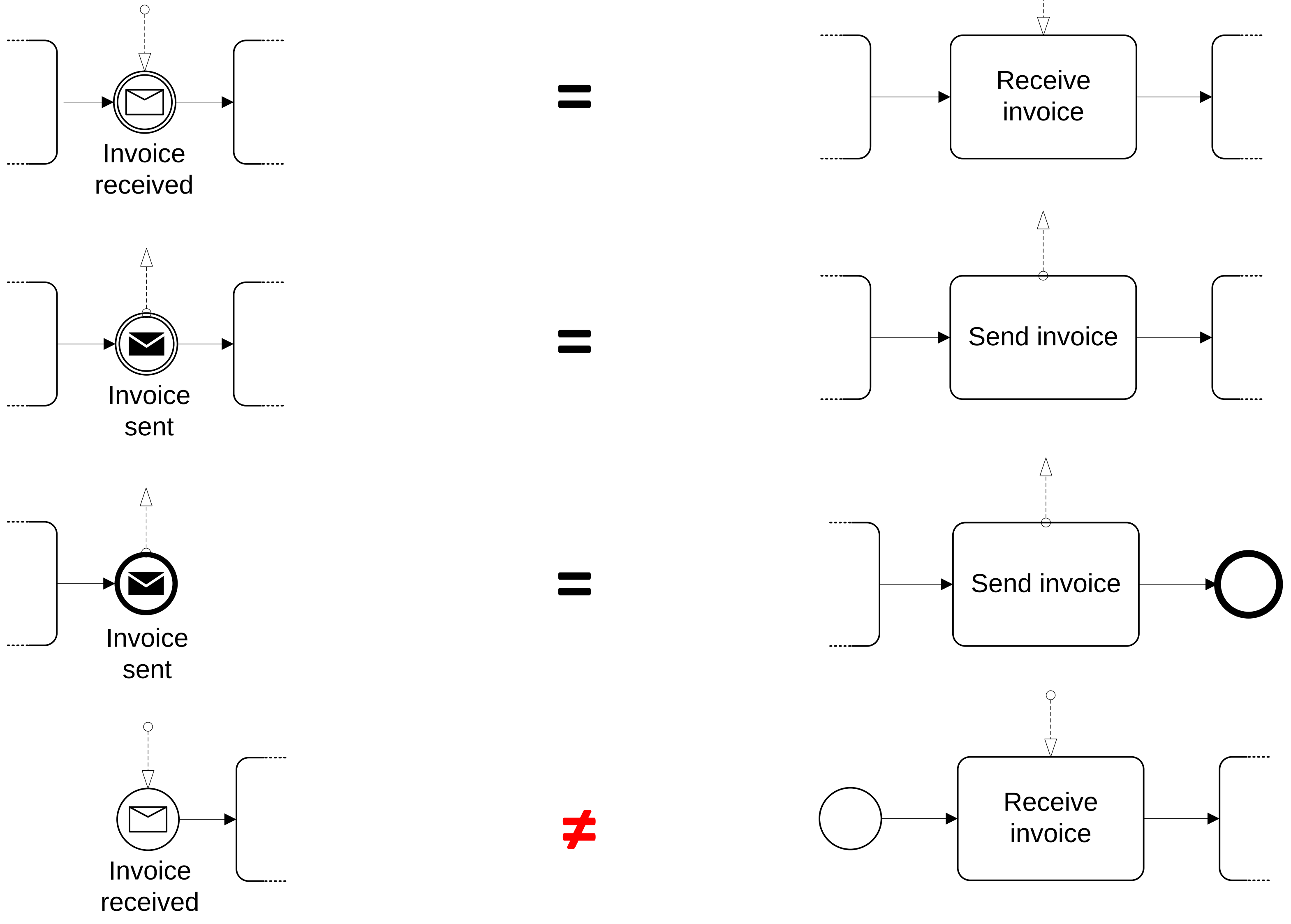
BPMN event types



Untyped Event – Indicates that an instance of the process is created (start) or completed (end), without specifying the cause for creation/completion

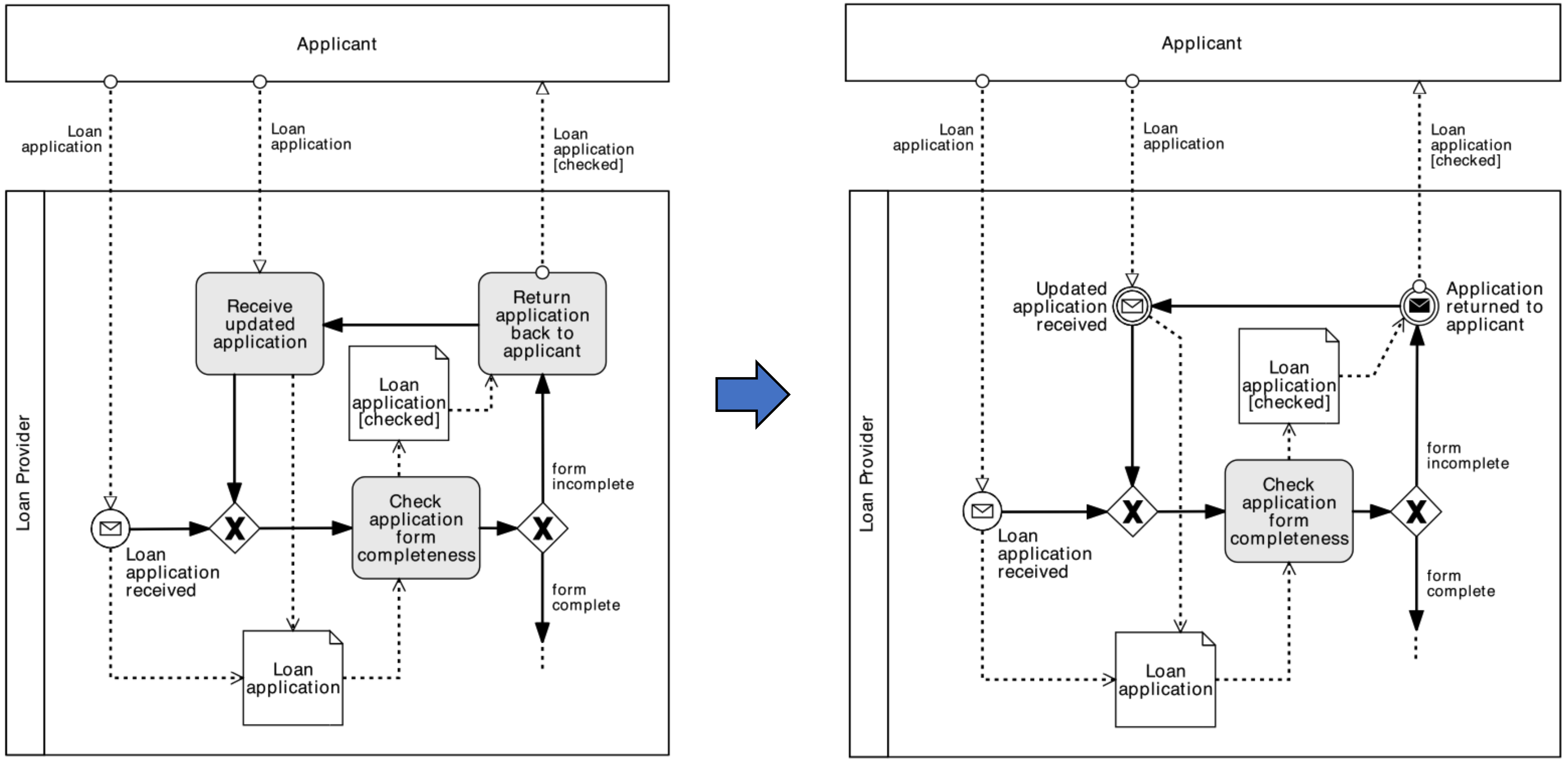
Start Message Event – Indicates that an instance of the process is created when a message is **received**

Comparison with sending/receiving tasks



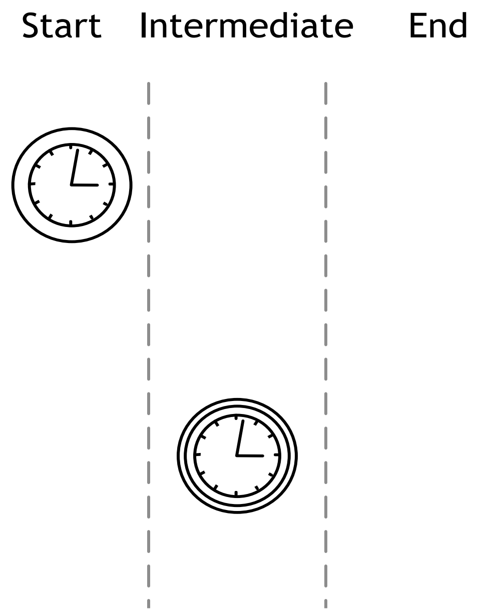
So, when to use what?

Use message events only when the corresponding activity would simply send or receive a message and do nothing else



Applicant
Loan application
Loan application
Loan application [checked]
Loan Provider
Receive updated application
Return application back to applicant
Loan application [checked]
Check application form completeness
form incomplete
form complete
Loan application received
Loan application
Applicant
Loan application
Loan application
Loan application [checked]
Loan Provider
Updated application received
Application returned to applicant
Loan application [checked]
Check application form completeness
form incomplete
form complete
Loan application received
Loan application

Temporal events



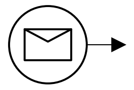
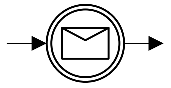
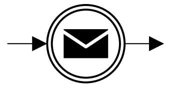
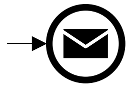
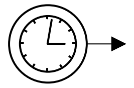
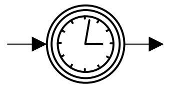
Start Timer Event – Indicates that an instance of the process is created at certain date(s)/time(s), e.g. start process at 6pm every Friday

Intermediate Timer Event – Triggered at certain date(s)/ time(s), or after a time interval has elapsed since the moment the event is "enabled" (delay)

Example

A Purchase Order (PO) handling process starts when a PO is received. The PO is first registered. If the current date is not a working day, the process waits until the following working day before proceeding. Otherwise, an availability check is performed and a PO response is sent back to the customer.

Recap: Message and Timer events

Start	Intermediate		End
Catching	Catching	Throwing	Throwing
 process starts upon message received	 message received during the process	 message sent during the process	 process ends upon message sent
 process starts when time event occurs	 time event occurred (to model delay)		

Data-based vs. event-based choices

- In an XOR-split gateway, one branch is chosen based on expressions evaluated over available data
  - →Choice is made immediately when the gateway is reached
- Sometimes, the choice must be delayed until something happens
  - →Choice is based on a “race between events”
- BPMN distinguishes between:
  - Exclusive decision gateway (XOR-split)
  - Event-based decision gateway

Event-based decision

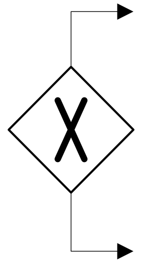
With the XOR-split gateway, a branch is chosen based on conditions that evaluate over available data

→ The choice can be made immediately after the token arrives from the incoming flow

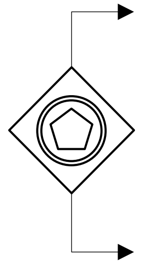
Sometimes, the choice must be delayed until an event happens

→ The choice is based on a "race" among events

data-driven XOR-split



event-based gateway

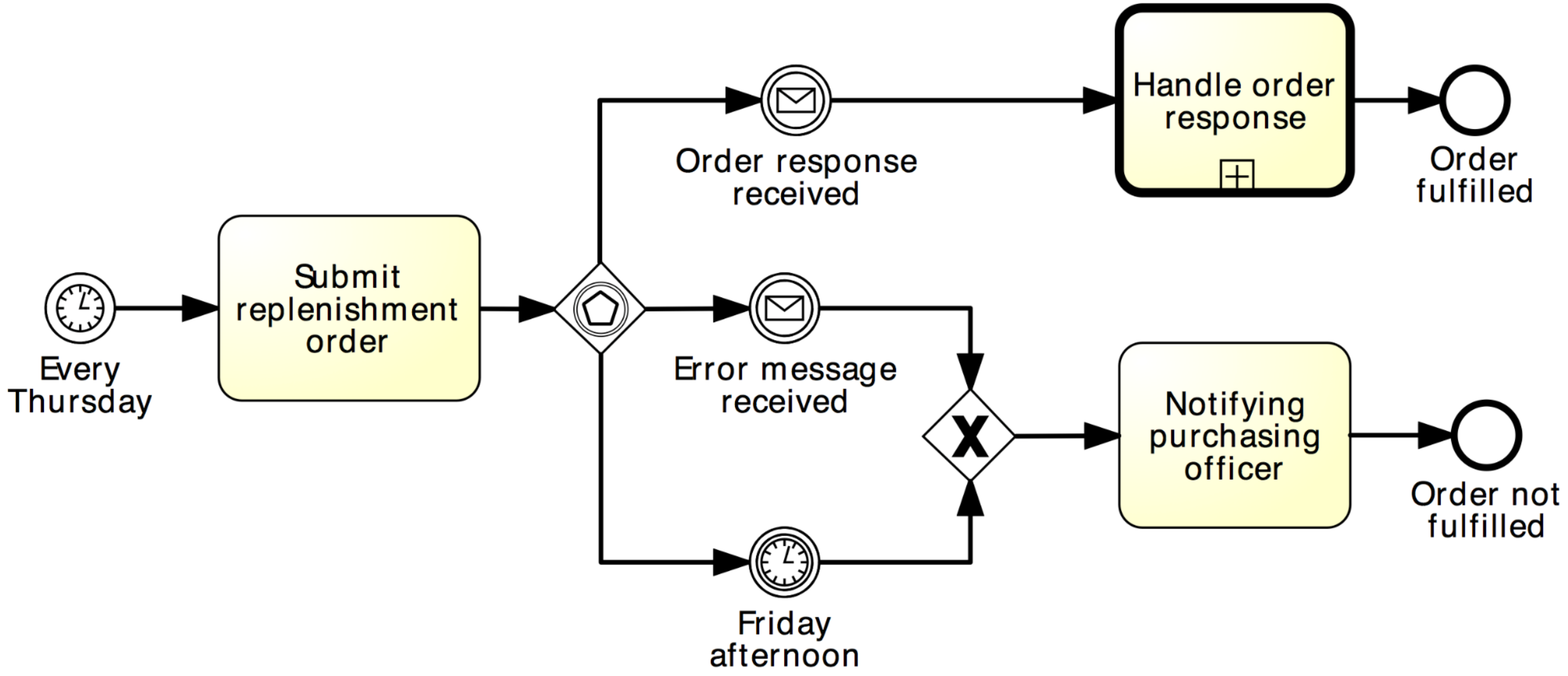


Choices outside our control…

A restaurant chain submits a purchase order (PO) to replenish its warehouses every Thursday. The restaurant chain's procurement system expects to receive either a "PO Response" or an error message. However, it may also happen that no response is received at all due to system errors or due to delays in handling the PO on the supplier's side. If no response is received by Friday afternoon or if an error message is received, a purchasing officer at the restaurant chain's headquarters should be notified. Otherwise, the PO Response is processed normally.

Solution: event-driven XOR split

Stock replenishment



Exercise

Model the following fragment of a process

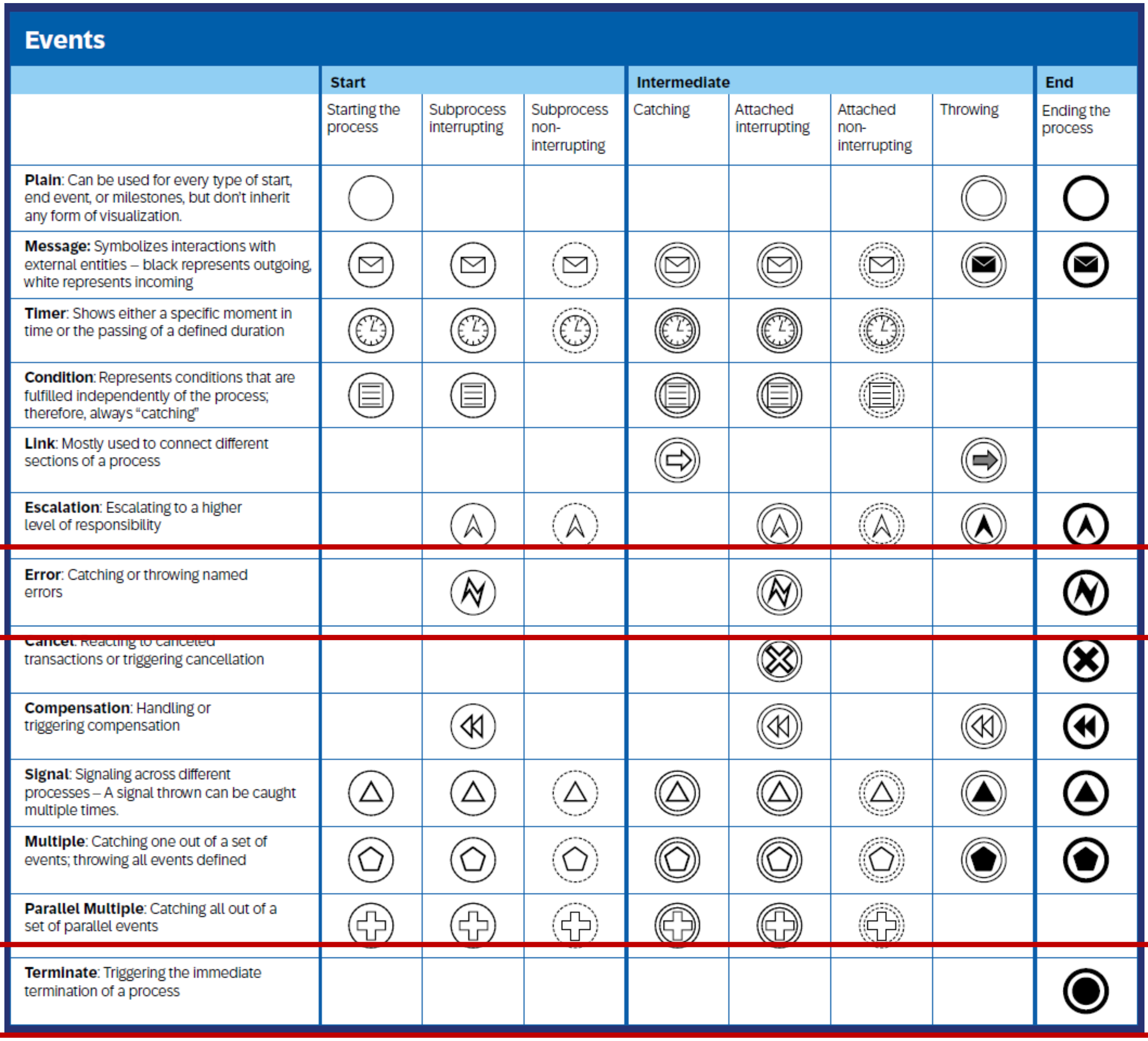
In the context of a claim handling process, it is sometimes necessary to send a questionnaire to the claimant to gather additional information.

The claimant is expected to return the questionnaire within five days.

If no response is received after five days, a reminder is sent to the claimant. If after another five days there is still no response, another reminder is sent and so on until the completed questionnaire is received.

Handling Exceptions

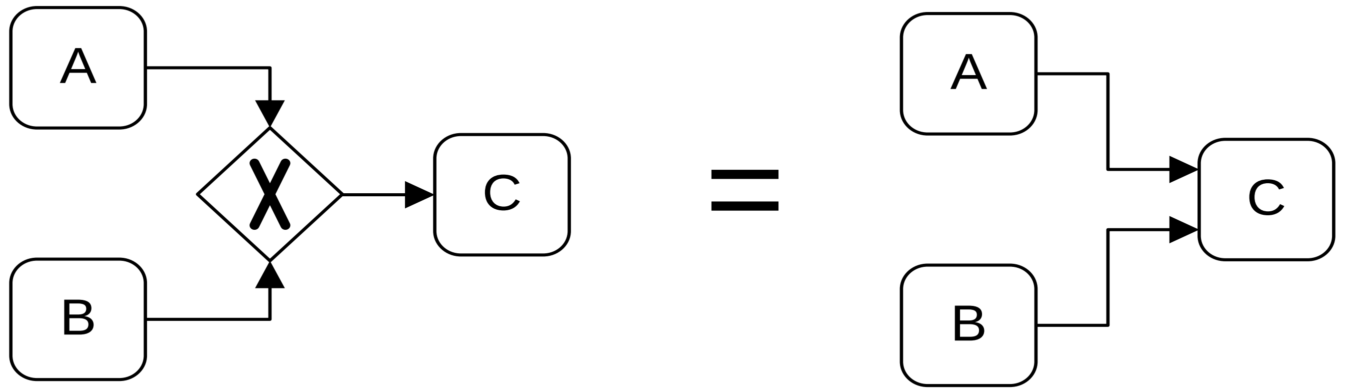
Process Abortion, Internal and External Exceptions, Activity Timeouts

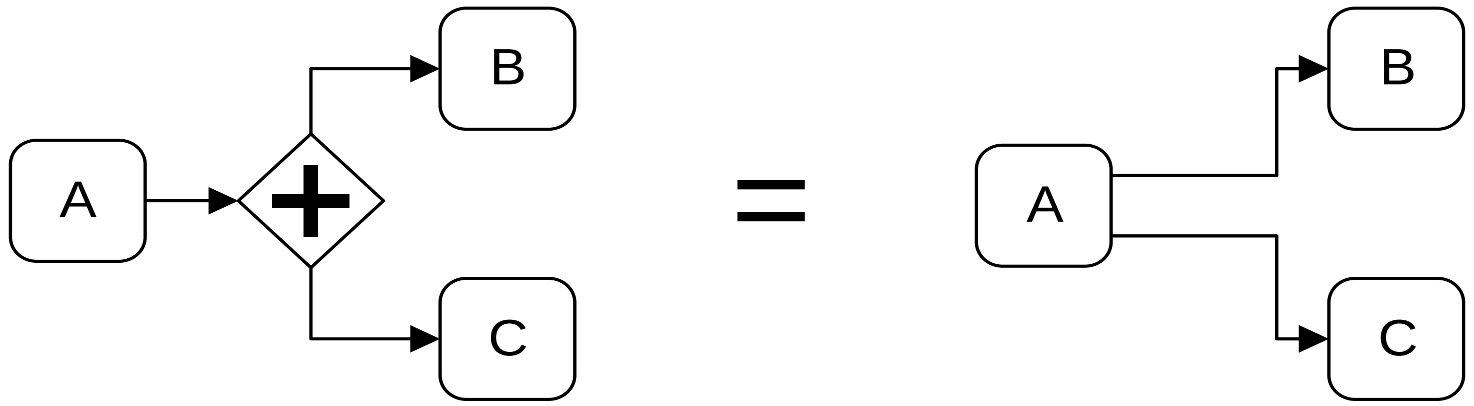


Events

	Start			Intermediate				End
	Starting the process	Subprocess interrupting	Subprocess non-interrupting	Catching	Attached interrupting	Attached non-interrupting	Throwing	Ending the process
Plain: Can be used for every type of start, end event, or milestones, but don't inherit any form of visualization.	○						○	○
Message: Symbolizes interactions with external entities – black represents outgoing, white represents incoming	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Shows either a specific moment in time or the passing of a defined duration	🕒	🕒	🕒	🕒	🕒	🕒		
Condition: Represents conditions that are fulfilled independently of the process; therefore, always "catching"	≡	≡		≡	≡	≡		
Link: Mostly used to connect different sections of a process				⇨			➡	
Escalation: Escalating to a higher level of responsibility		∧	∧		∧	∧	▲	▲
Error: Catching or throwing named errors		ϟ			ϟ			ϟ
Cancel: Reacting to canceled transactions or triggering cancellation					⊗			⊗
Compensation: Handling or triggering compensation		⏪			⏪		⏪	⏪
Signal: Signaling across different processes – A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events; throwing all events defined	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events	✚	✚	✚	✚	✚	✚		
Terminate: Triggering the immediate termination of a process								●

Quick Note: Implicit vs. explicit gateways

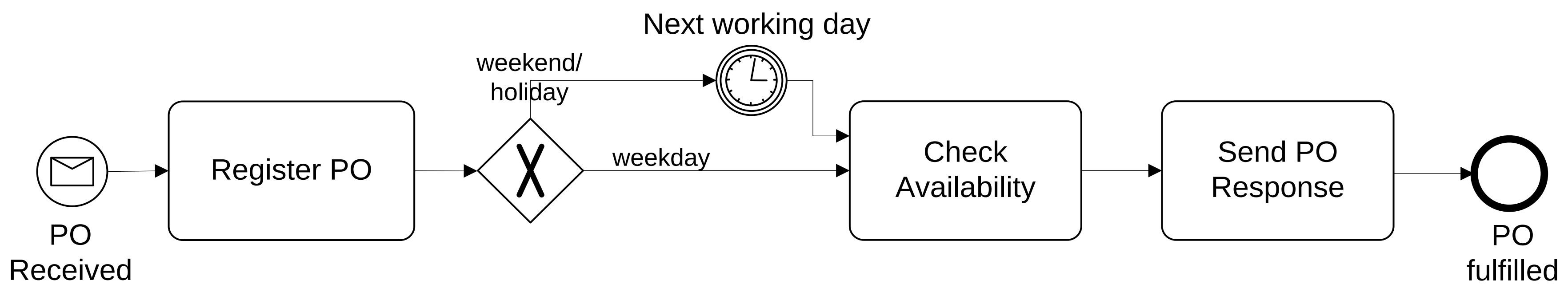




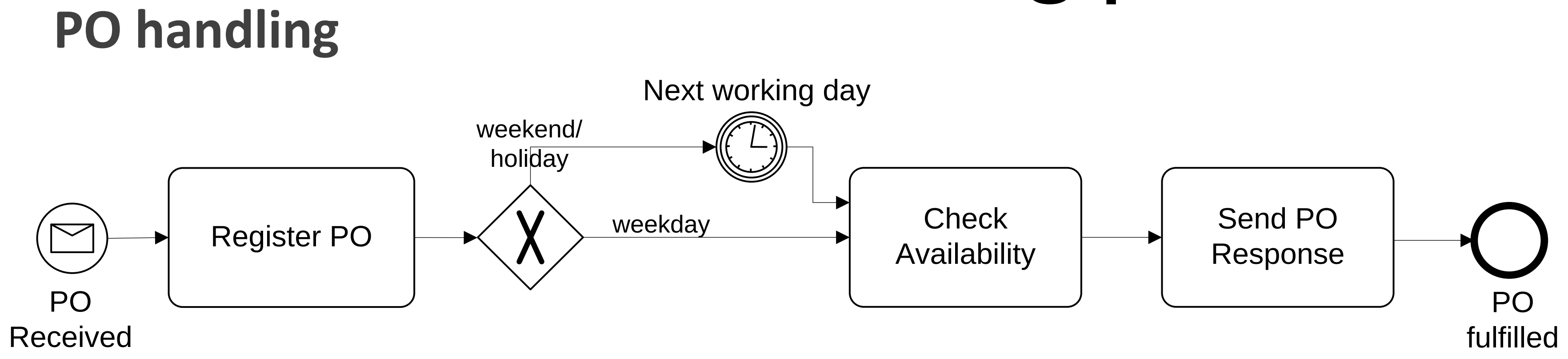
Let's extend our PO handling process

PO handling

A PO handling process starts when a PO is received. The PO is first registered. If the current date is not a working day, the process waits until the following working day before proceeding. Otherwise, an availability check is performed and a PO response is sent back to the customer.



Let's extend our PO handling process



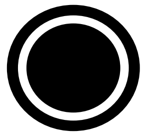
A PO change request may be received anytime after the PO is registered. This request includes a change in quantity or line items. When such a request is received, any processing related to the PO must be stopped. The PO change request is then registered. Thereafter, the process proceeds as it would do after a normal PO is registered. Further, if the customer sends a PO cancelation request after the PO registration, the PO processing must be stopped and the cancelation request must be handled.

Abortion (terminate event)

Exceptions are events that deviate a process from its “normal” course

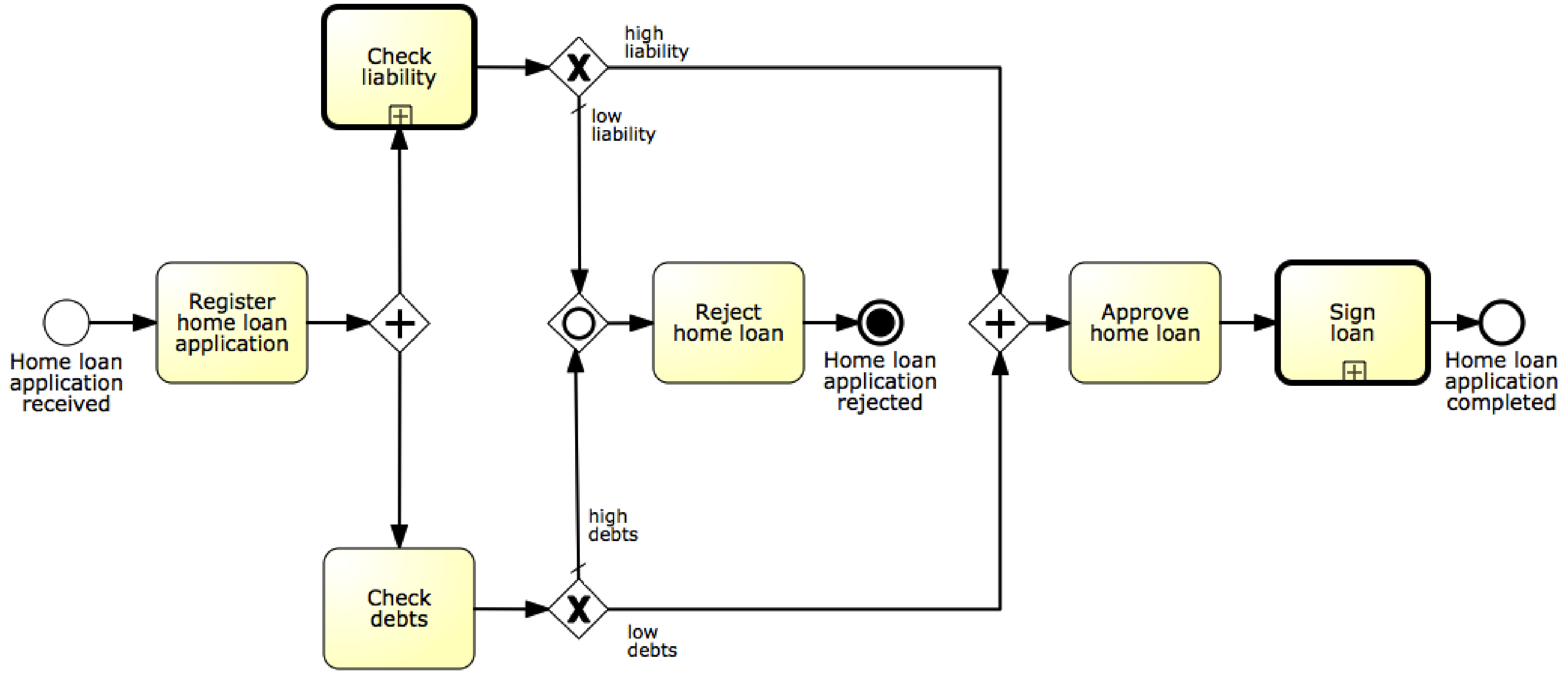
The simplest form of exception is to notify that there is an exception (negative outcome)

This can be done via the Terminate end event: it forces the whole process to *abort* (“wipes off” all tokens left behind, if any)



Example: terminate event

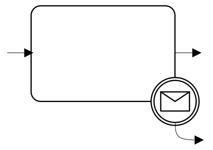
Abort the process by removing all tokens…


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

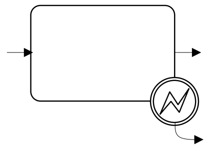
Exception handling – boundary events

Handling exceptions often involves stopping a sub-process and performing a special activity

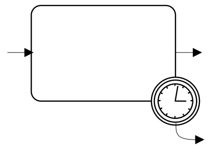
Types of exceptions for an activity (task/sub-process) in BPMN:



External: something goes wrong outside the process, and the execution of the current activity must be interrupted. Handled with the Message event



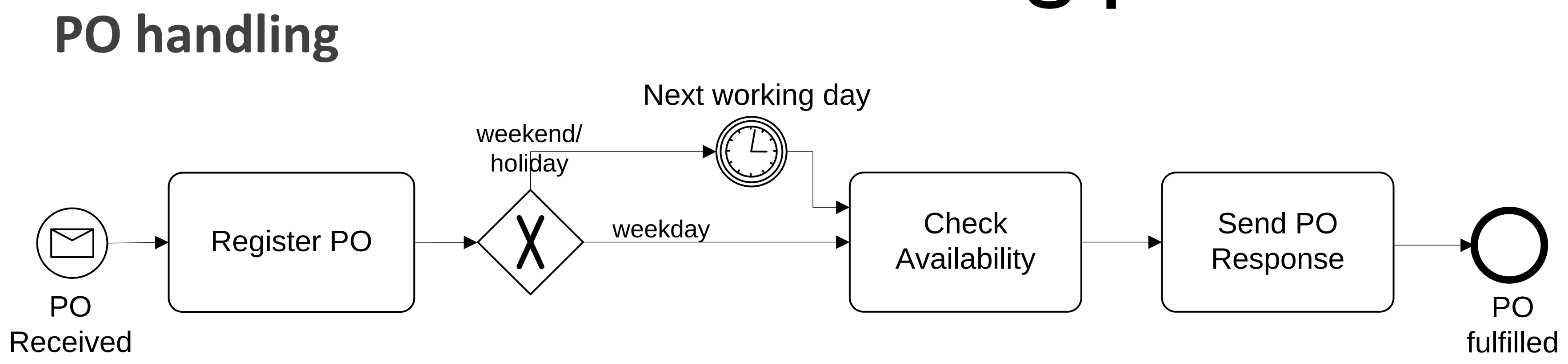
Internal: something goes wrong inside an activity, whose execution must thus be interrupted. Handled with the Error event



Timeout: an activity takes too long and must be interrupted. Handled with the Timer event

All these events are catching intermediate events. They stop the enclosing activity and start an exception handling routine called *exception flow*.

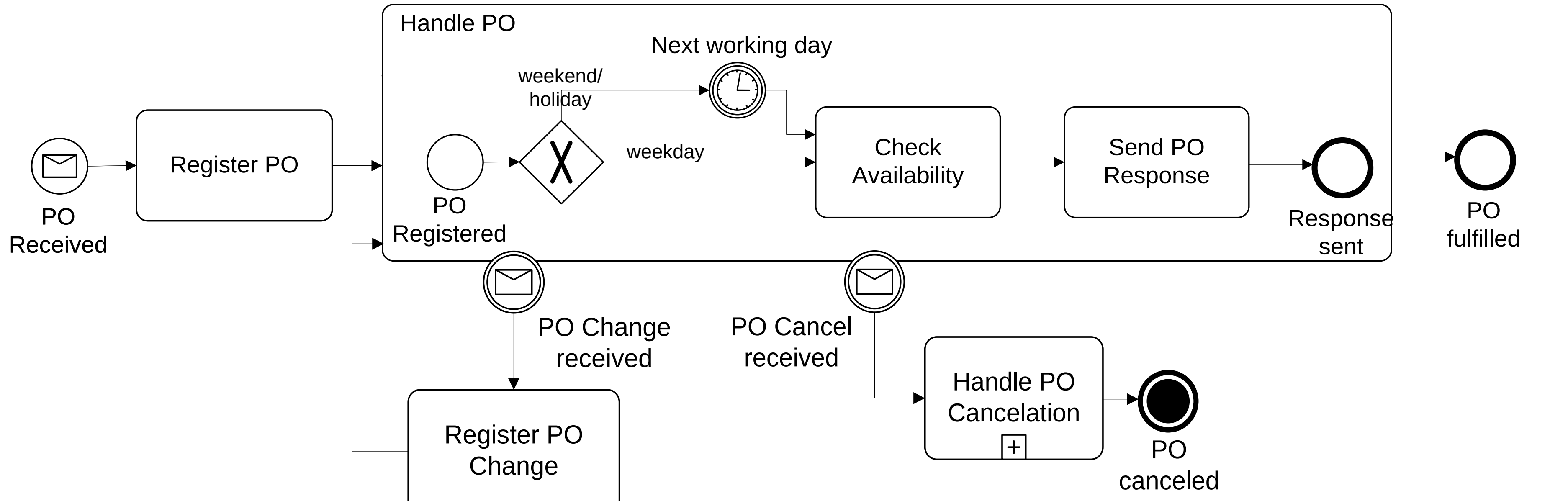
Let’s extend our PO handling process



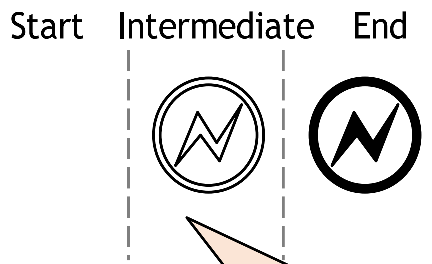
A PO change request may be received anytime after the PO is registered. This request includes a change in quantity or line items. When such a request is received, any processing related to the PO must be stopped. The PO change request is then registered. Thereafter, the process proceeds as it would do after a normal PO is registered. Further, if the customer sends a PO cancelation request after the PO registration, the PO processing must be stopped and the cancelation request must be handled.

Solution: exception handling

PO handling



Internal exception: error event



Error Event – Indicates an error: the "end" version generates an error event while the "catching intermediate" version consumes it when attached to the boundary of an activity

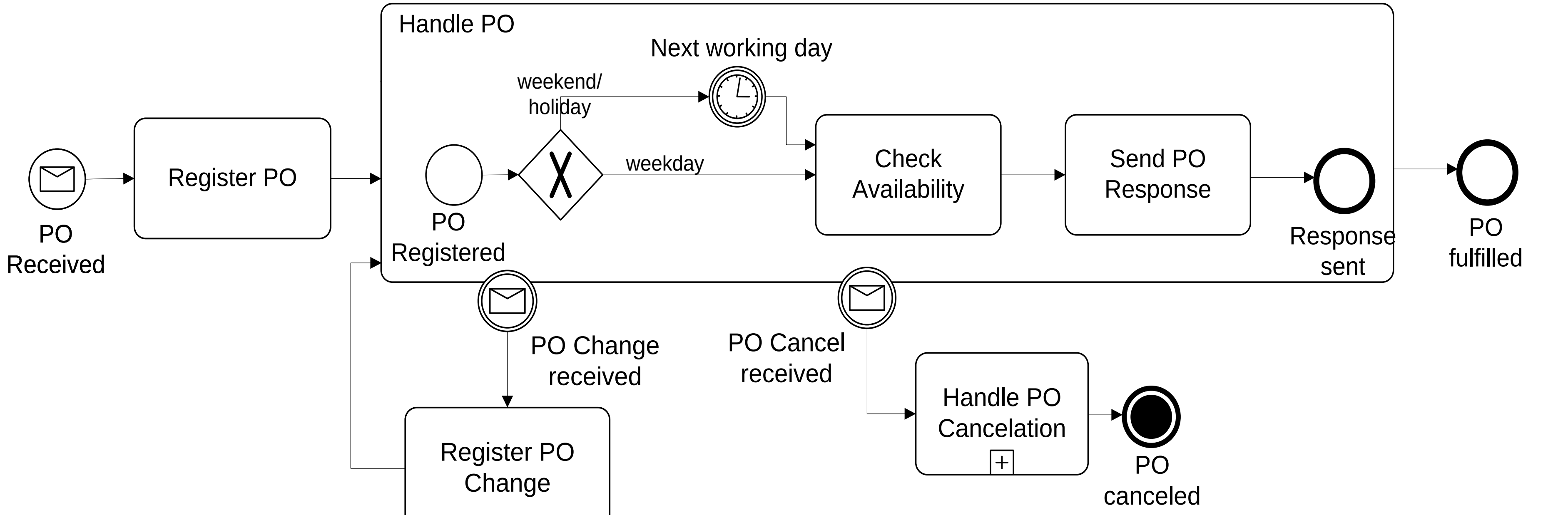
Must be attached to the activity's boundary

```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to be sure you have adequate disk space. If a driver is
identified in the Stop message, disable the driver or check
with the manufacturer for driver updates. Try changing video
adapters.

Check with your hardware vendor for any BIOS updates. Disable
BIOS memory options such as caching or shadowing. If you need
to use Safe Mode to remove or disable components, restart your
computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.

Technical information:

*** STOP: 0x0000007E (0xC0000005,0x46069F2E,0xF79B0AC4,0xF79B07C0)


Beginning dump of physical memory
Physical memory dump complete.
Contact your system administrator or technical support group for further
assistance.
```

Example: internal exception

PO handling

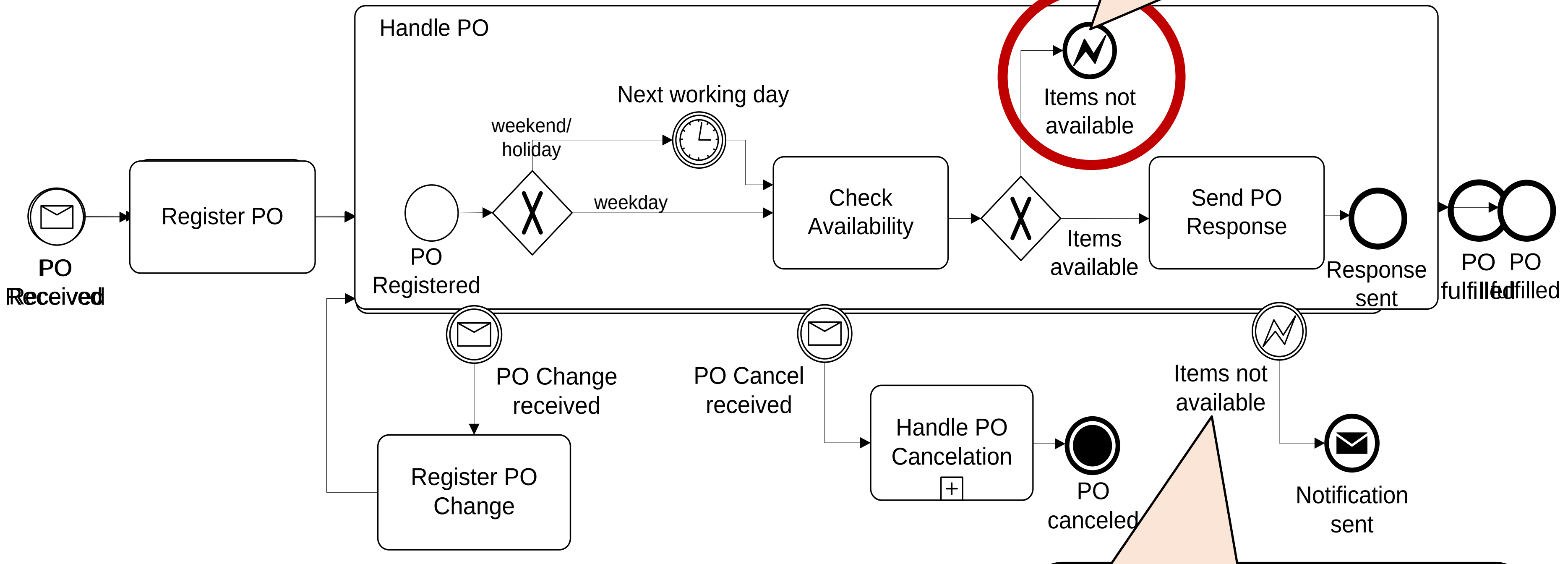
Consider again our "PO Handling process" example with the following extension: if an item is not available, any processing related to the PO must be stopped. Thereafter, the client needs to be notified that the PO cannot be further processed.



Solution: internal exception

PO handling

Throwing and catching error events must have the **same** label



Must catch an error event thrown from **within** the same activity

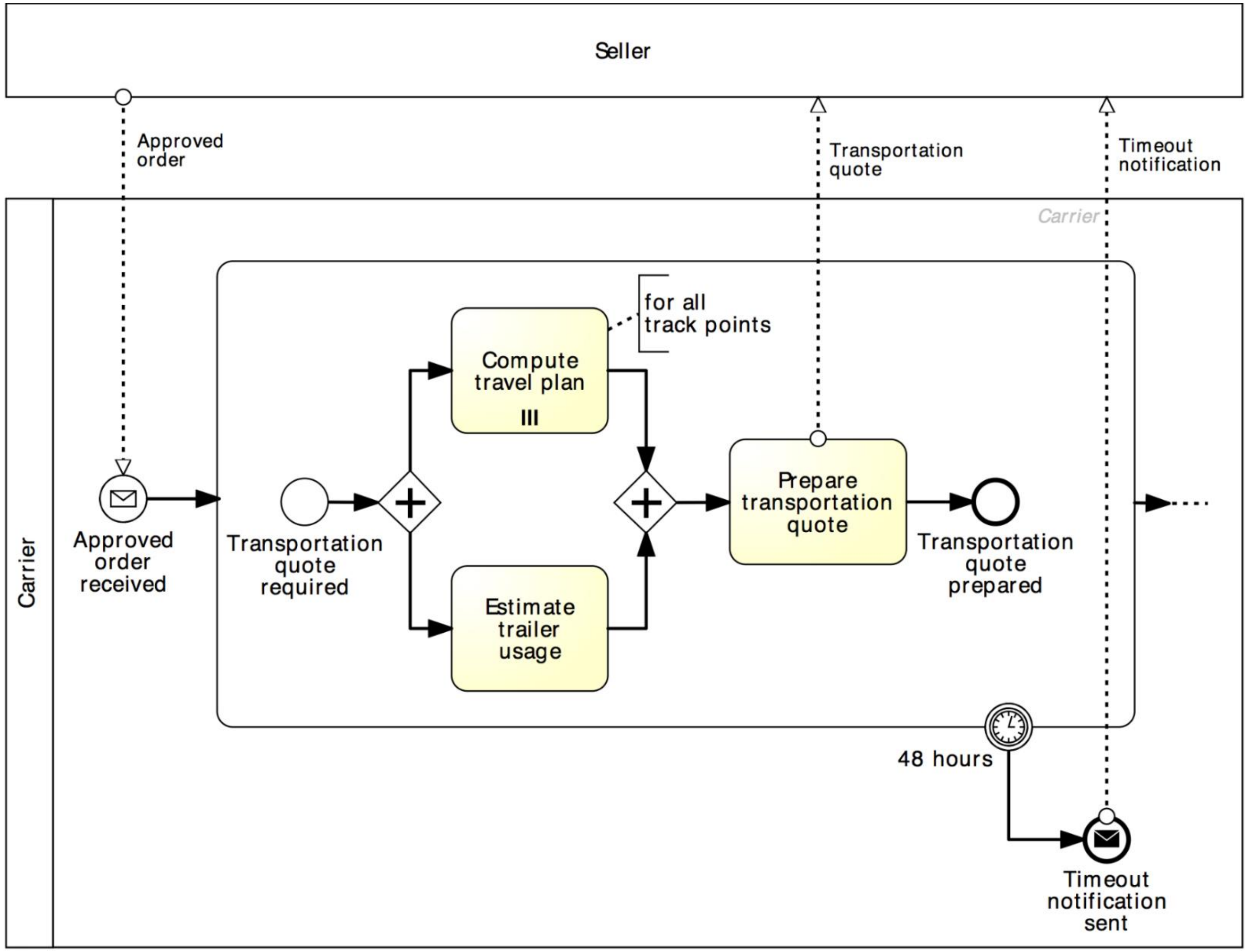
Example: activity timeout

Order-to-transportation quote

Once a wholesale order has been confirmed, the supplier transmits this order to the carrier for the preparation of the transportation quote. In order to prepare the quote, the carrier needs to compute the route plan (including all track points that need to be traversed during the travel) and estimate the trailer usage.

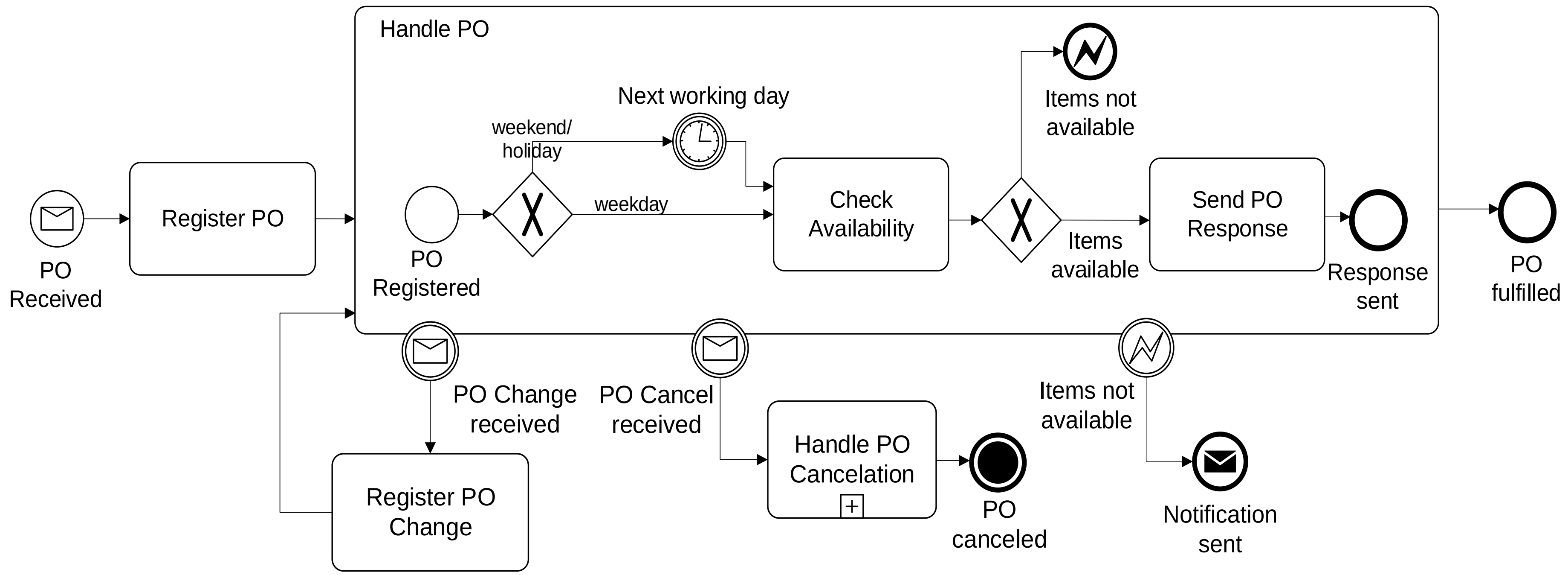
By contract, wholesale orders have to be dispatched within four days from the receipt of the order. This implies that transportation quotes have to be prepared <u>within 48 hours from the receipt of the order</u> to remain within the terms of the contract.

Solution: activity timeout



More on the PO handling example…

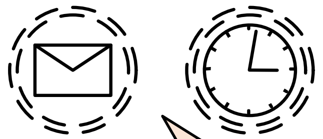
The customer may send a request for address change after the PO registration. When such a request is received, it is registered, without further action.



Non-interrupting boundary events

Sometimes we may need to trigger an activity **in parallel** to the normal flow, i.e. without interrupting the normal flow.

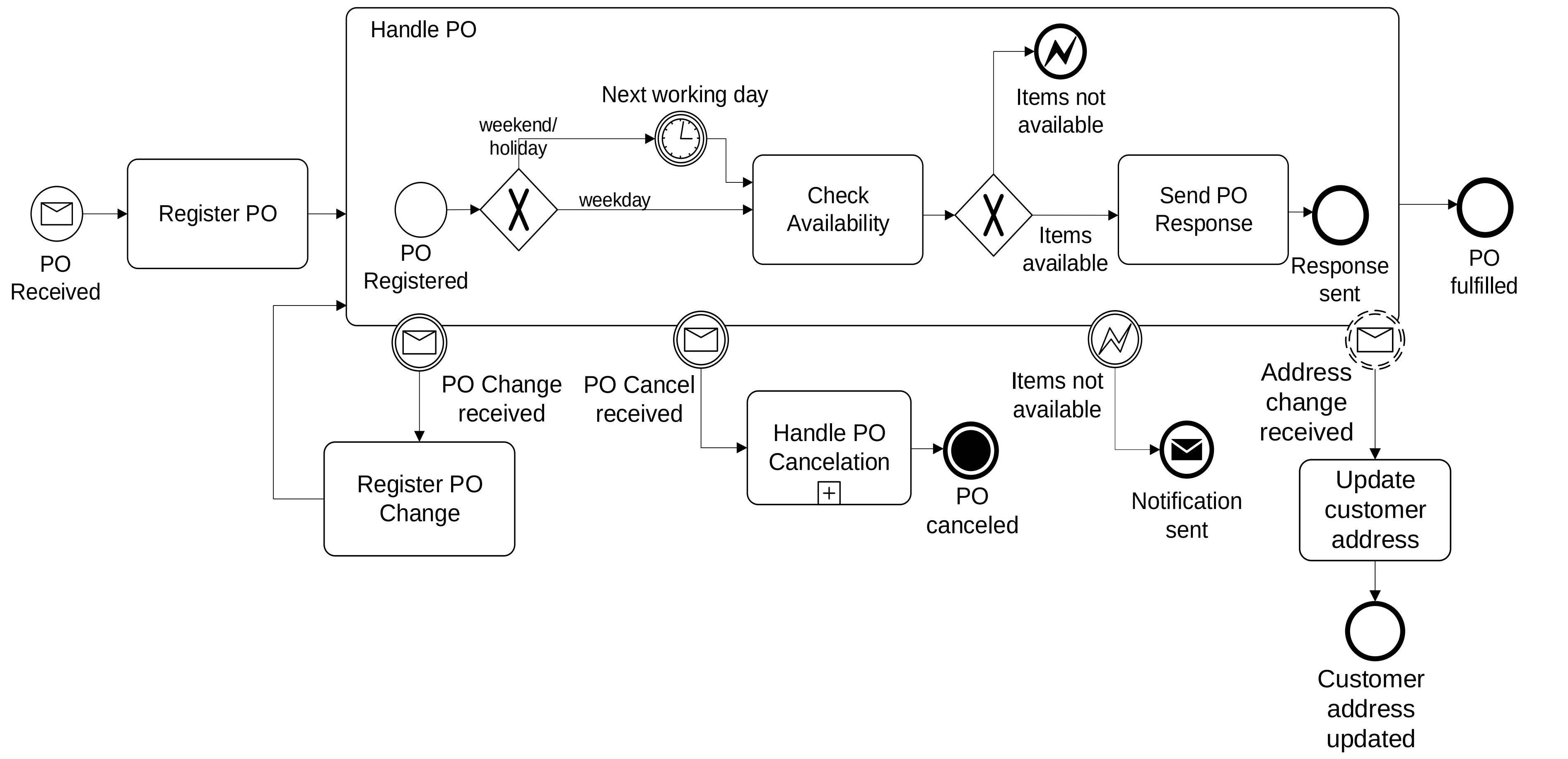
This can be achieved by using *non-interrupting* boundary events



Must be attached to the activity’s boundary

Solution: non-interrupting boundary events

PO handling



Exercise 4.9 (simplified)

Model the following fragment of a process

The routine for logging into an Internet bank account starts once the credentials entered from the user have been retrieved. First, the username is validated. If the username is not valid, the routine is interrupted and the invalid username is logged. If the username is valid, the number of password trials is set to zero. Then, the password is validated. If this is not valid, the counter for the number of trials is incremented and if lower than three, the user is asked to enter the password again. If the number of failed attempts reaches three times, the routine is interrupted and the account is frozen. Moreover, the username and password validation may be interrupted should the validation server not be available. In these cases, the procedure is interrupted after notifying the user to try again later. At any time during the log in routine, the customer may close the web page, resulting in the interruption of the routine.

Summary

- In this lecture we have learned about:
  - Repetition markers: loop marker and parallel multi-instance marker
  - Events: timer, message and error events
  - Event-based choice gateway
  - Boundary events: interrupting and non-interrupting
  - Error events (throw and catch)

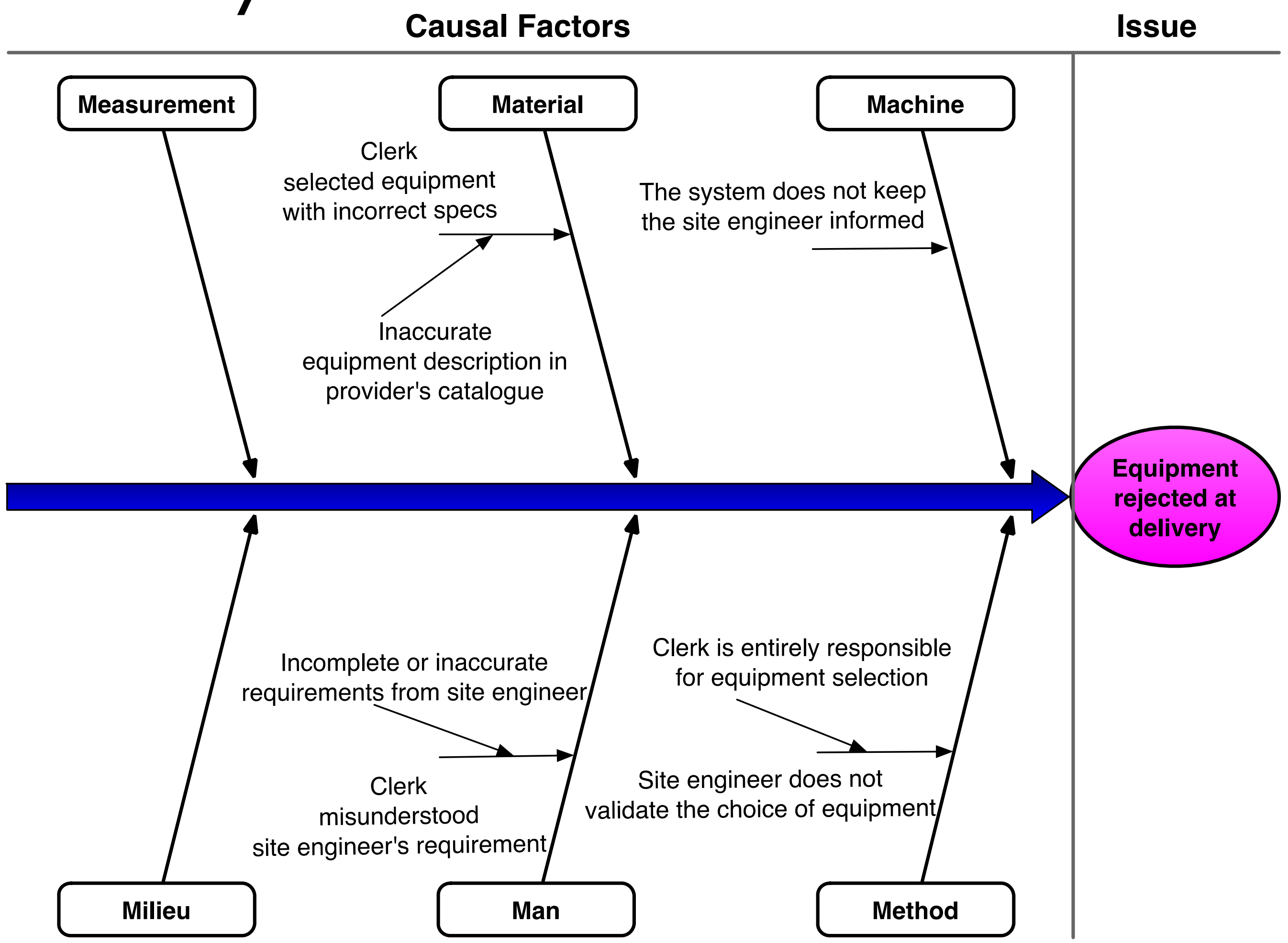
And once I’ve got a model, what’s next?

Process analysis techniques:

- Added-value and waste analysis
- Root-cause analysis
- Flow Analysis

Next Week

Qualitative Process Analysis


Causal Factors
Issue
Measurement
Material
Machine
Clerk selected equipment with incorrect specs
Inaccurate equipment description in provider's catalogue
The system does not keep the site engineer informed
Equipment rejected at delivery
Incomplete or inaccurate requirements from site engineer
Clerk misunderstood site engineer's requirement
Clerk is entirely responsible for equipment selection
Site engineer does not validate the choice of equipment
Milieu
Man
Method

Acknowledgements

- The content notes for this lecture feature content borrowed with or without modification from the following sources:
  - "Source: M. Dumas, M. La Rosa, J. Mendling and H. Reijers, *Fundamentals of Business Process Management*, 2nd edition, Springer, 2018".
  - Chapter 4