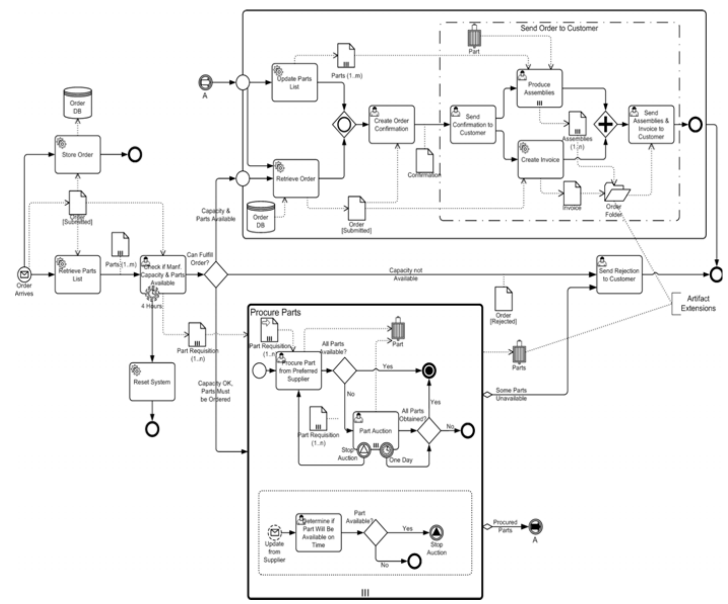
Business Processes Modelling

MPB (6 cfu, 295AA)

Roberto Bruni
http://www.di.unipi.it/~bruni

22 - Business process modelling notation

Object



We overview BPMN and their analysis
based on Petri nets

Ch.4.7, 5.7 of Business Process Management: Concepts, Languages, Architectures
Ch.3, 4 of Fundamental of Business Process Management. M. Dumas et al.

Standardisation

The development of BMPN is an important step in

reducing the fragmentation that existed
with myriad of process modelling tools and notations

exploiting experiences with many divergent proposals to
consolidate the best ideas

supporting the adoption of **inter-operable**
business process management systems

Business Process Management Initiative

August 2000

The **Business Process Management Initiative**
was an **independent organization** devoted to

the development of **open specifications**

for the management of **e-Business processes**
that span multiple applications, corporate departments, and business partners, behind the firewall and over the Internet

The membership of the BPMI Notation Working Group represents a large segment of the BP modelling community

BMI-DTF

June 2005

The Business Process Management Initiative (BPMI.org) and the Object Management Group™ (OMG™) decided to merge their activities on **Business Process Management** (**BPM**) to provide thought leadership and industry standards for this vital and growing industry.

The combined group has named itself the **Business Modeling & Integration Domain Task Force** (**BMI -DTF**)

Business process diagram

BPMN defines a standard for
Business Process Diagrams (**BPD**)

based on **flowcharting technique**
tailored to graphical models of business process operations

Four basic categories of elements:
Swimlanes
Flow objects
Artefacts
Connecting objects

BPMN 1.0 (2004/06)

Main goal:

provide a **notation** that is **readily understandable by all** business users

from the **business analysts** who create initial drafts of the processes

to the **technical developers** responsible for implementing the technology that will perform those processes

to the **business people** who will manage those processes

BPMN Versioning

BPMN 1.0 approved 2006
BPMN 1.1 approved 2007
BPMN 1.2 approved 2009

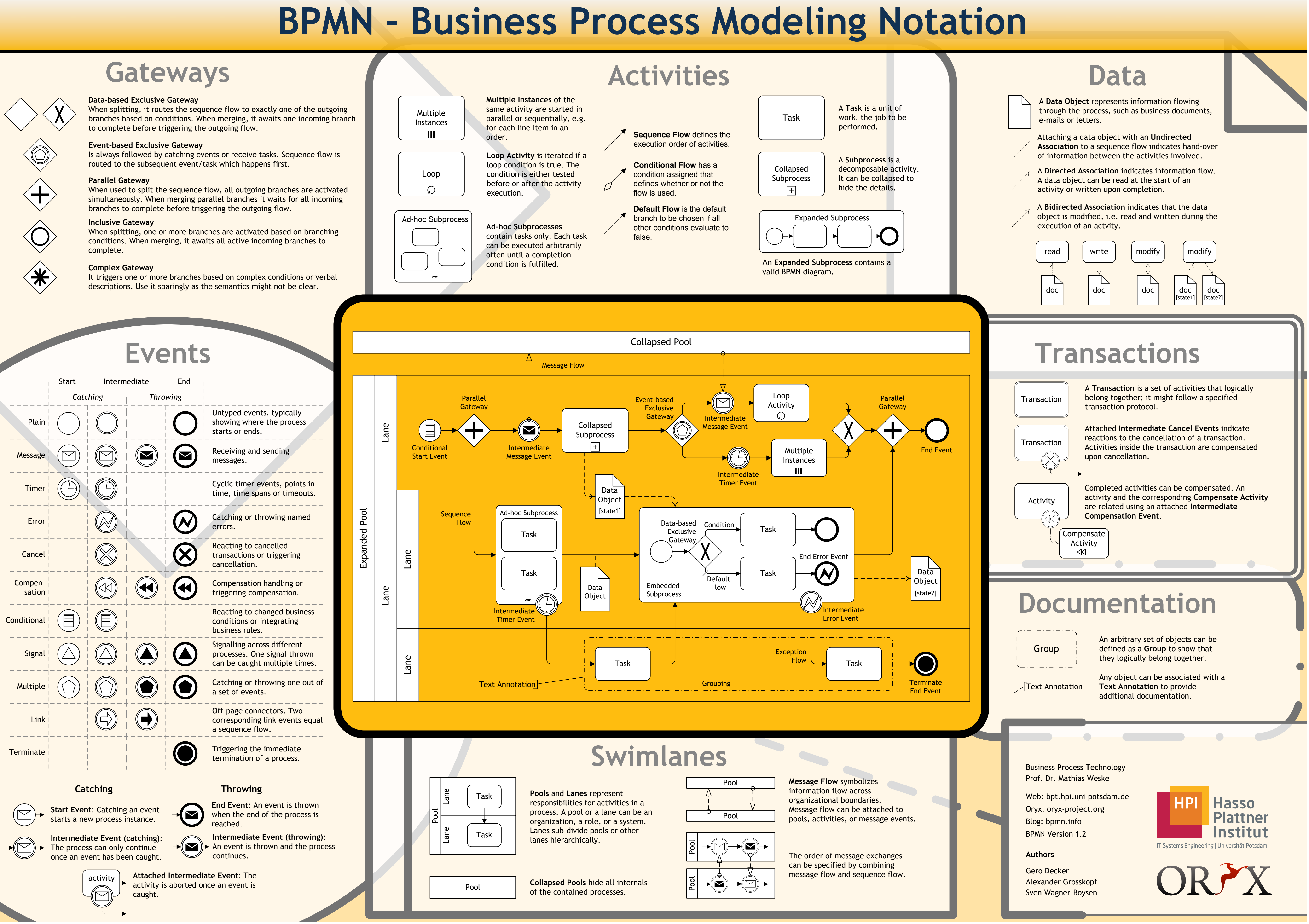
BPMN 2.0 Beta 1 proposed 2009
BPMN 2.0 Beta 2 proposed 2010
BPMN 2.0 Final delivered 2011

Disclaim

Formal rigor and conciseness are not primary concerns
(over 100 symbols,
shorthands and alternative constructs are often available)

The large number of object types
and their continuous evolution
makes it hard to define mappings
and to prove their consistency under all contexts

Inconsistencies and ambiguities in BPMN standard
are **present but hard to detect**

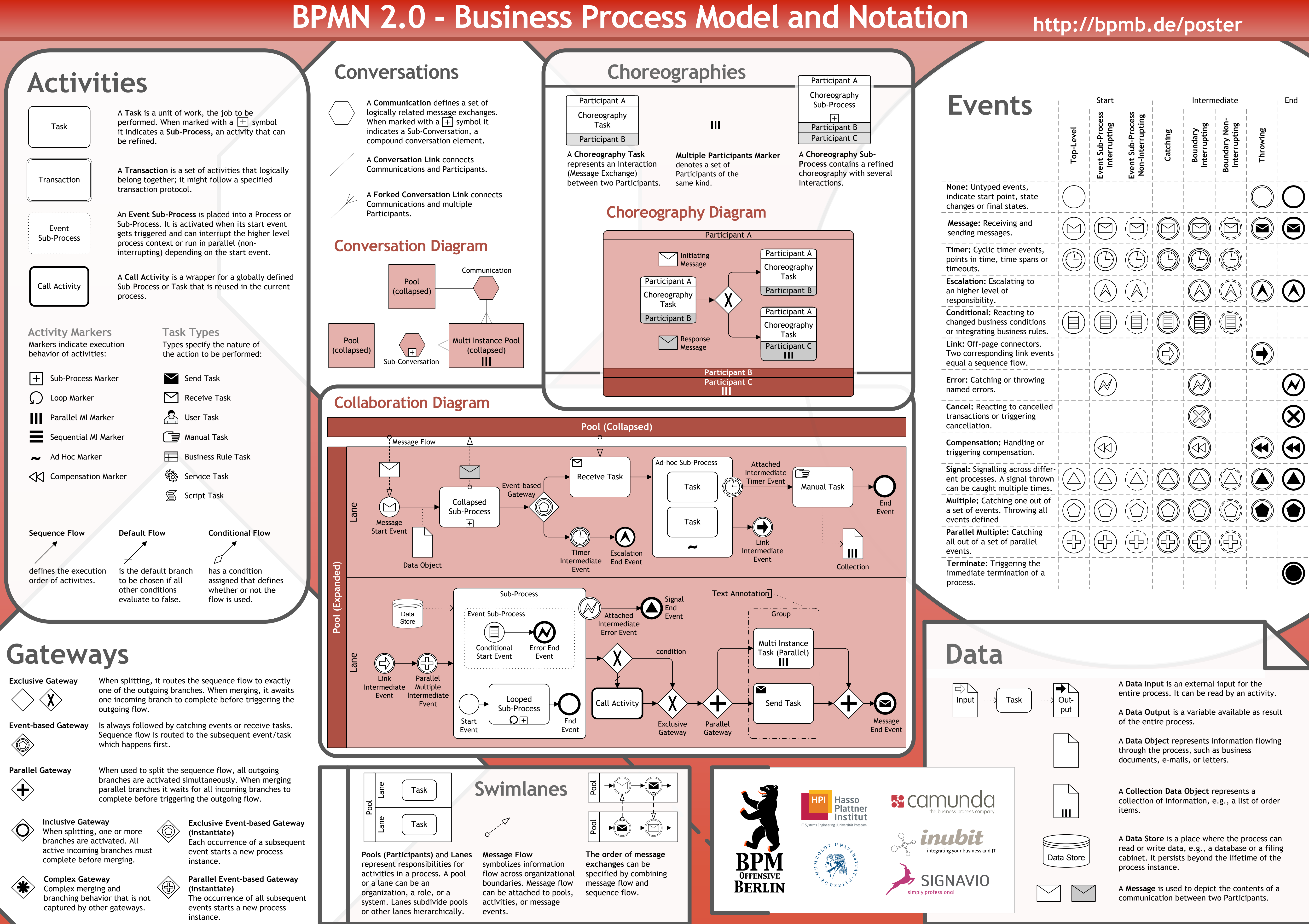


BPMN 1.0 poster

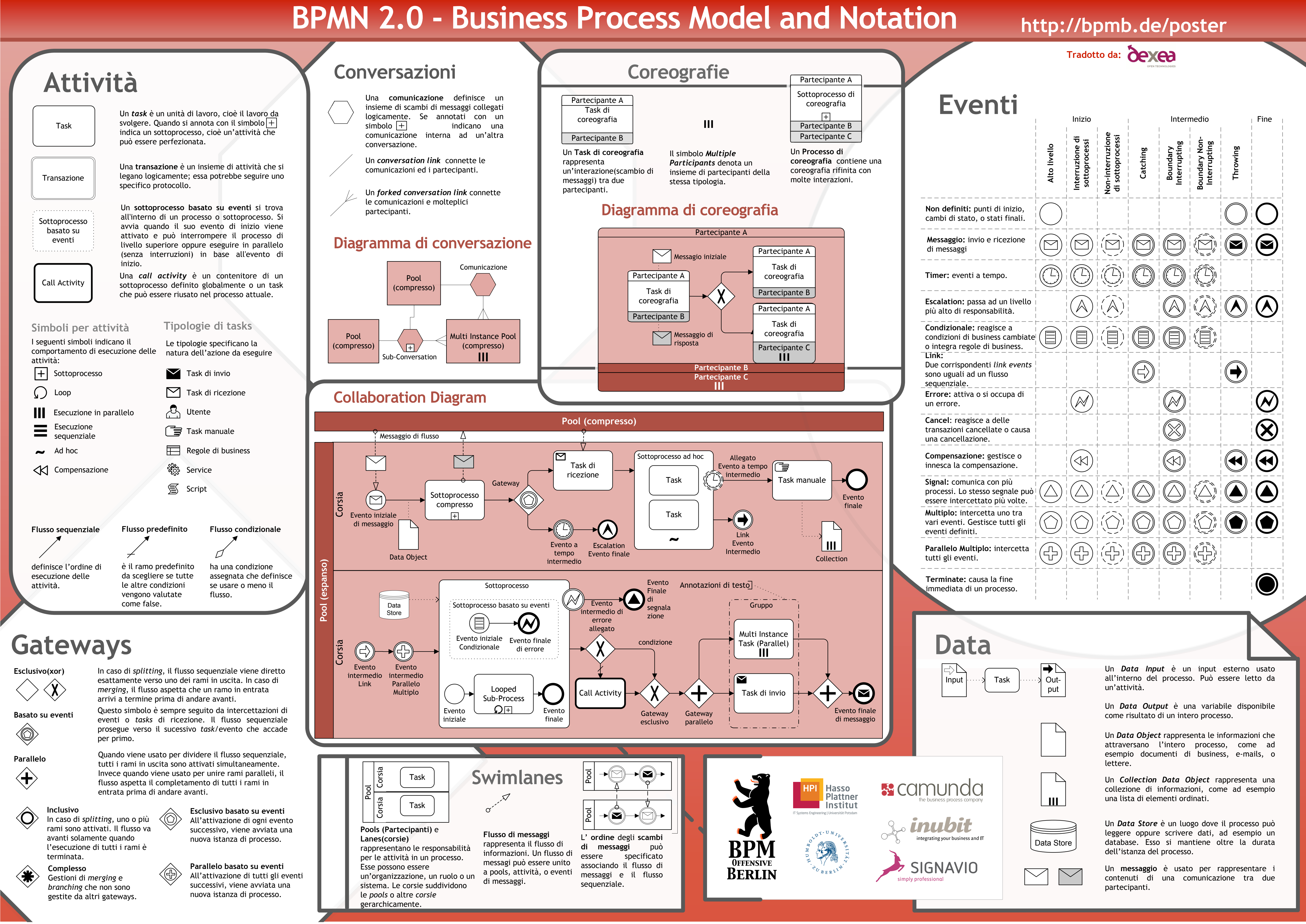
BPMN 2.0 vs 1.0

Updated (new markers):
Tasks/SubProcesses
Events
Gateways
Artefacts

Added:
Choreographies
Full metamodel
XML Serialization
Diagram Interchange
BPMN Execution Semantics (verbal)



BPMN 2.0 poster



BPMN 2.0 poster (in Italian)

BPMN 2.0 (2009/11) FAQ

What is BPMN?

BPMN is a graphical notation that depicts the steps (end to end flow) in a business process.

The notation has been specifically designed to coordinate the sequence of processes and the messages that flow between different participants in a related set of activities.

BPMN 2.0 (2009/11) FAQ

Why is BPMN important?

The world of business processes has changed dramatically over the past few years. Processes can be coordinated from behind, within and over organizations boundaries. A business process now spans multiple participants and coordination can be complex.

Until BPMN, there has not been a standard modelling technique developed that addresses these issues.
BPMN provides users with a **royalty free notation**.

This will benefit users in a similar manner in which UML standardised the world of software engineering.
There will be training courses, books and a body of knowledge that users can access in order to better implement a business process.

BPMN 2.0 (2009/11)
FAQ

Who is BPMN targeted at?

BPMN is targeted at a **high level for business users** and at a **lower level for process implementers**.

The former should be able to easily read and understand a BPMN diagram.
The latter should be able to adorn a BPMN diagram with further details in order to represent the process in a physical implementation.

BPMN is targeted at users, vendors and service providers that need to communicate business processes in a standard manner.

BPMN 2.0 (2009/11) FAQ

Will there be a major rewrite?

Not for 2 or 3 years…

(2018 and still no revision is planned)

Strong points of BPMN

Simplicity: A small set of basic symbols

Extensibility: many decorations available
(new ones can be added in the future)

Graphical design: **intuitive**

Generality: **orchestration** + **choreography**

Tool availability: **exchange format**

Weaknesses of BPMN

over 100 graphical elements

verbose description (500 pages)

difficult to learn comprehensively:
different reading of the same diagram are possible

different BPMN vendors implement the execution of
BPMN diagrams in different ways (and for different subsets)

BPMN basics: Swimlanes (pools, lanes)

Swimlanes

Many process modelling methodologies utilise the concept of a **swimlane** as a mechanism to **organise activities into separate visual categories** in order to illustrate different capabilities or responsibilities

BPMN supports two main swimlane objects:

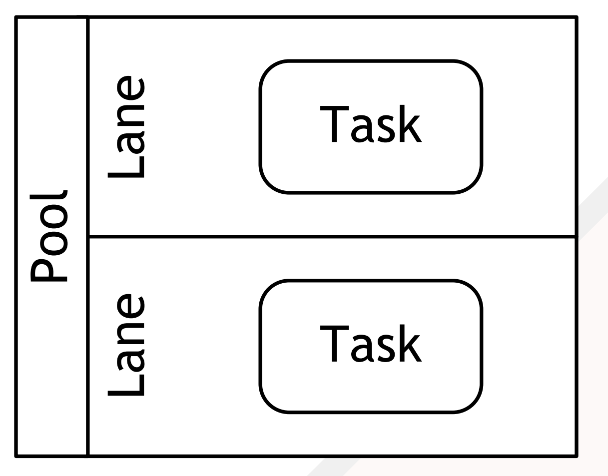
Pool

Lane

Pools and Lanes

A **pool** represents a participant (or role) in a process
A pool is represented as a rectangle with a name

A **lane** is a hierarchical sub-partition within a pool
that is used to organise and categorise activities



A lane is an inner rectangle to the pool
that extends to the entire length of the pool

Swimlanes

Pools and lanes are used to represent organizations, roles, systems and responsibilities. Examples: 'University', 'Sales division', 'Warehouse', 'ERP system',...

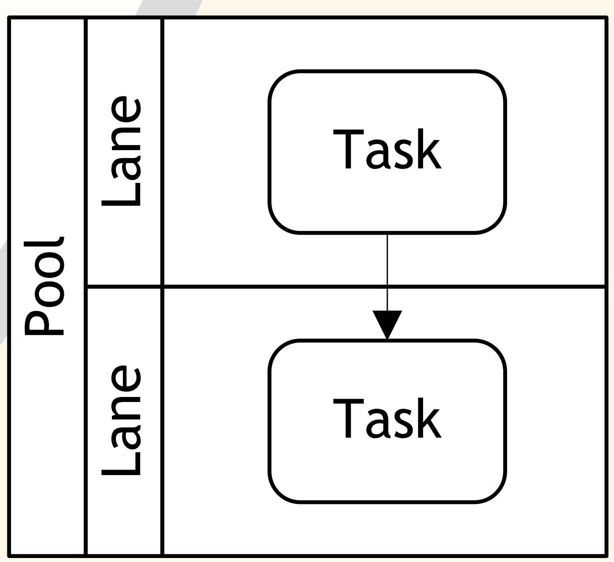
A Pool MUST contain 0 or 1 business process.

A Pool can contain 0 or more lanes.

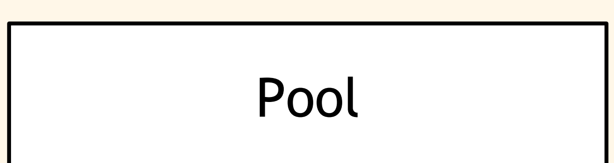
Two pools can only be connected with message flows.

A **Pool** represents a participant in a process. It contains a business process and is used in B2B situations.

A **Lane** is a sub-partition within a pool used to organize and categorize activities.



Pools and **Lanes** represent responsibilities for activities in a process. A pool or a lane can be an organization, a role, or a system. Lanes sub-divide pools or other lanes hierarchically.



Collapsed Pools hide all internals of the contained processes.

Naming conventions

Process models:
a noun possibly preceded by an adjective

the label is often obtained by ``nominalizing” the verb that describe the main action in the process (e.g., claim handling, order fulfillment)

Avoid long labels
Articles are often omitted

BPMN basics: Flow Objects (events, activities, gateways)

Flow objects

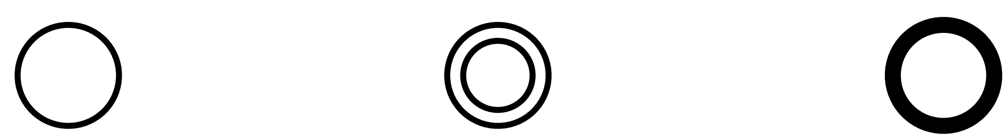
Theory:
fix a small set of core elements
so that modellers do not have to learn and recognise
a large number of different shapes:
Events
Activities
Gateways

Practice:
use different border styles and internal markers
to add many more information
(this way the notation is **extensible**)

Events

An **event** is something that "happens" during the course of a business process

The type of an event is one among:
start, **intermediate**, **end**



An event is represented as a circle
its type depends on the style of the border
(thin, double, thick)

An event can have a cause (**trigger**) or an impact (**result**)
Internal markers denote the trigger or result

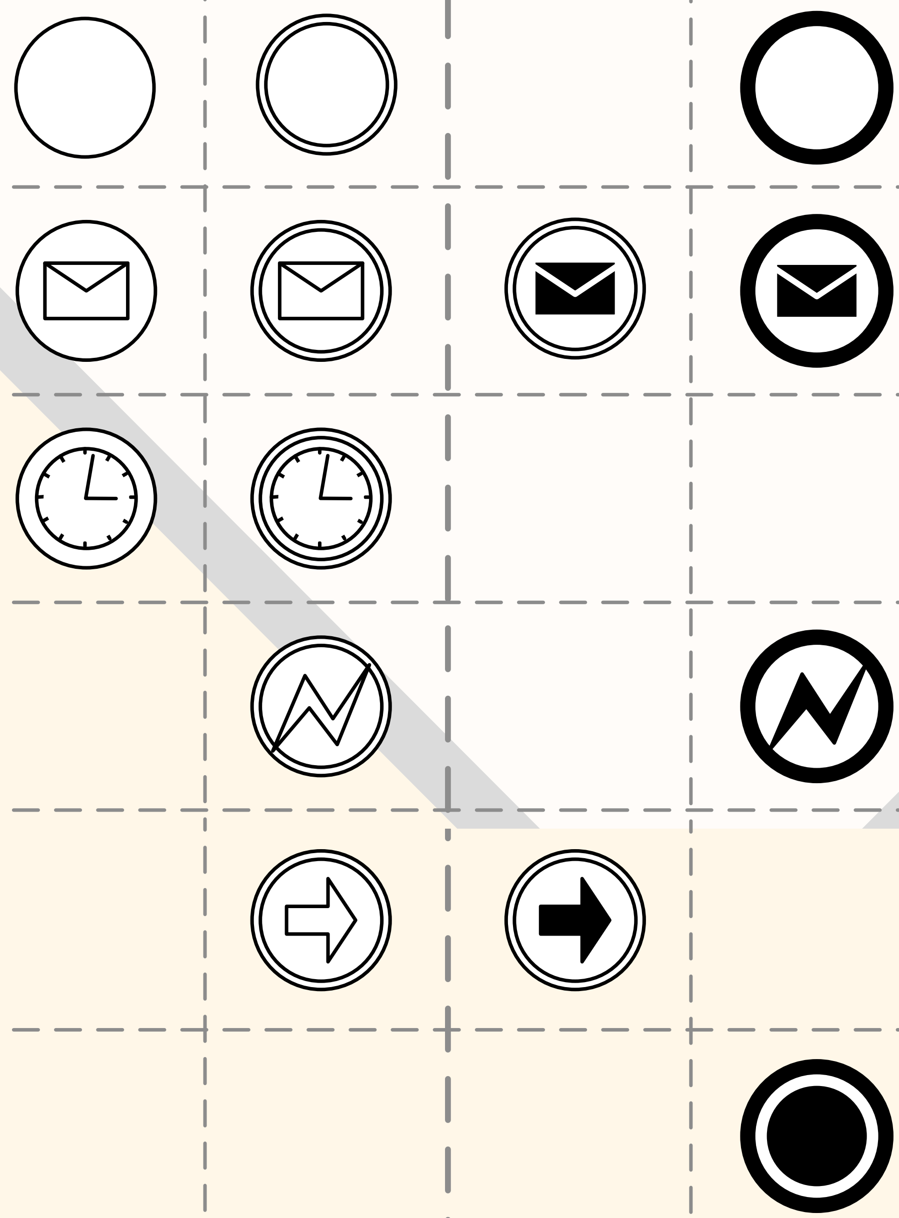
Naming conventions

Events:

the label should begin with a noun and
end with a verb in past participle form
to indicate something that just happened
(e.g., Invoice emitted)

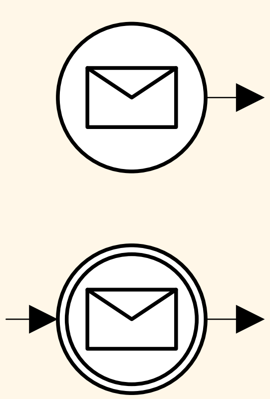
the noun can be preceded by an adjective
(e.g., Urgent order sent)

Avoid long labels
Articles are often omitted



	Start (Catching)	Intermediate (Catching)	Intermediate (Throwing)	End (Throwing)	
Plain					Untyped events, typically showing where the process starts or ends.
Message					Receiving and sending messages.
Timer					Cyclic timer events, points in time, time spans or timeouts.
Error					Catching or throwing named errors.
Link					Off-page connectors. Two corresponding link events equal a sequence flow.
Terminate					Triggering the immediate termination of a process.

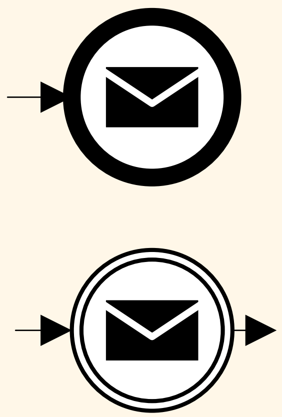
Catching



Start Event: Catching an event starts a new process instance.

Intermediate Event (catching): The process can only continue once an event has been caught.

Throwing



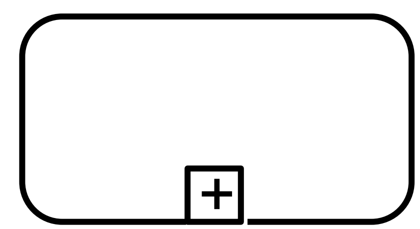
End Event: An event is thrown when the end of the process is reached.

Intermediate Event (throwing): An event is thrown and the process continues.

Activities

An **activity** is some “unit of work” (job) to be done during the course of a business process

An activity can be
atomic (**task**) or compound (**sub-process**)



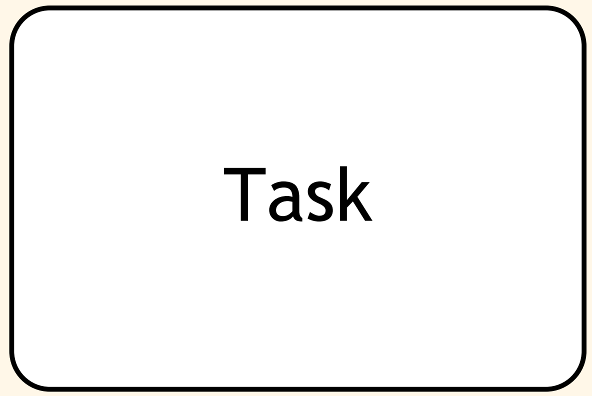
An activity is represented as a rounded box,
Suitable markers are used to indicate
the nature of the action to be performed (**task type**)
and the execution behaviour (**activity marker**)

Sub-processes

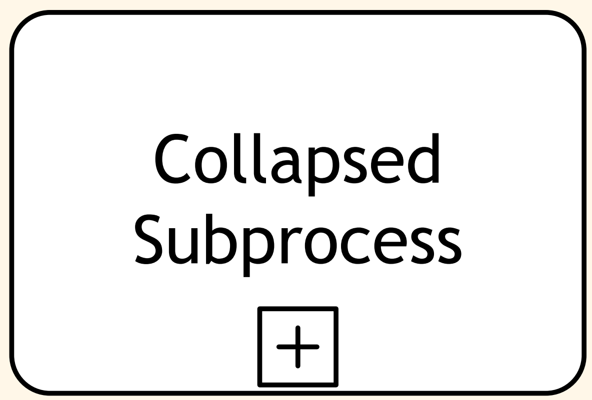
Process models tend to be too large
to be understood at once

Hiding certain parts within sub-processes
we improve readability

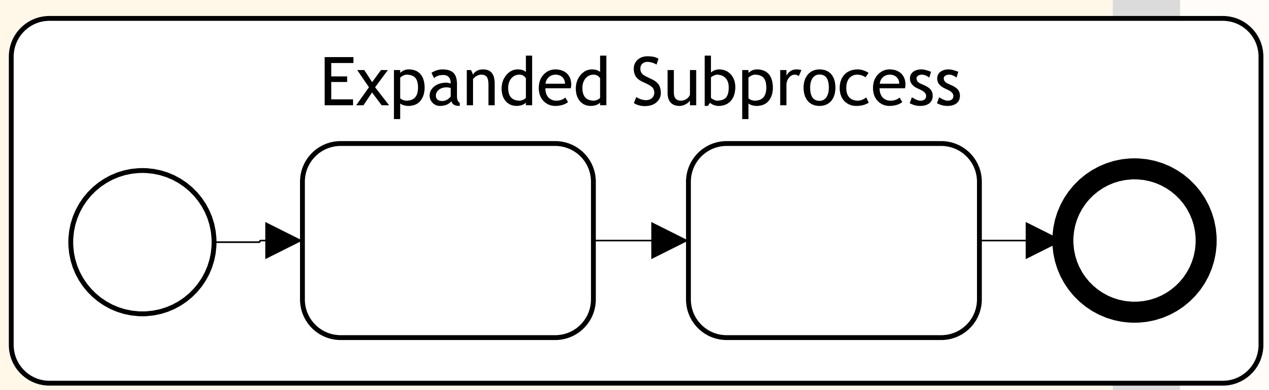
A **sub-process** is a self-contained, composite activity
that can be broken into smaller units of work



A **Task** is a unit of work, the job to be performed.



A **Subprocess** is a decomposable activity. It can be collapsed to hide the details.



An **Expanded Subprocess** contains a valid BPMN diagram.

Activity types and markers

Activity Markers

Markers indicate execution behavior of activities:

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

Task Types

Types specify the nature of the action to be performed:

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

Naming conventions

Activities:
verb in the imperative form followed by a noun
(e.g., Approve order)

the noun can be preceded by an adjective
(e.g., Issue driver license)

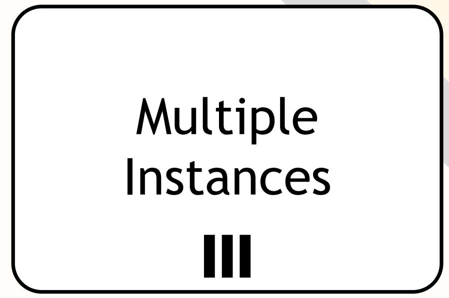
the verb may be followed by a complement
(e.g., Renew driver license via offline agencies)

Avoid long labels
Articles are often omitted

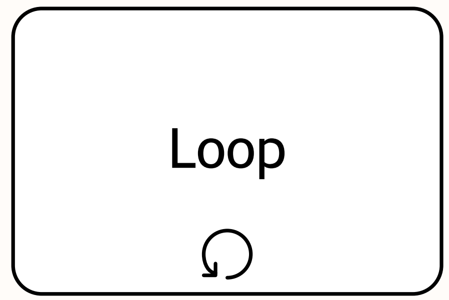
Events vs Activities

Events are instantaneous

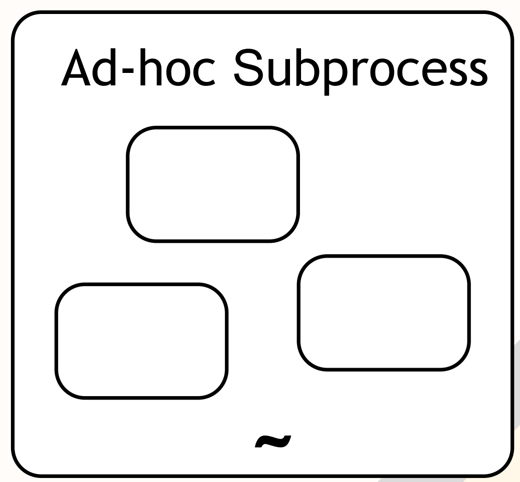
Activities take time (have a duration)



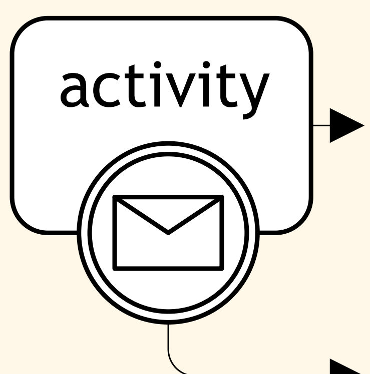
Multiple Instances of the same activity are started in parallel or sequentially, e.g. for each line item in an order.



Loop Activity is iterated if a loop condition is true. The condition is either tested before or after the activity execution.



Ad-hoc Subprocesses contain tasks only. Each task can be executed arbitrarily often until a completion condition is fulfilled.

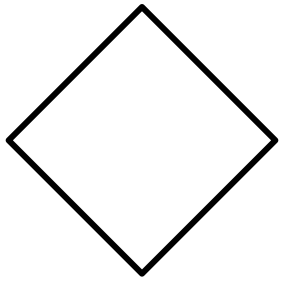


Attached Intermediate Event: The activity is aborted once an event is caught.

A **Call Activity** is a wrapper for a globally defined Sub-Process or Task that is reused in the current process.

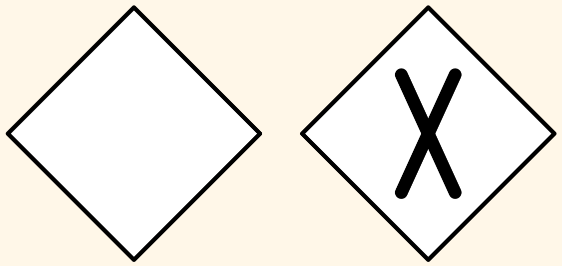
Gateways

A **gateway** is used to control the splitting and joining of paths in the sequence flow (conditional, fork, wait)

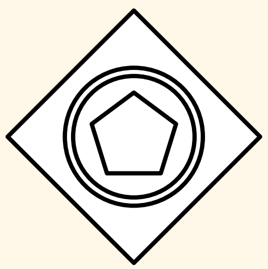


A gateway is represented as a diamond shape
Suitable markers are used to indicate the nature of behaviour control

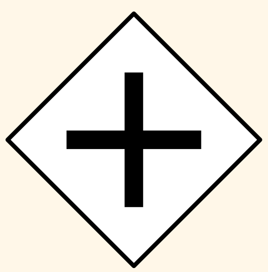
Gateway markers



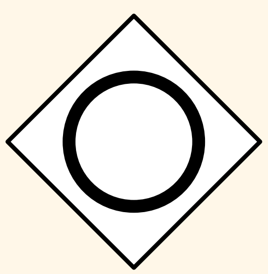
Data-based Exclusive Gateway
When splitting, it routes the sequence flow to exactly one of the outgoing branches based on conditions. When merging, it awaits one incoming branch to complete before triggering the outgoing flow.



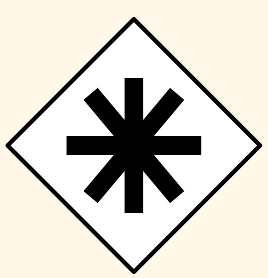
Event-based Exclusive Gateway
Is always followed by catching events or receive tasks. Sequence flow is routed to the subsequent event/task which happens first.



Parallel Gateway
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
When splitting, one or more branches are activated based on branching conditions. When merging, it awaits all active incoming branches to complete.



Complex Gateway
It triggers one or more branches based on complex conditions or verbal descriptions. Use it sparingly as the semantics might not be clear.

BPMN basics: Artefacts (data-objects, groups, text annotations)

Artefacts

BPMN is designed to allow modellers and modelling tools some flexibility in extending the basic notation

Any number of artefacts can be added to a diagram as appropriate for the specific context of the business process being modelled

BPMN includes three pre-defined types of artefacts:

Data object

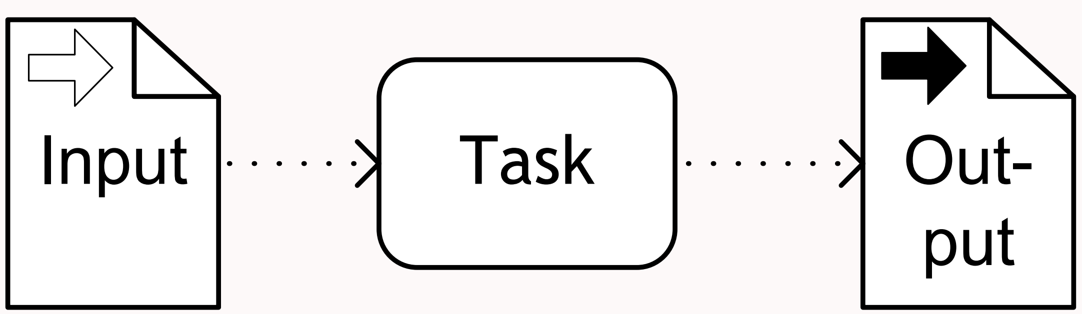
Group

Text annotation

Data object

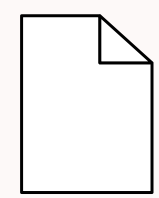
A **data object** specifies the data that are required or produced by an activity

A data object is often represented by the usual file icon

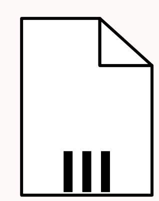


A **Data Input** is an external input for the entire process. It can be read by an activity.

A **Data Output** is a variable available as result of the entire process.

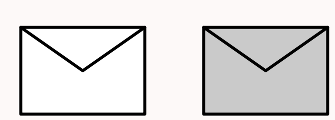


A **Data Object** represents information flowing through the process, such as business documents, e-mails, or letters.



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

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



A **Message** is used to depict the contents of a communication between two Participants.

Group

An arbitrary set of objects can be defined as a **group** to show that they logically belong together

A group is represented by rounded corner rectangles with dashed lines

Annotation

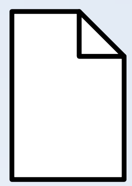
Any object can be associated with a **text annotation** to provide any additional information and documentation that can be needed

A text annotation is represented as a dotted-line call-out

Artefacts

Artefacts are used to provide additional information about the process. If required, modellers and modelling tools are free to add new artefacts.

<u>Examples of data objects: 'A letter', 'Email message', 'XML document', 'Confirmation',...</u>

Set of standardized artefacts		
Data object	 [state]	Data objects provide information about what activities are required to be triggered and/or what they produce. They are considered as Artefacts because they do not have any direct effect on the Sequence Flow or Message Flow of the Process. The state of the data object should also be set.
Group		Grouping can be used for documentation or analysis purposes. Groups can also be used to identify the activities of a distributed transaction that is shown across Pools. Grouping of activities does not affect the Sequence or Message Flow.
Annotation	[Description	Text Annotations are a mechanism for a modeller to provide additional information for the reader of a BPMN Diagram.

BPMN basics: Connecting objects

(sequence flow, message flow, association)

Connecting objects

The Flow objects are connected together in a diagram to create the basic skeletal structure of a business process

Three connecting objects can be used:

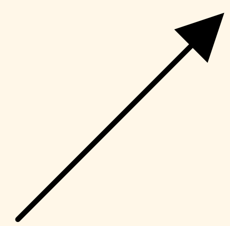
Sequence flow
Message flow
Association

Sequence flow

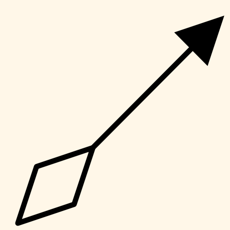
A **sequence flow** is used to show the order in which activities are to be performed

Note: connected objects must **reside in the same pool** (but they can be in different lanes)
the term "control flow" is generally avoided in BPMN

A sequence flow is represented by a solid line with a solid arrowhead

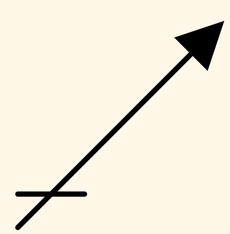


Sequence Flow defines the execution order of activities.



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

read as ``otherwise”

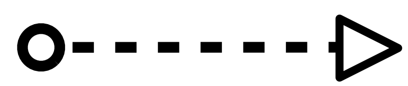


Default Flow is the default branch to be chosen if all other conditions evaluate to false.

Message flow

A **message flow** is used to show the flow
of messages between two separate process participants
(business entities or business roles)
that send and receive them

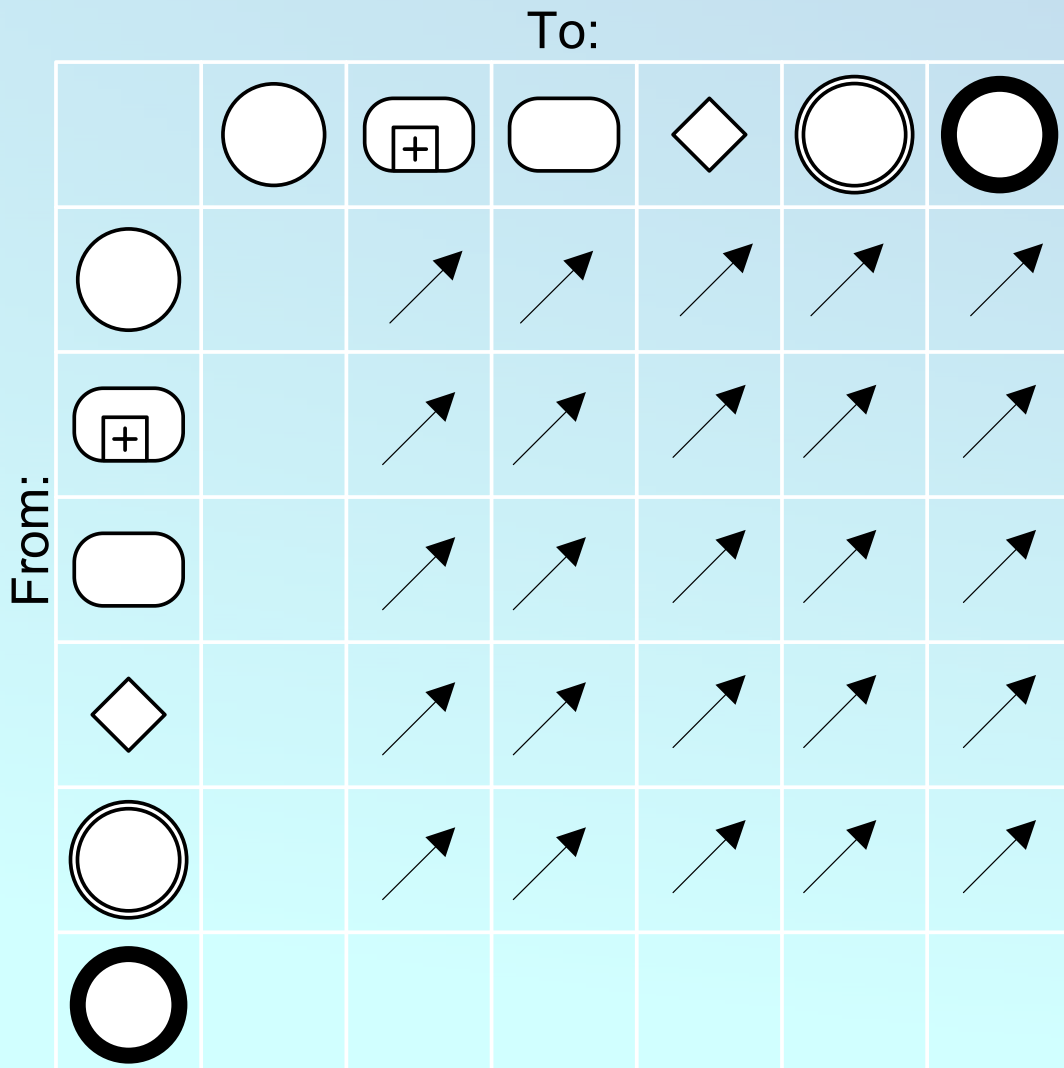
Note: the participants **reside in separate pools**

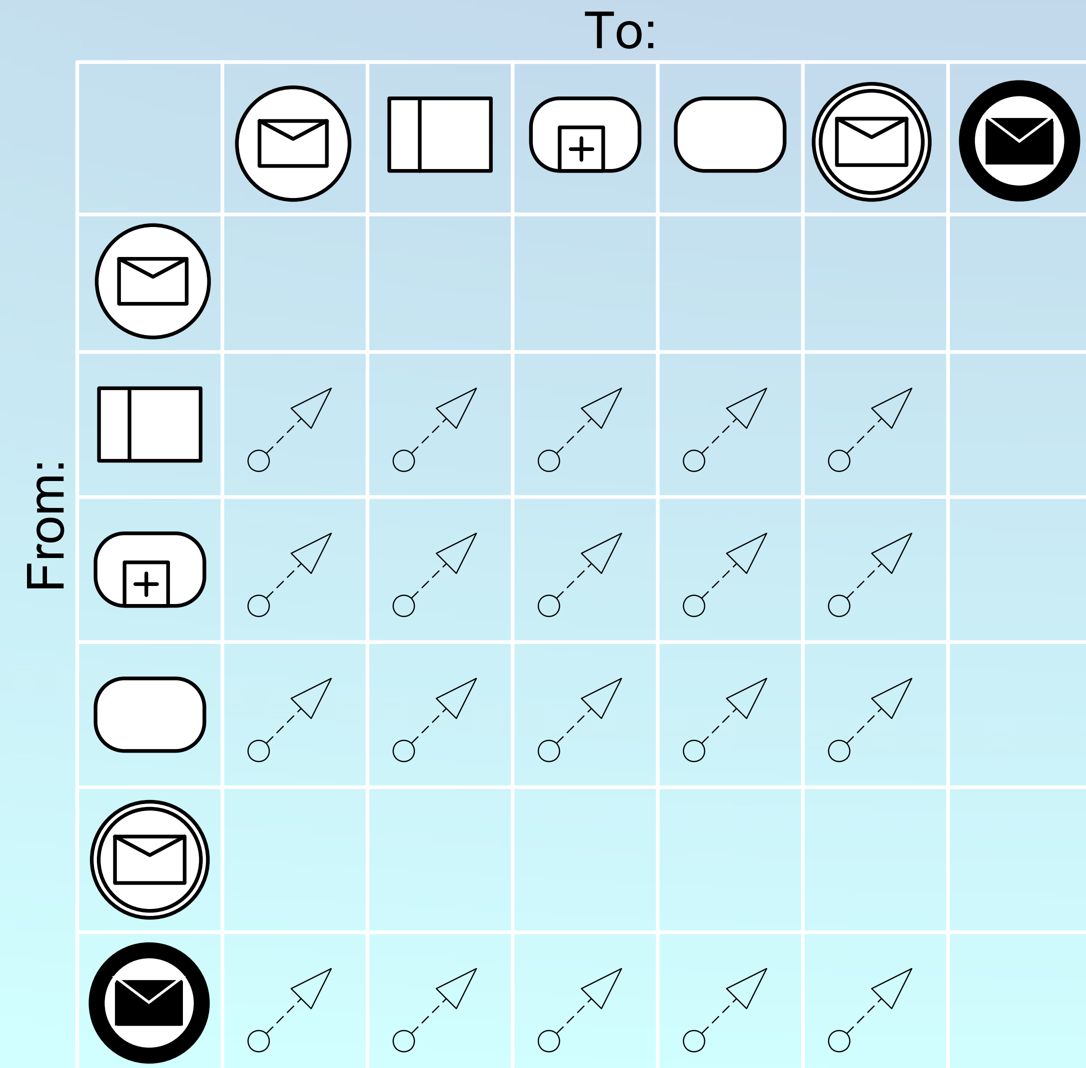


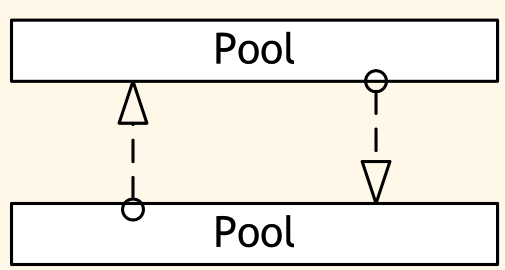
A message flow is represented by
a dashed line with a open arrowheads (see above)

Sequence Flow and Message Flow rules

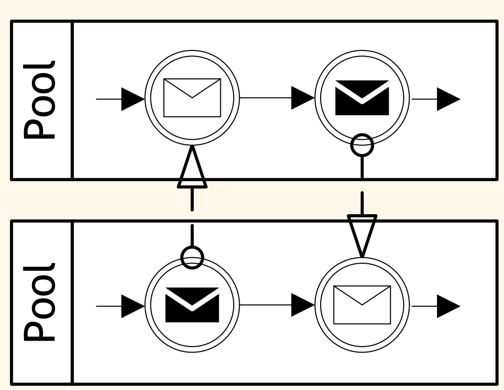
Only objects that can have an incoming and/or outgoing Sequence Flow / Message Flow are shown in the Tables Below.







Message Flow symbolizes information flow across organizational boundaries. Message flow can be attached to pools, activities, or message events.

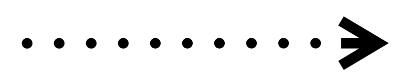


The order of message exchanges can be specified by combining message flow and sequence flow.

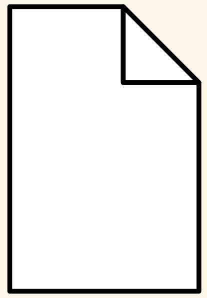
Association

An **association** is used to associate data, text, and other artefacts with flow objects

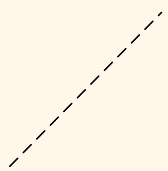
Note: in particular, input and output of activities



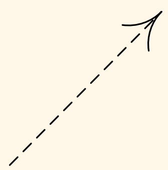
An association is represented by a dotted line with a line arrowhead



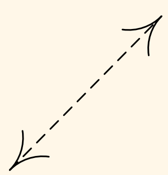
A **Data Object** represents information flowing through the process, such as business documents, e-mails or letters.



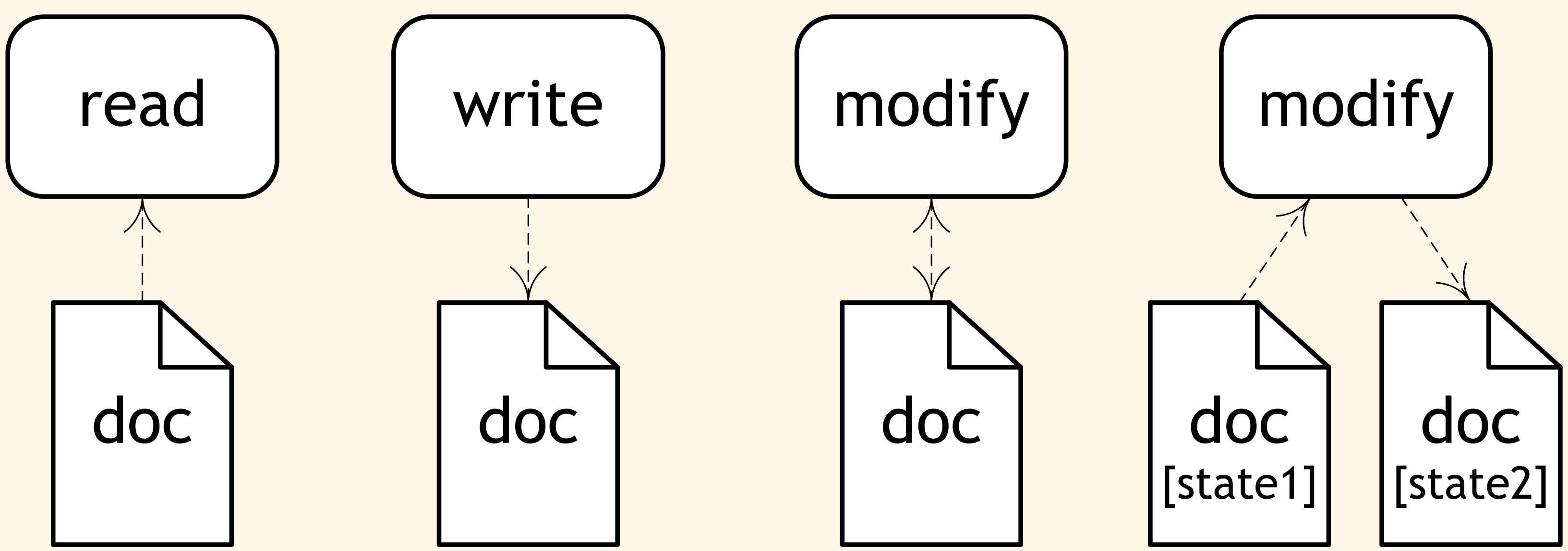
Attaching a data object with an **Undirected Association** to a sequence flow indicates hand-over of information between the activities involved.



A **Directed Association** indicates information flow. A data object can be read at the start of an activity or written upon completion.

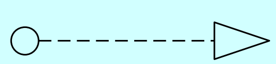
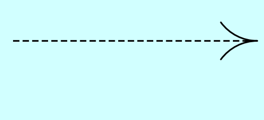


A **Bidirected Association** indicates that the data object is modified, i.e. read and written during the execution of an actvity.



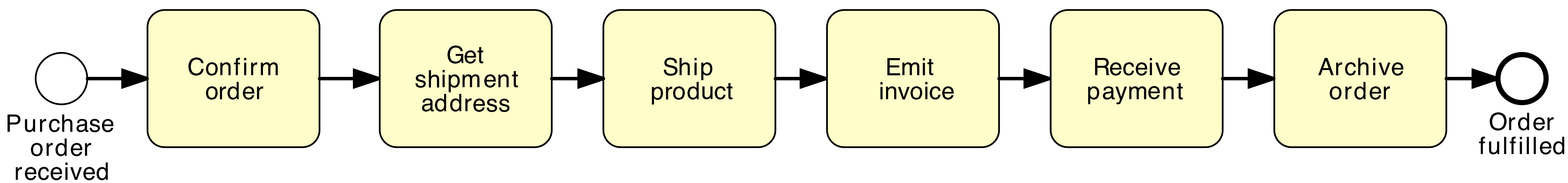
Graphical connecting objects

There are three ways of connecting **Flow objects (Events, Activities, Gateways)** with each other or with other information – using sequence flows, message flows or associations.

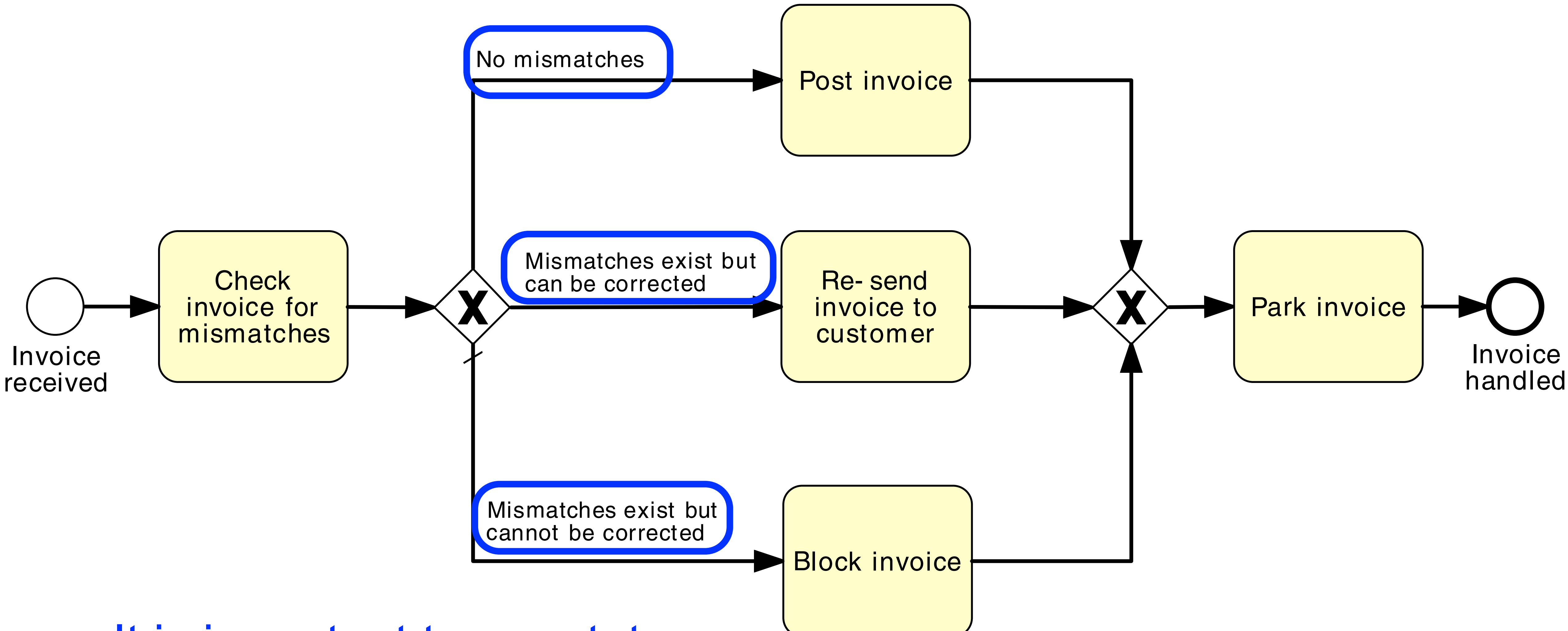
Graphical connecting objects		
Normal sequence flow		A Sequence Flow is used to show the order In which the activities in a process will be performed.
Conditional sequence flow		A Sequence Flow can have condition expressions which are evaluated at runtime to determine whether or not the flow will be used.
Default sequence flow		For Data-Based Exclusive Decisions or Inclusive Decisions, one type of flow is the Default condition flow. This flow will be used only if all other outgoing conditional flows are NOT true at runtime.
Message flow		A Message Flow is used to show the flow of messages between two participants that are prepared to send and receive them. In BPMN, two separate Pools in a Diagram can represent the two participants.
Association		An Association (directed, non-directed) is used to associate information with Flow Objects. Text and graphical non-Flow Objects can be associated with Flow objects.

A few patterns

Sequence: order fulfillment

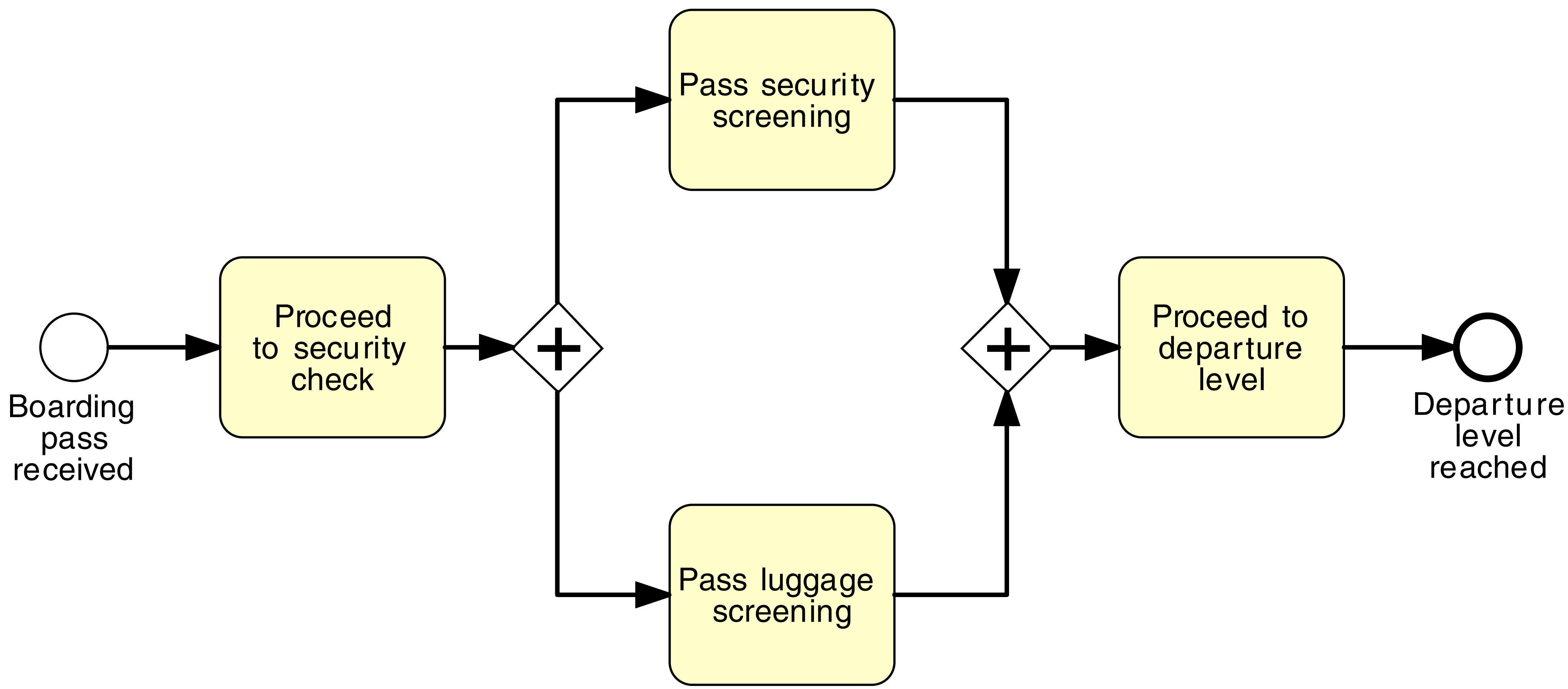


Exclusive decisions: invoice checking process

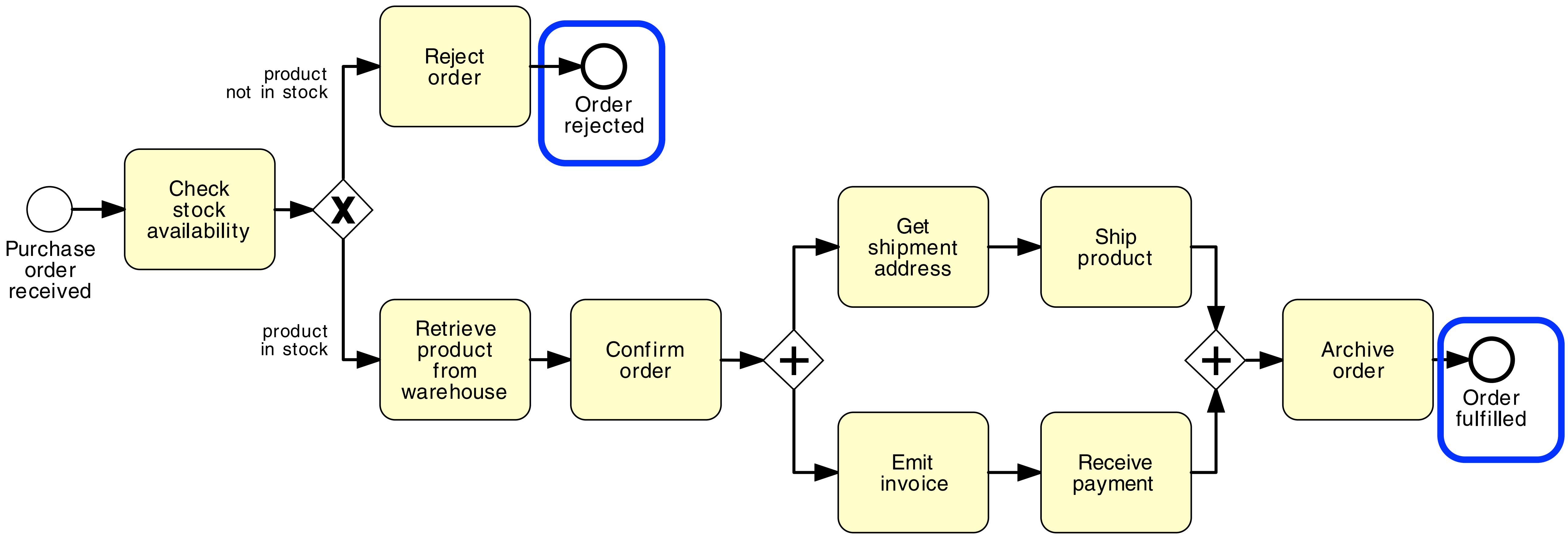


It is important to annotate branches with the conditions under which they are taken

Parallel activities: airport security check



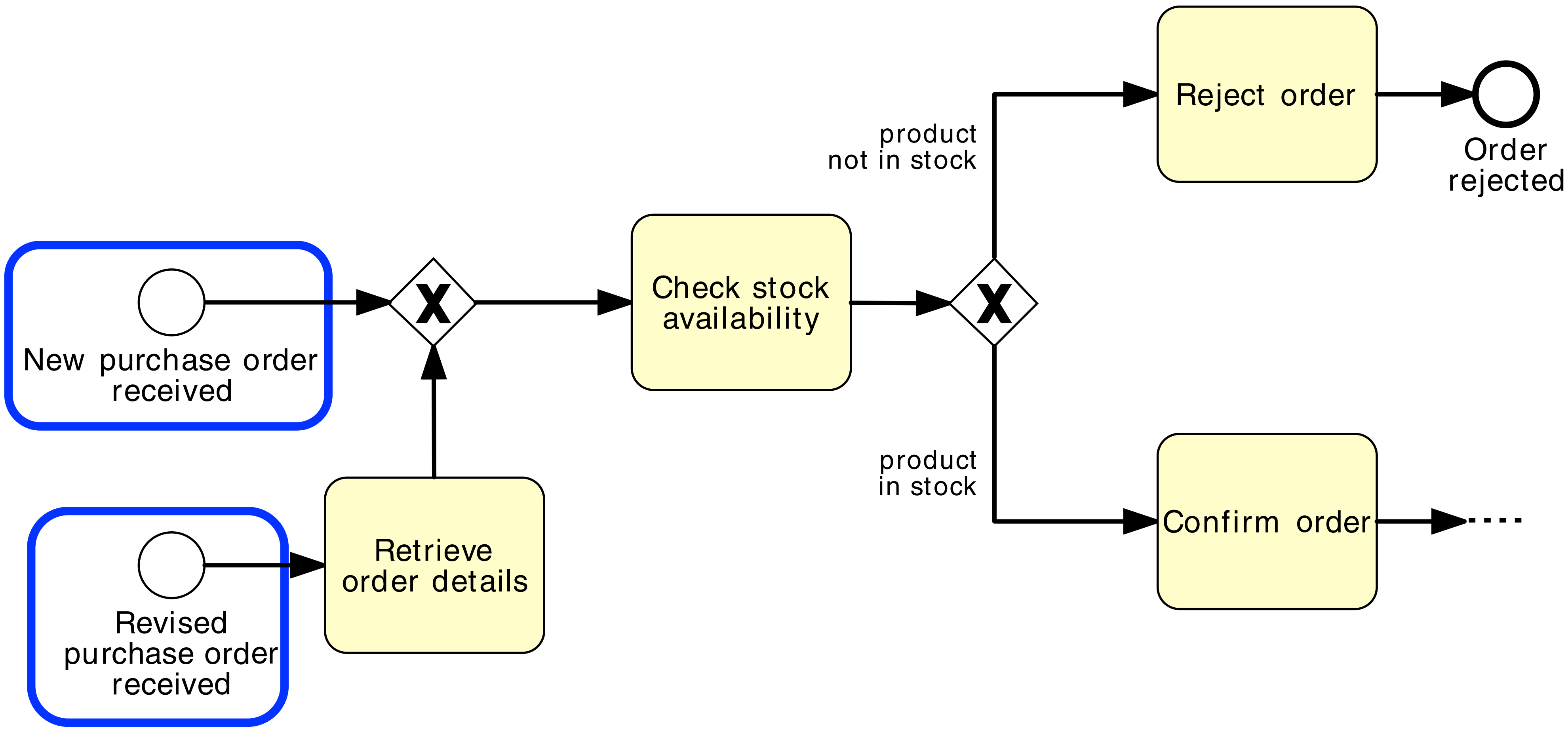
XOR + AND: order fulfillment



Multiple end events are often considered as a convenient notation (they are mutually exclusive in the example)

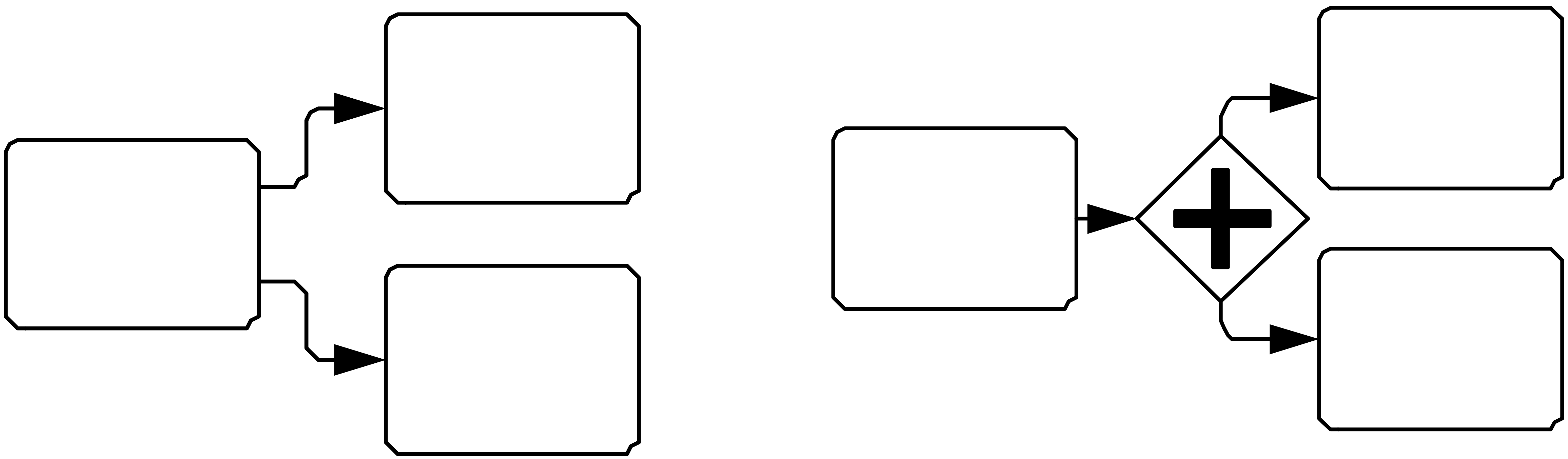
BPMN adopts **implicit termination** semantics: a case ends only when each ``token'' reaches the end

Multiple start events: order fulfillment



Multiple start events are often considered as a convenient notation (they capture mutually exclusive triggers to start a process instance)

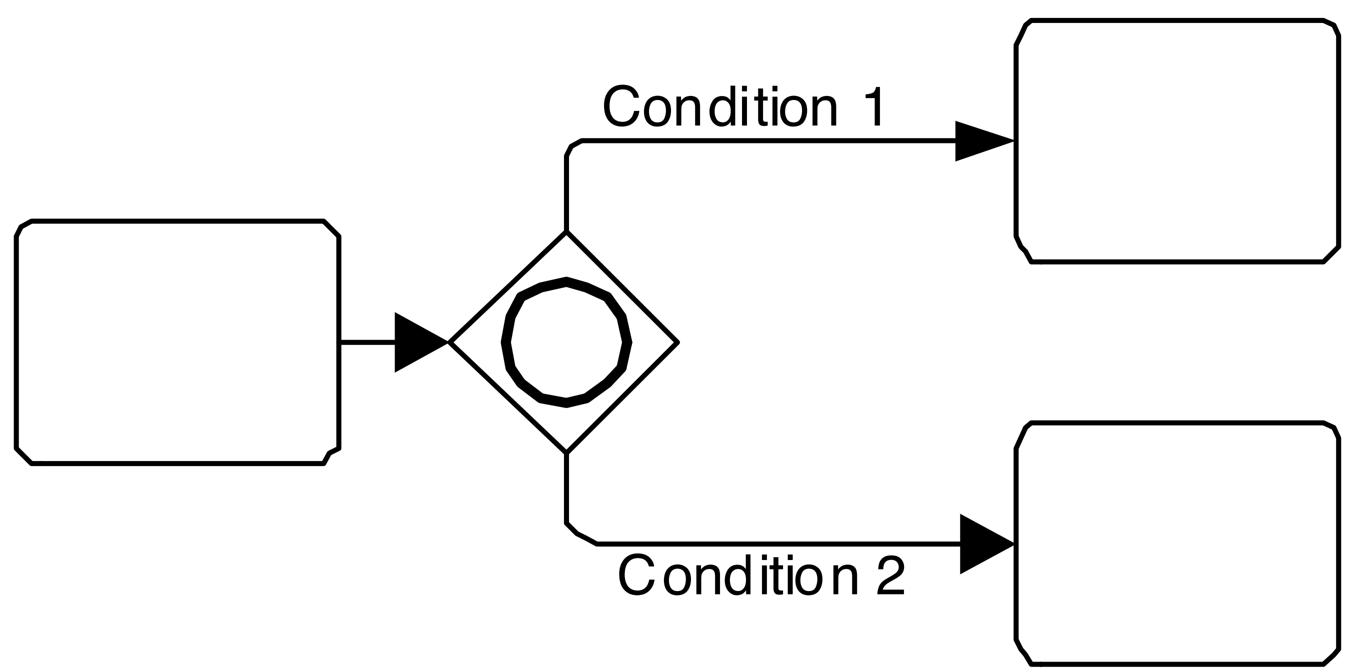
Omitting gateways



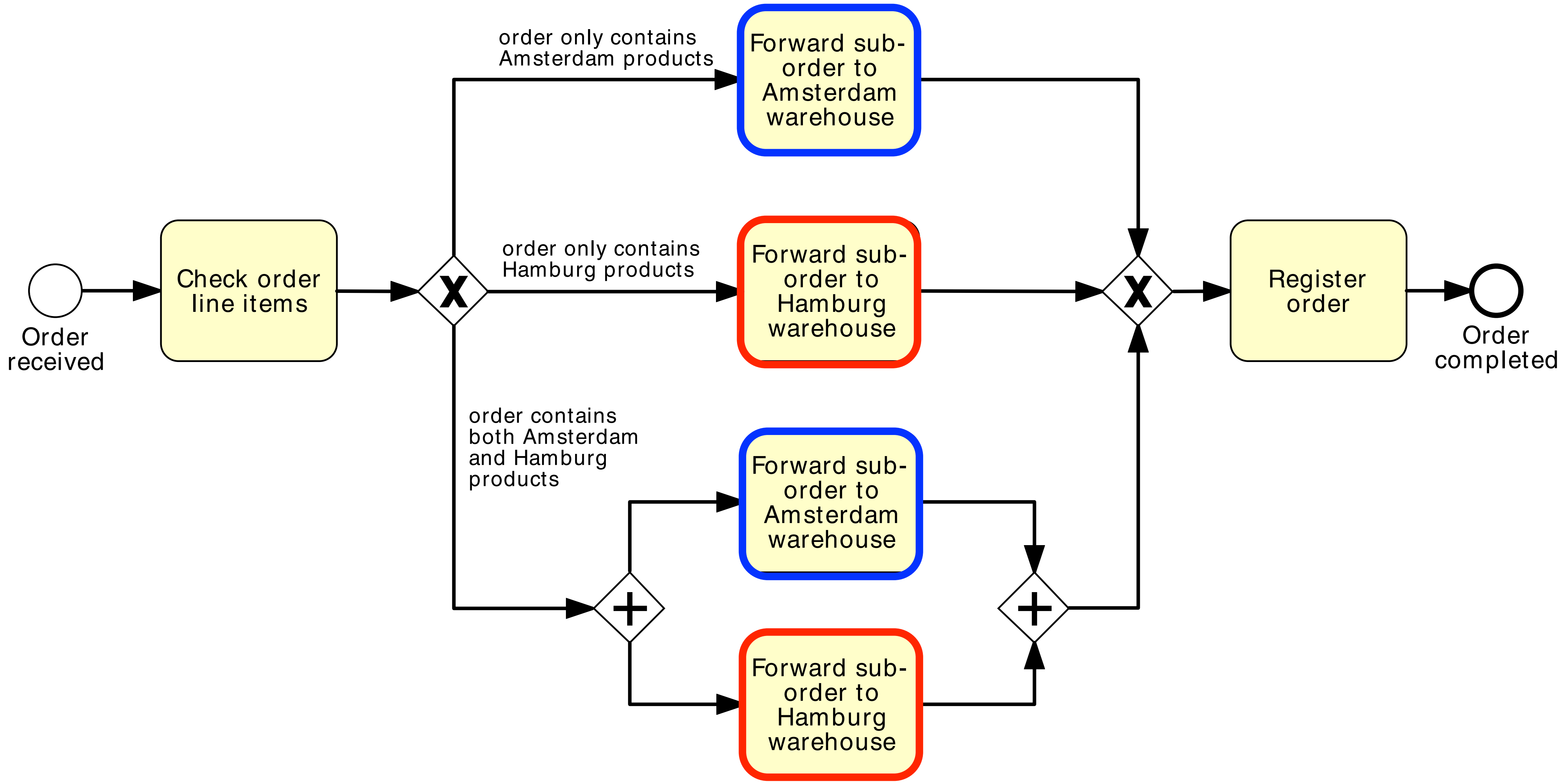
An **AND-gateway** can be omitted when it **follows** an activity or event

Similarly, a **XOR-gateway before** an activity or event can be omitted

Inclusive decisions (one, many)

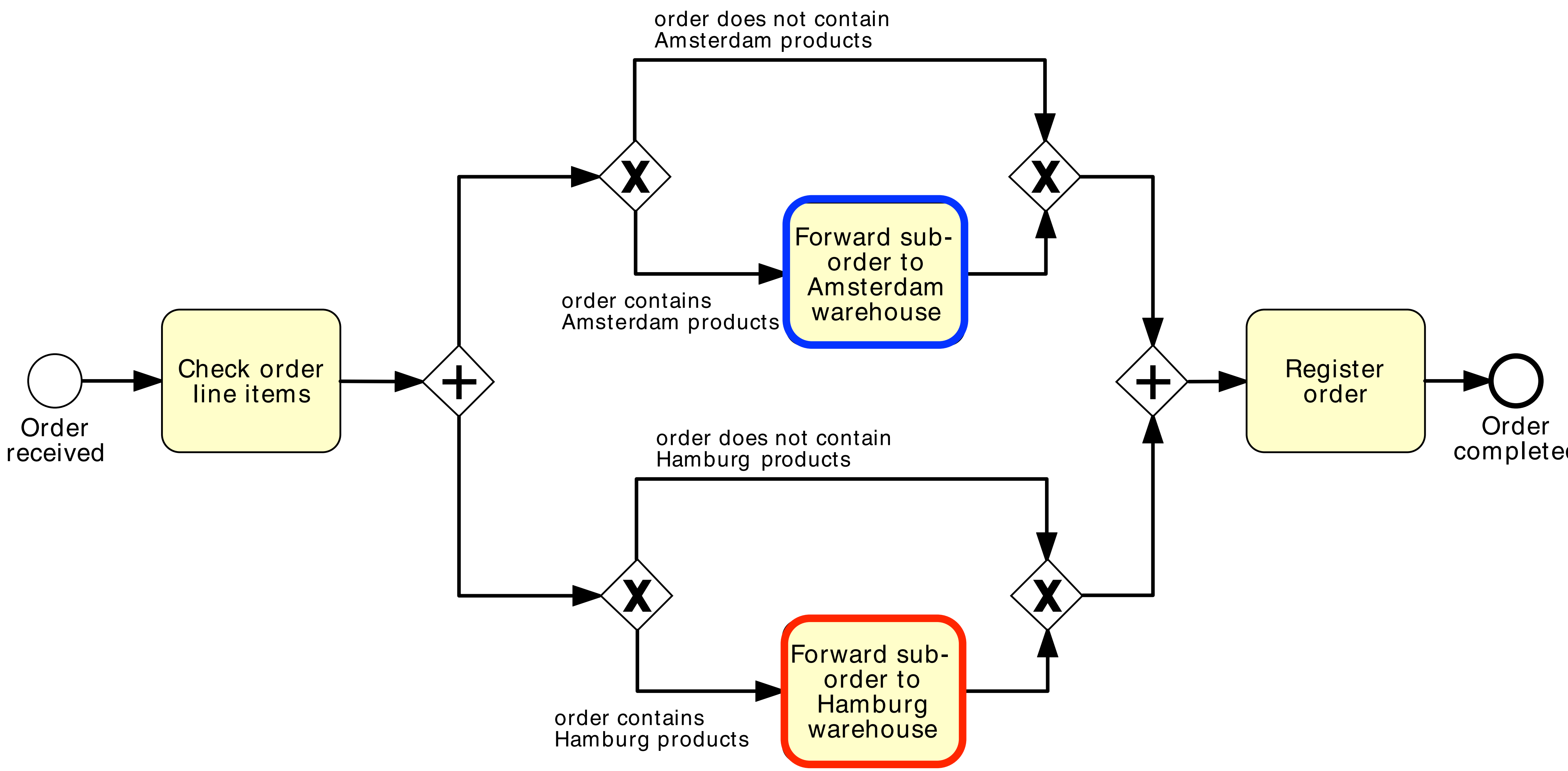


Inclusive decisions: order distribution



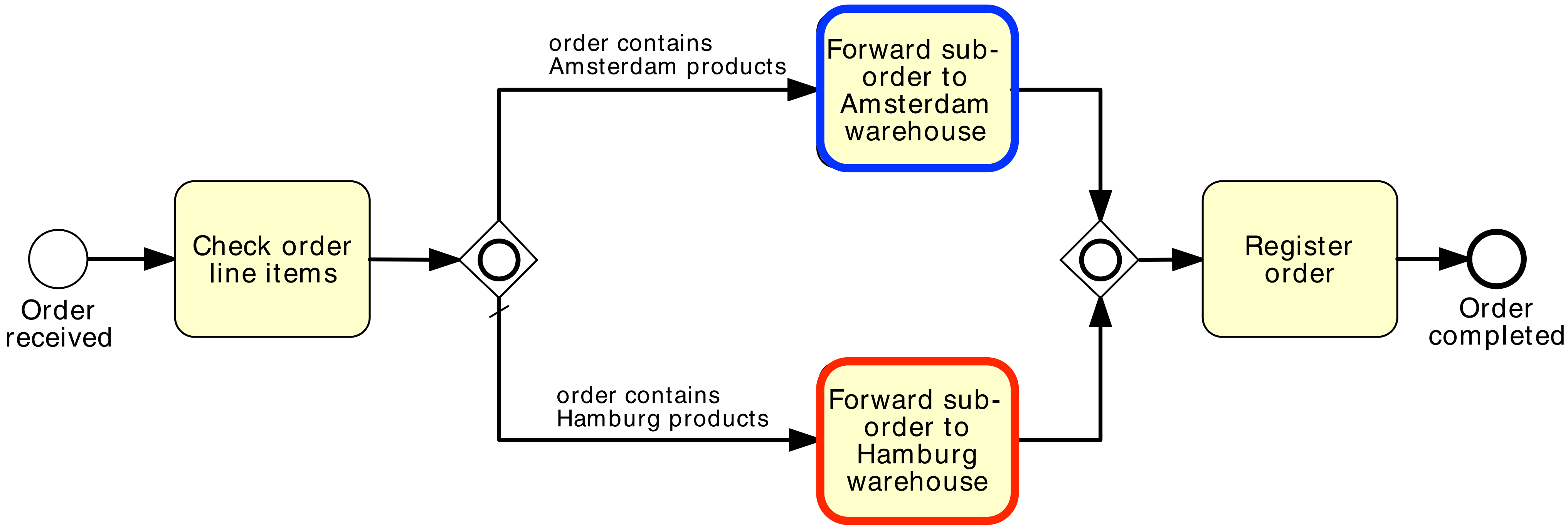
Only XOR / AND gateways, but the diagram is convoluted!
What if we had three or more warehouses? (does not scale)

Inclusive decisions: order distribution



Only XOR / AND gateways, the diagram can ``scale'',
but is it correct? (also the case no-warehouse is now possible)

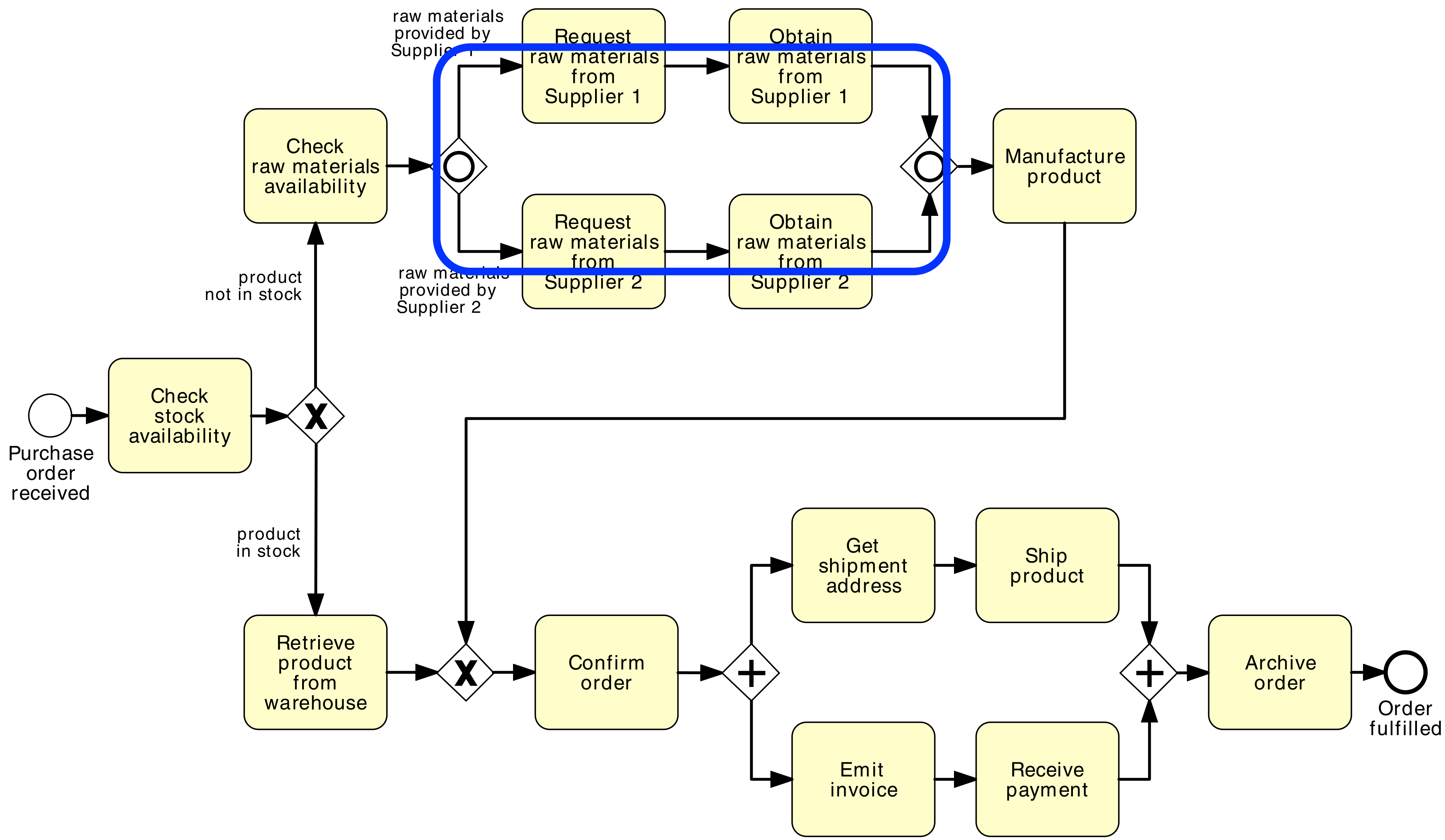
Inclusive decisions: order distribution



OR gateways, the diagram can ``scale'',
but remember all the **issues with unmatched OR-joins**: they are still valid!

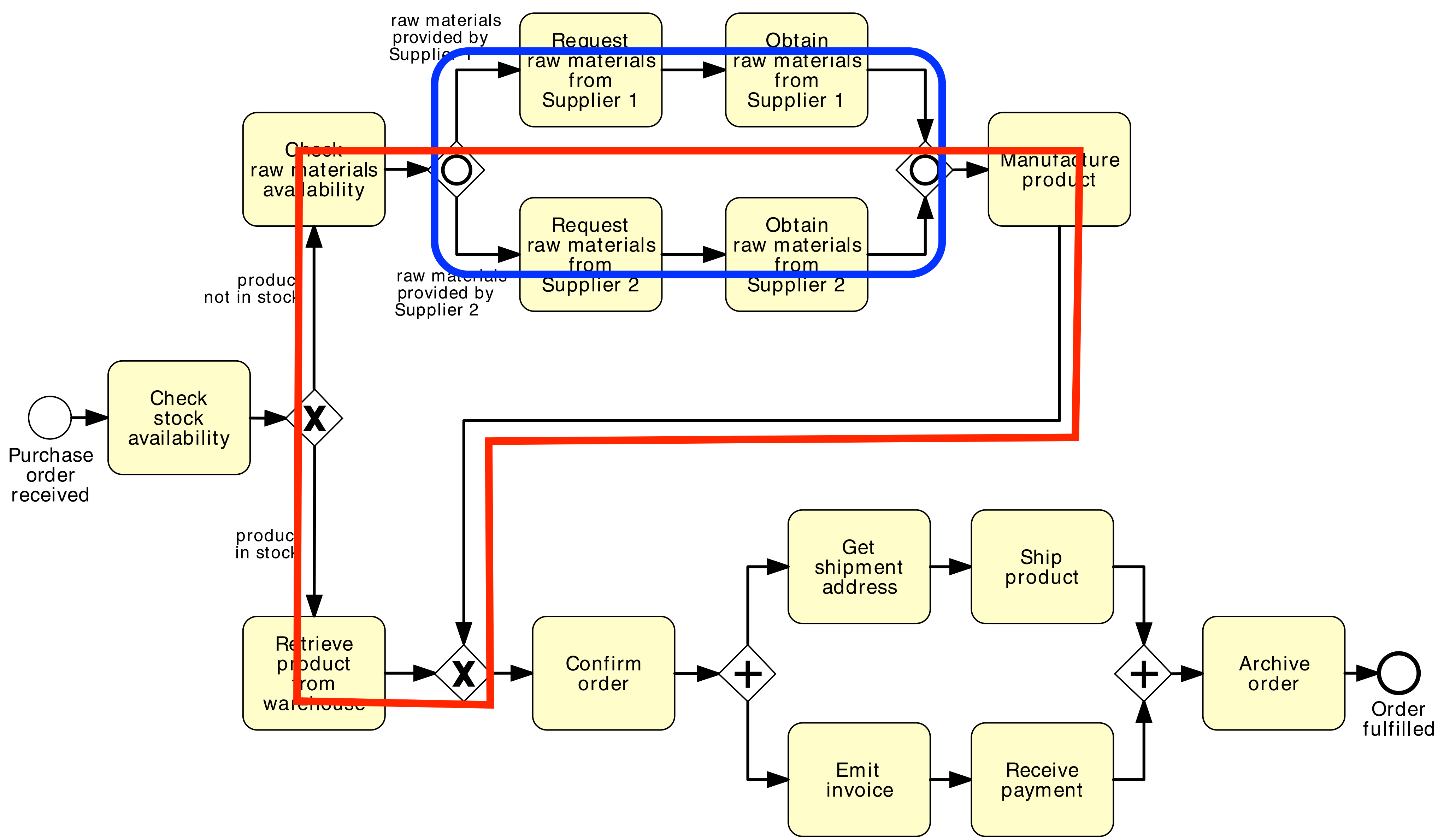
Use OR-gateways only when strictly necessary

XOR + AND + OR: order fulfillment



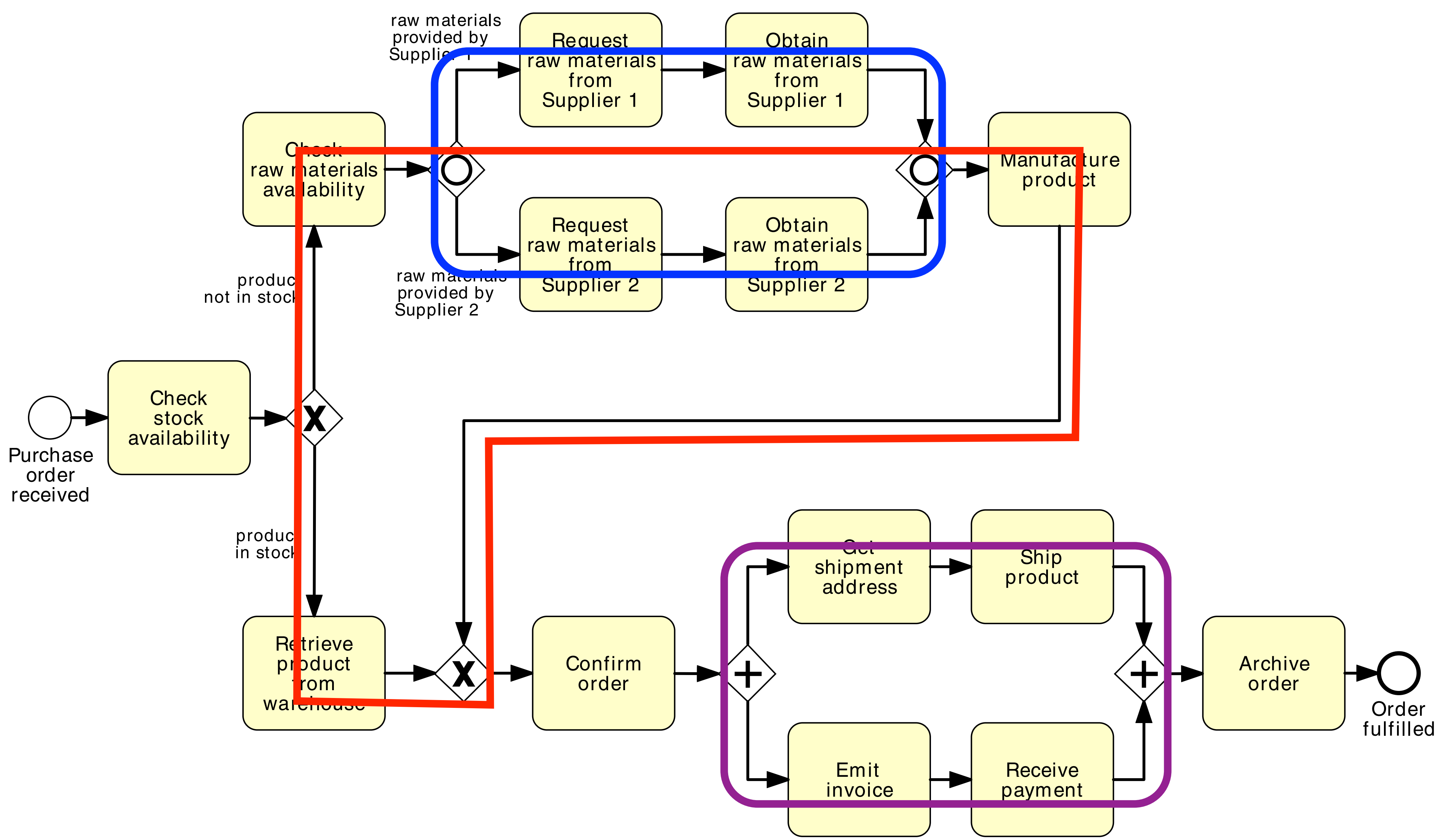
Better if gateways are balanced

XOR + AND + OR: order fulfillment



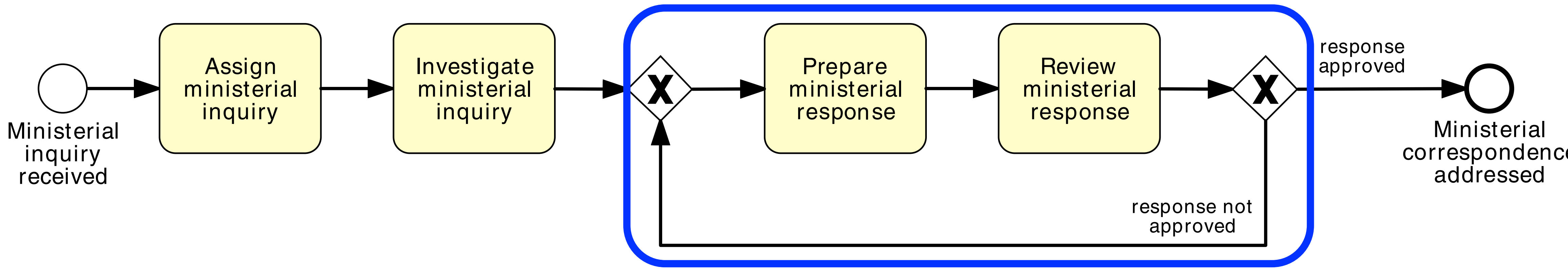
Better if gateways are balanced

XOR + AND + OR: order fulfillment



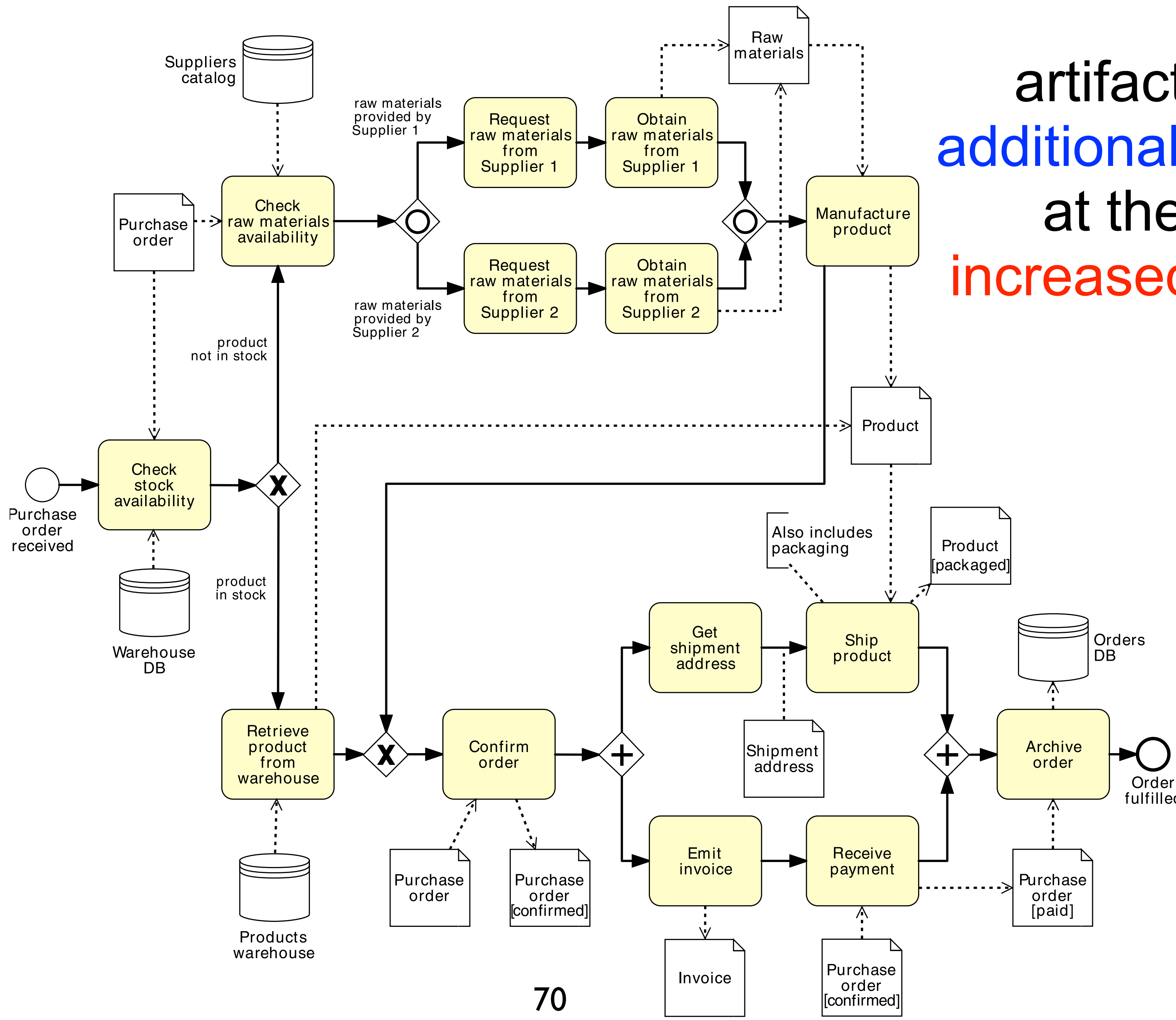
Better if gateways are balanced

Rework and repetition: ministerial correspondence



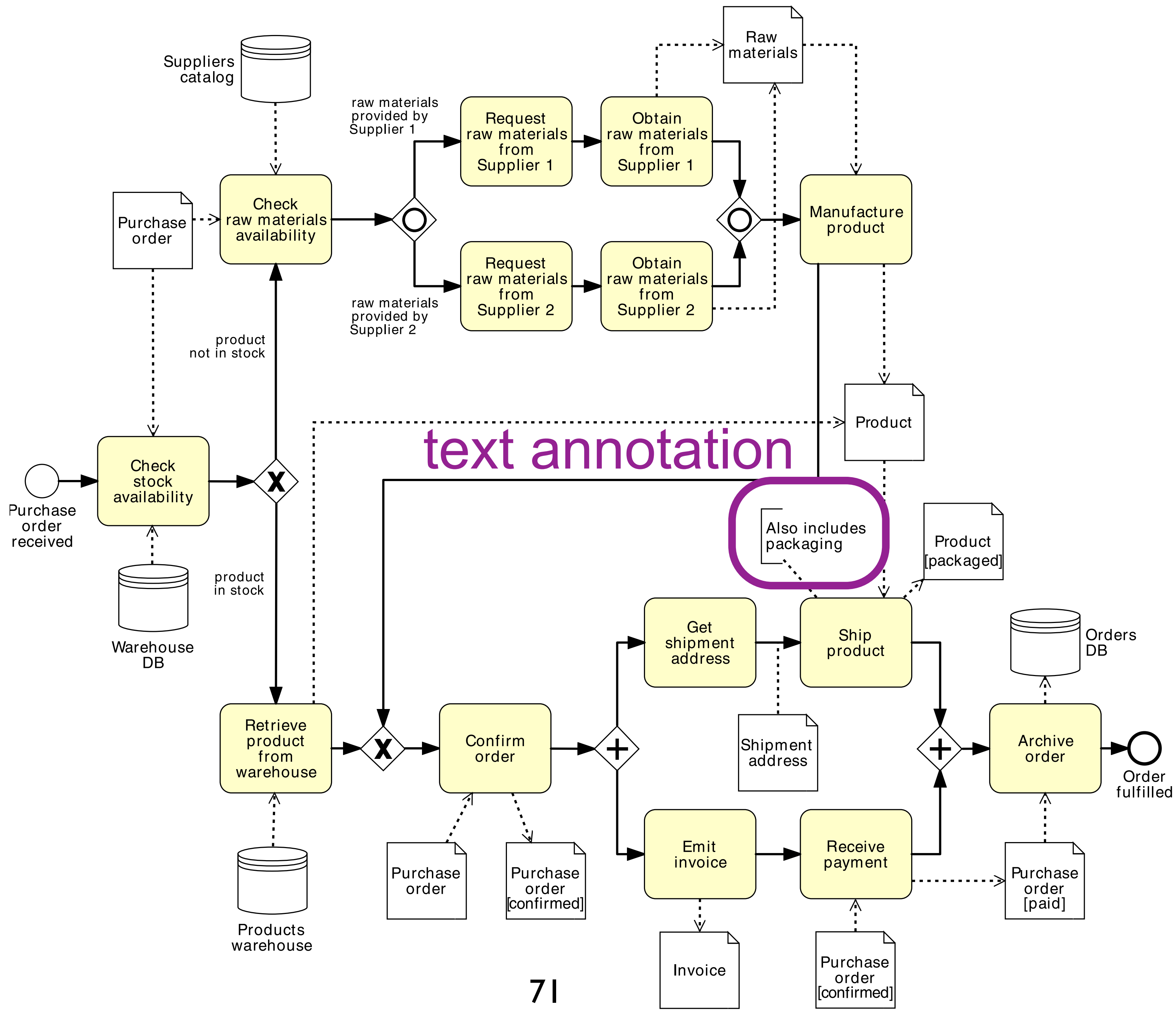
A repetition block starts with a XOR-join
and ends with a decision gateway (XOR-split)

Information artifacts: order fulfillment

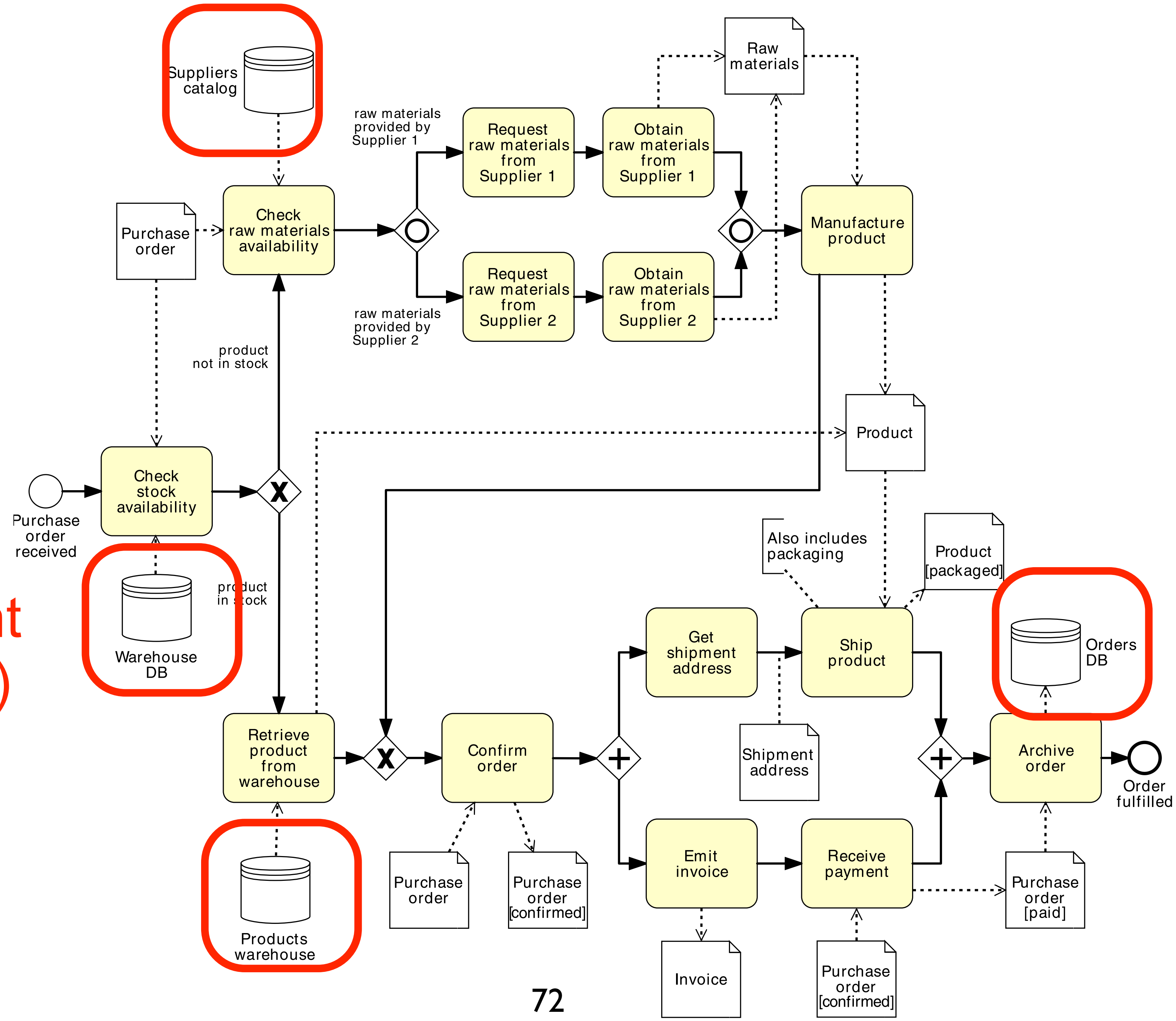


artifacts provide additional information, at the price of increased complexity

Information artifacts: order fulfillment



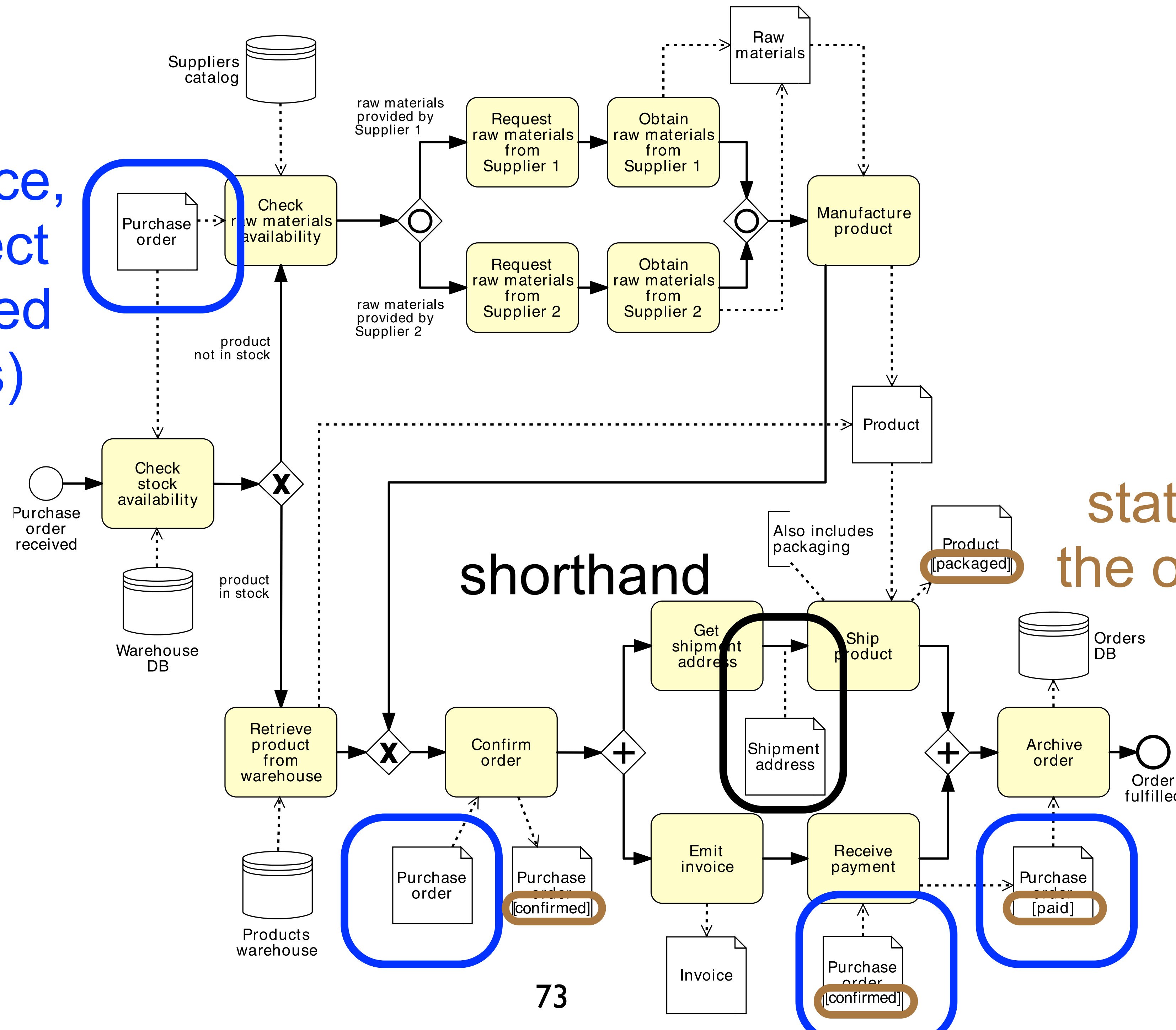
Information artifacts: order fulfillment



data stores (for persistent data objects)

Information artifacts: order fulfillment

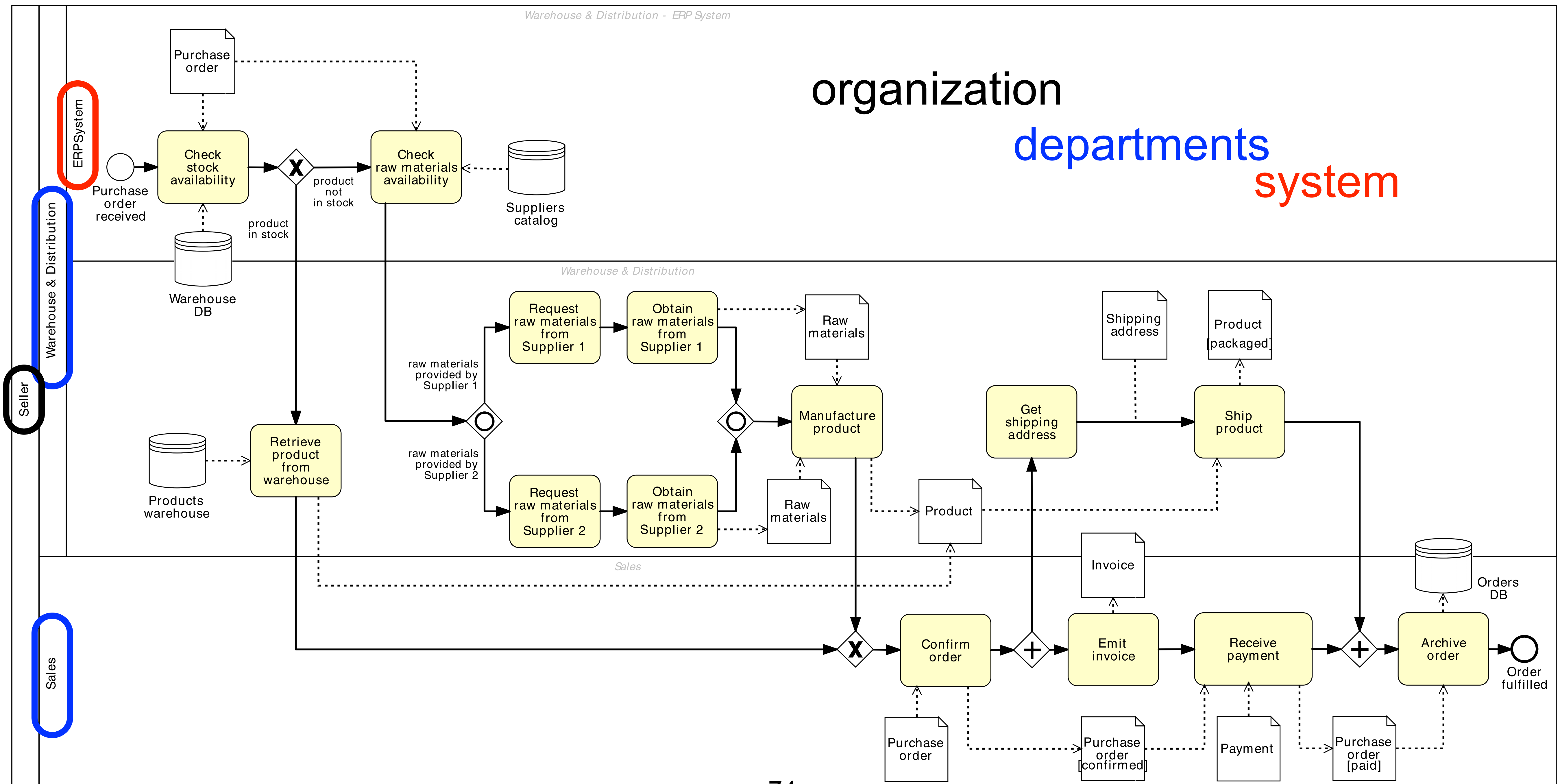
data objects (for convenience, the same object can be repeated several times)



shorthand

state of the object

Resources as lanes: order fulfillment



Placing items

events: must be placed in the proper lane

activities: must be placed in the proper lane

data-objects: placement is irrelevant

gateways:
(X)OR-splits: same lane as preceding decision activity
AND-split, joins: placement is irrelevant

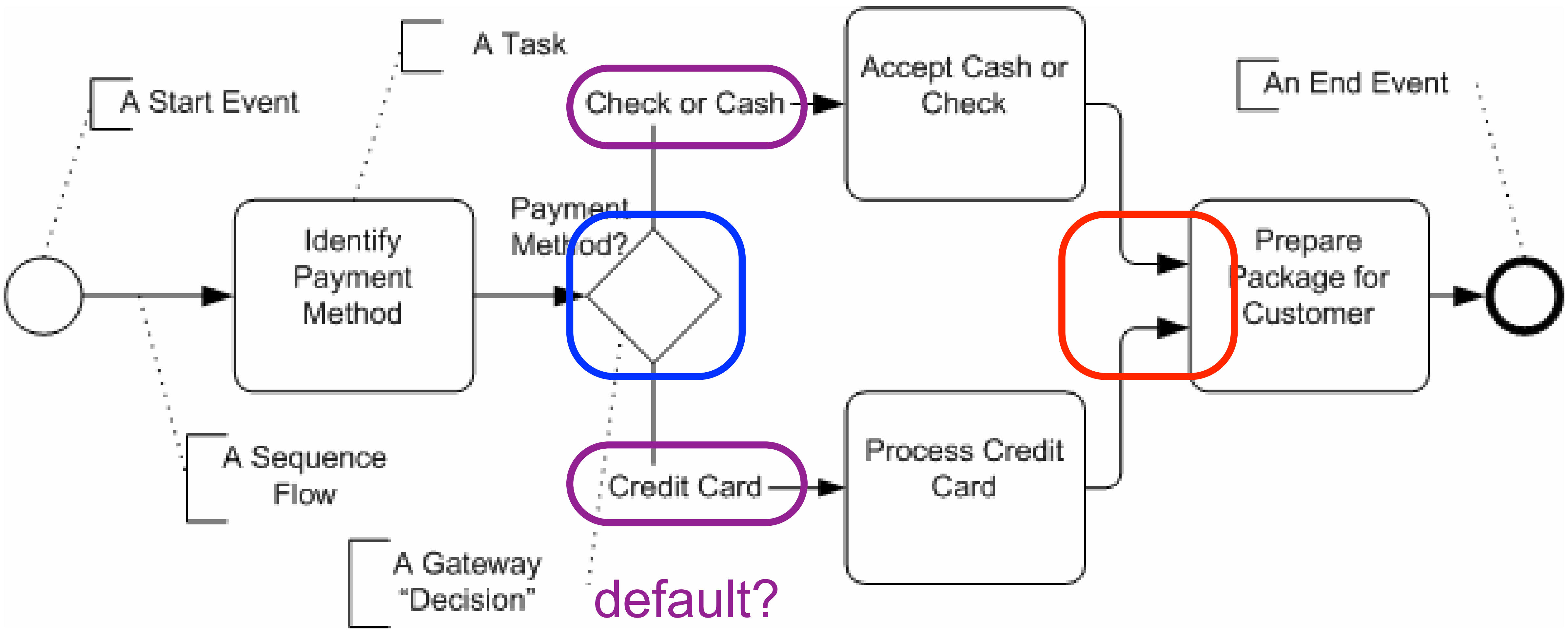
Some remarks

Lanes are often used to separate activities associated with a specific company function or role

Sequence flow may cross the boundaries of Lanes within the same Pool

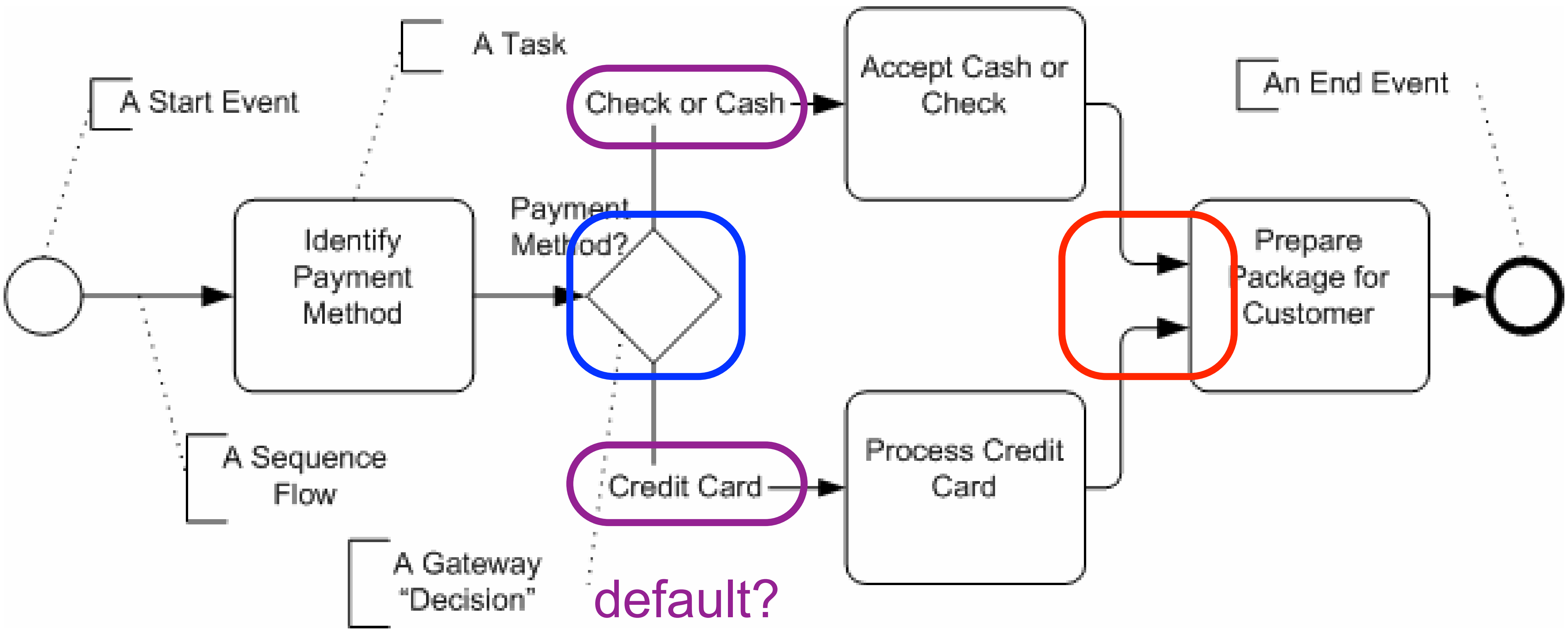
Message flow may not be used between Flow objects in Lanes of the same Pool

Question time



which symbol? which implicit gateway?

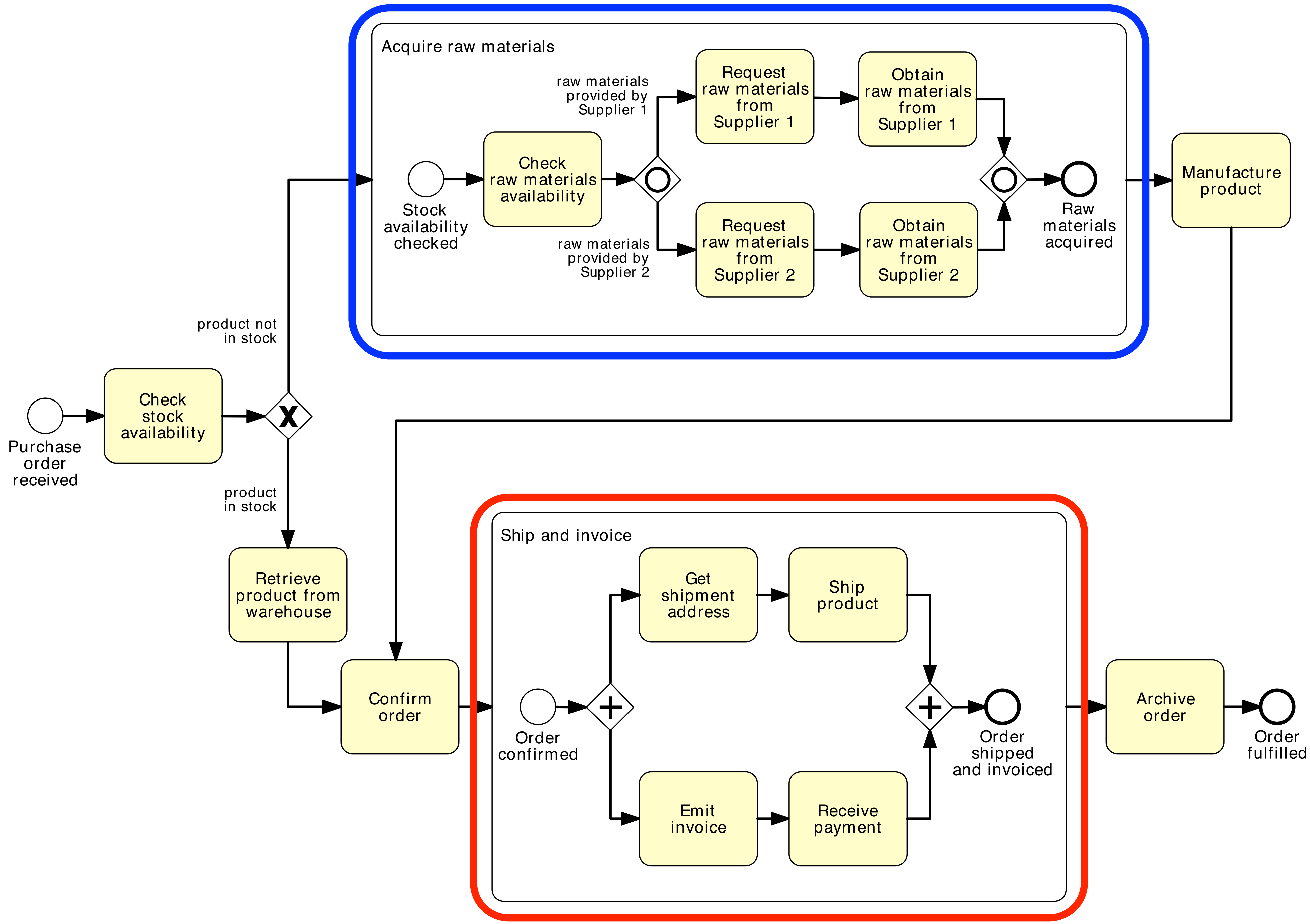
Question time



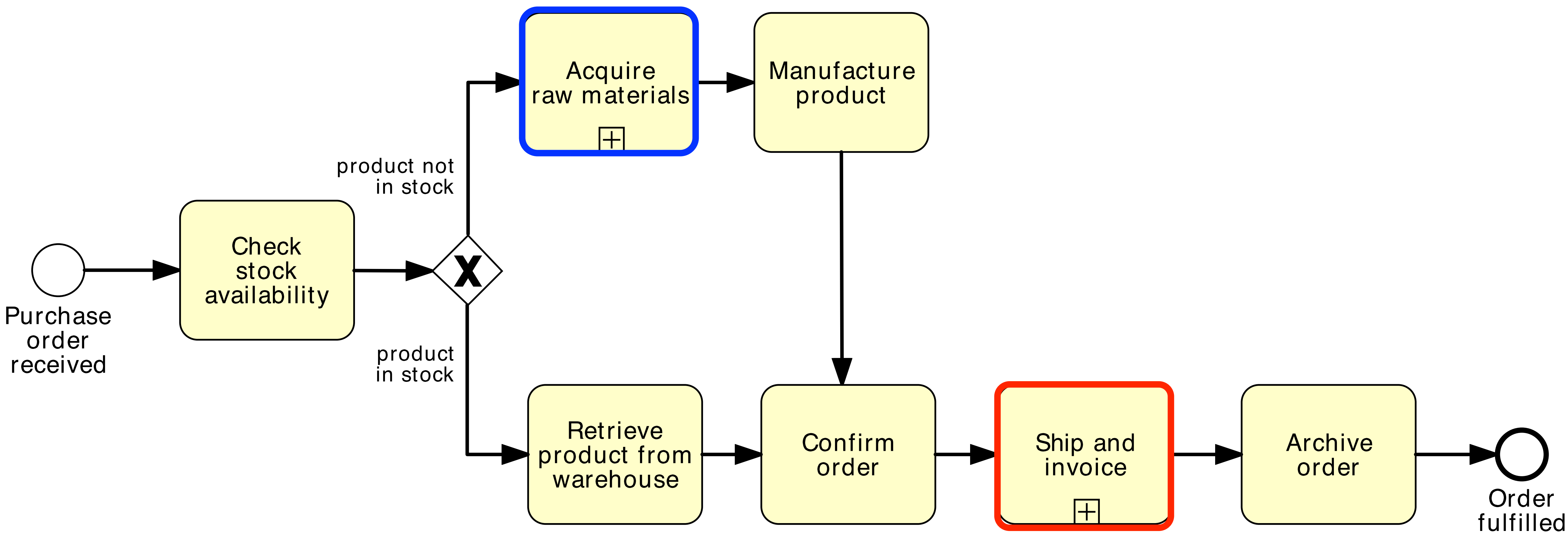
which symbol?
XOR

which implicit gateway?
XOR

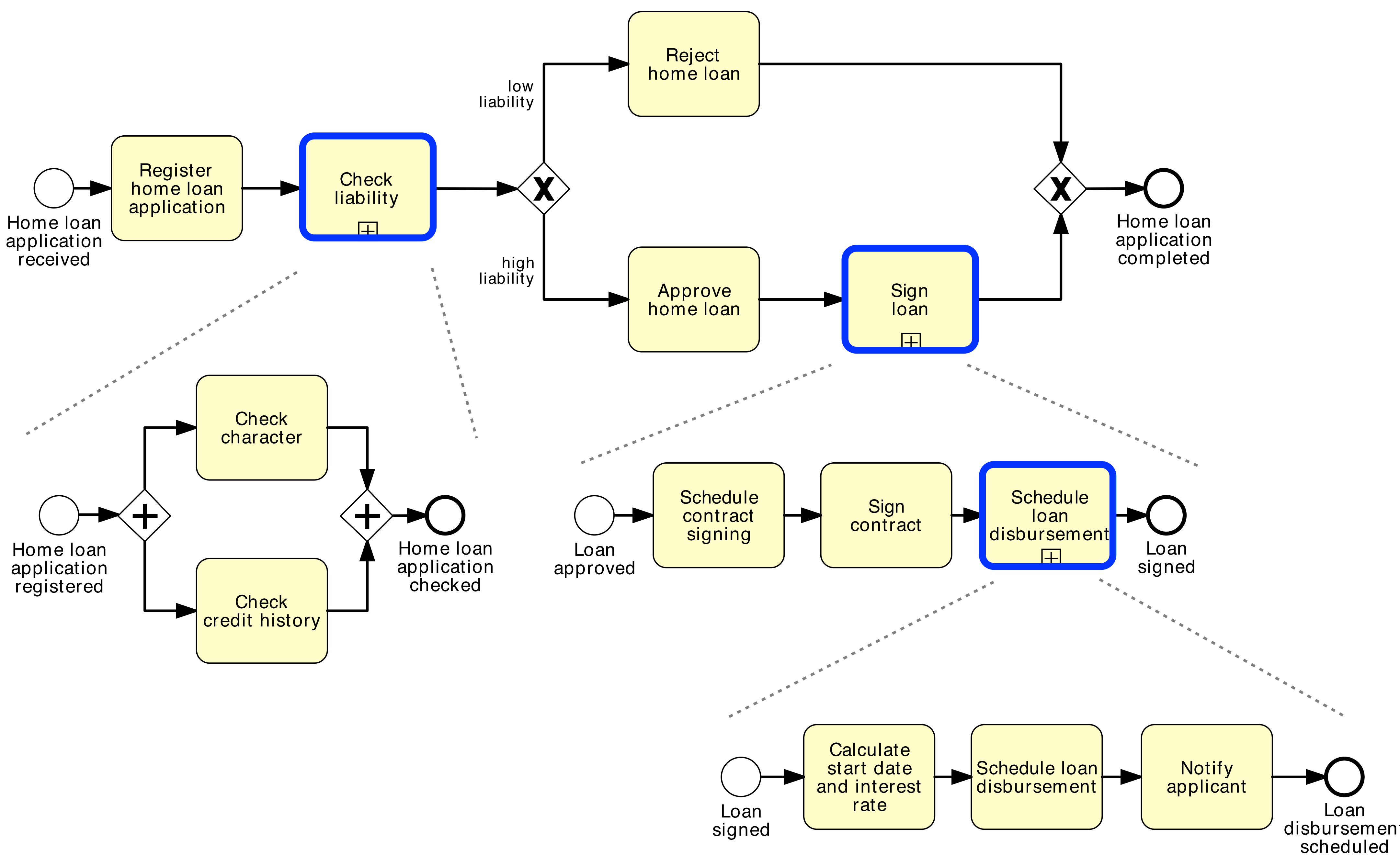
Identify sub-processes: order fulfillment



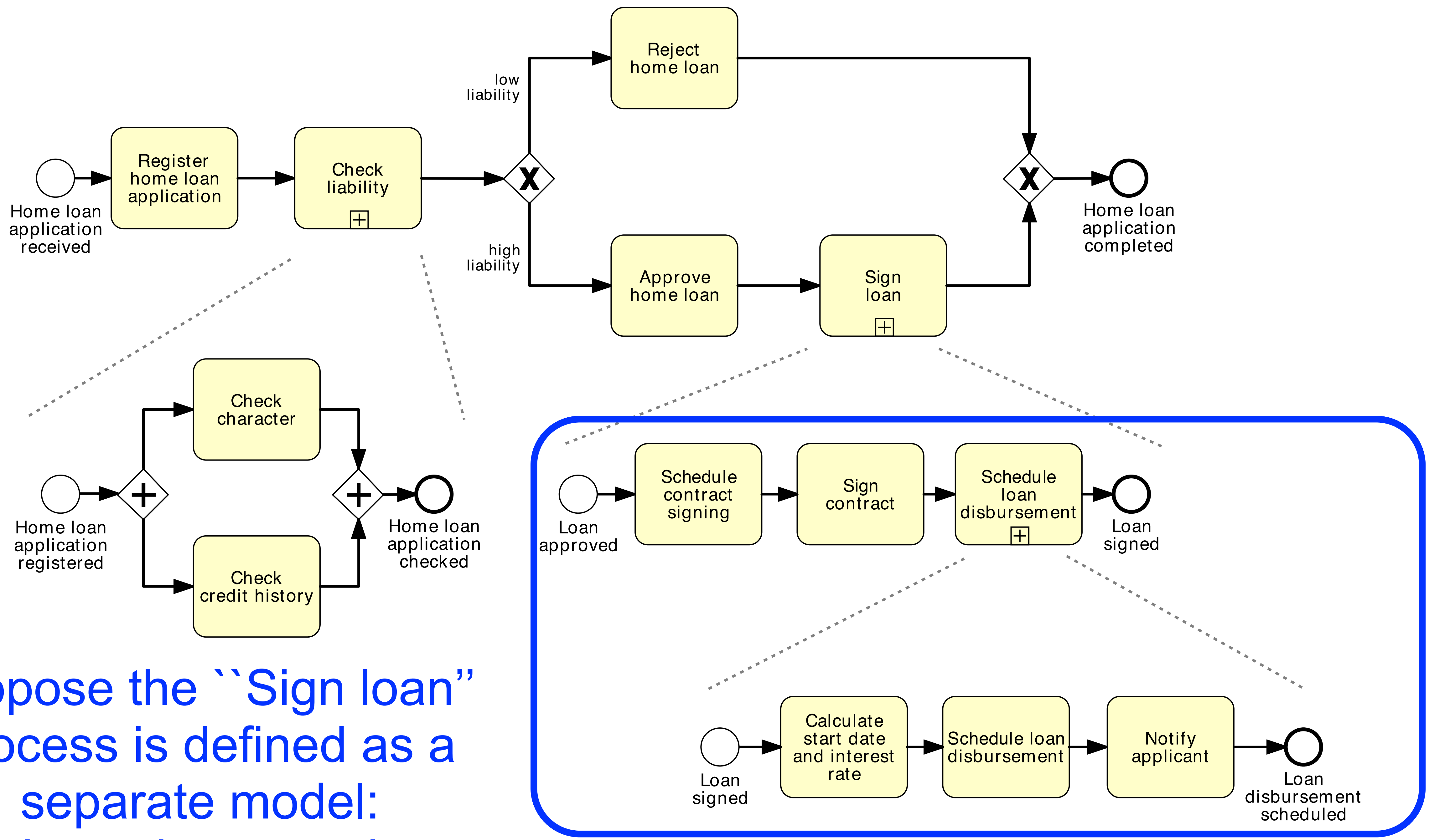
Hiding sub-processes: order fulfillment



Nesting sub-processes: home loans

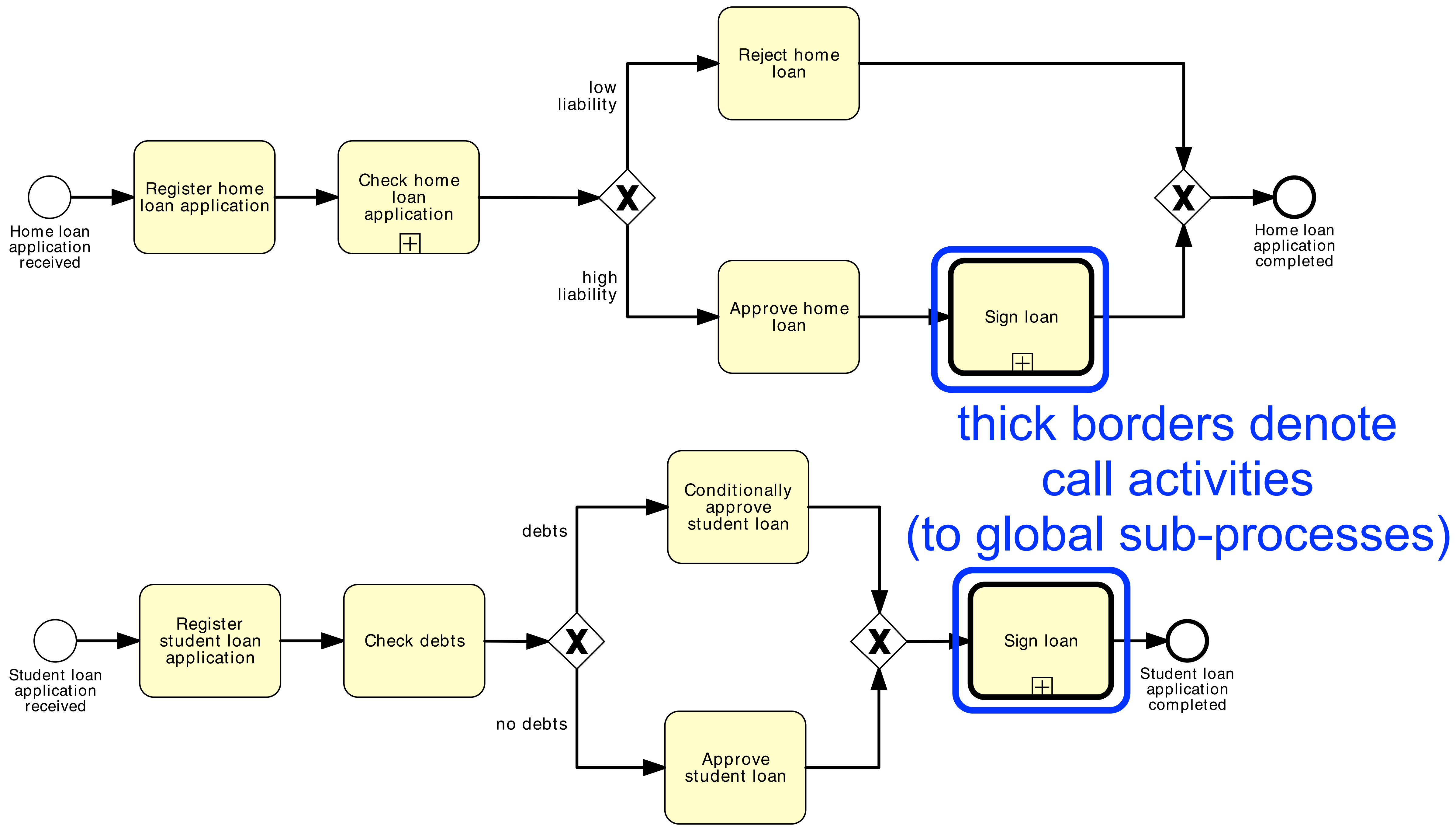


Global sub-processes: home / student loans



suppose the ``Sign loan'' process is defined as a separate model: it can be reused

Call activities: home / student loans



Global processes: advantages

Readability: processes tend to be smaller

Reusability: define once, use many time

Sharing: any change made to a global process
is automatically propagated to all models that invoke it

Exercises

Model the following fragments of business processes for assessing loan applications:

Exercise: loan application 1

Once a loan application has been **approved** by the loan provider, an acceptance pack is **prepared** and **sent** to the customer.

The acceptance pack includes a repayment schedule which the customer needs to agree upon by **sending the signed documents** back to the loan provider.

The latter then verifies the repayment agreement:
if the applicant disagreed with the repayment schedule, the loan provider **cancels** the application;
if the applicant agreed, the loan provider **approves** the application.
In either case, the process completes with the loan provider **notifying** the applicant of the application status.

Exercise: loan application 2

A loan application is **approved** if it passes **two checks**:
(i) the applicant's loan **risk assessment**, which is done automatically by a system, and
(ii) the **appraisal** of the property for which the loan has been asked, carried out by a property appraiser.

The risk assessment requires a **credit history check** on the applicant, which is performed by a financial officer.

Once both the loan risk assessment and the property appraisal have been performed, a loan officer can **assess** the applicant's eligibility.

If the applicant is not eligible, the application is **rejected**,
otherwise the acceptance pack is **prepared and sent** to the applicant.

Exercise: loan application 3

A loan application may be coupled with a home insurance which is offered at discounted prices.
The applicant may express their interest in a home insurance plan at the time of submitting their loan application to the loan provider.

Based on this information, **if** the loan application is **approved**, the loan provider may **either only** send an **acceptance pack** to the applicant, **or also** send a **home insurance quote**.

The process then continues with the **verification** of the repayment agreement.

Exercise: loan application 4

Once a loan application is **received** by the loan provider, and before proceeding with its assessment, the application itself needs to be **checked** for completeness.

If the application is incomplete, it is **returned** to the applicant, so that they can **fill out** the missing information and **send it back** to the loan provider.

This process is **repeated** until the application is complete.

Exercise: loan application 5

Put together the four fragments of the loan assessment process that you created in previous Exercises.

Then extend the resulting model by adding all the required artifacts.

Moreover, attach annotations to specify the business rules behind:
(i) checking an application completeness,
(ii) assessing an application eligibility, and
(iii) verifying a repayment agreement.

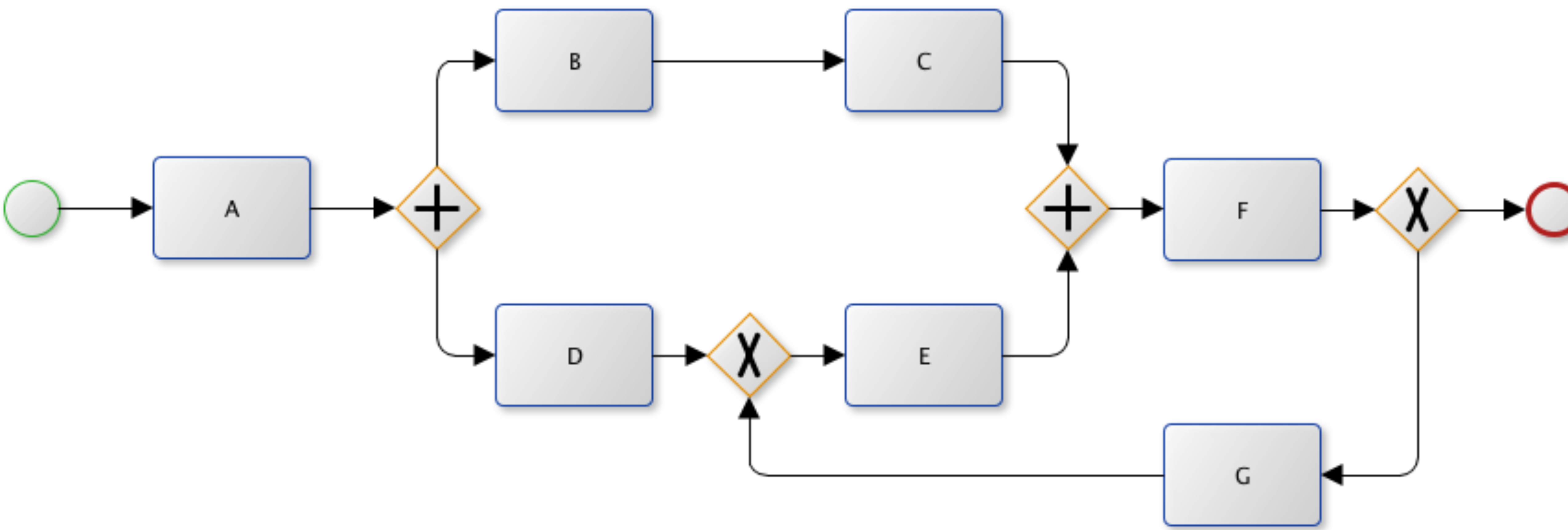
Exercise: loan application 6

Extend the business process for assessing loan applications that you created in previous exercises by considering the following resource aspects.

The process for assessing loan applications is executed by four roles within the **loan provider**:
a **financial officer** takes care of checking the applicant's credit history;
a **property appraiser** is responsible for appraising the property;
an **insurance sales representative** sends the home insurance quote to the applicant if this is required.
All other activities are performed by the **loan officer** who is the main point of contact with the applicant.

Exercises: refactoring

Can the process model below execute correctly?
If not, how can it be fixed without affecting the cycle, i.e. such that F, G, and E all remain in a cycle?

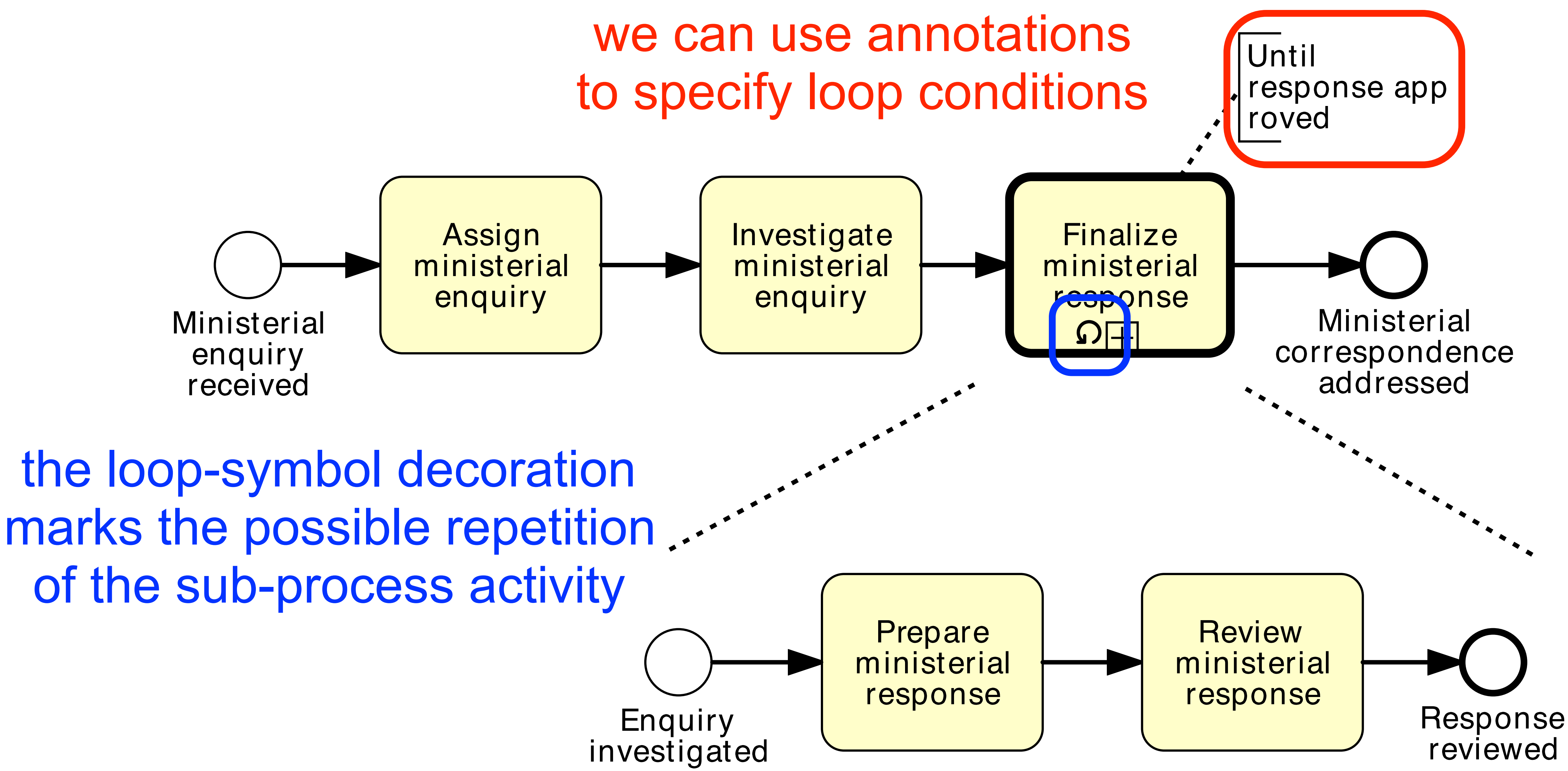


Semantics annotations

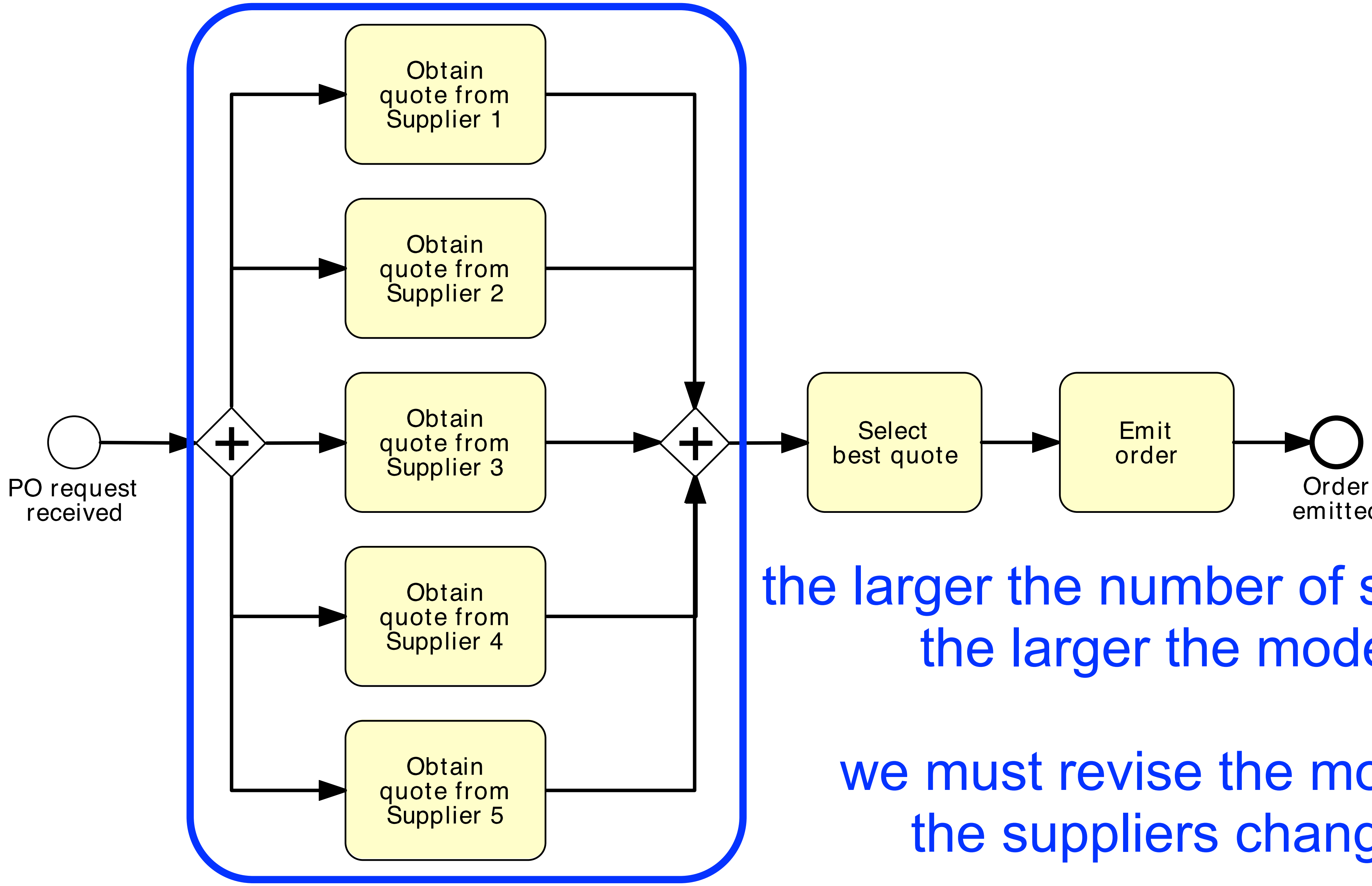
The graphical syntax is not expressive enough to model exactly all interesting situations

In many cases part of the behaviour is moved to decorations and annotations (without them no implementation is possible)

Loop annotation: ministerial correspondence



Parallel repetition: procurement process



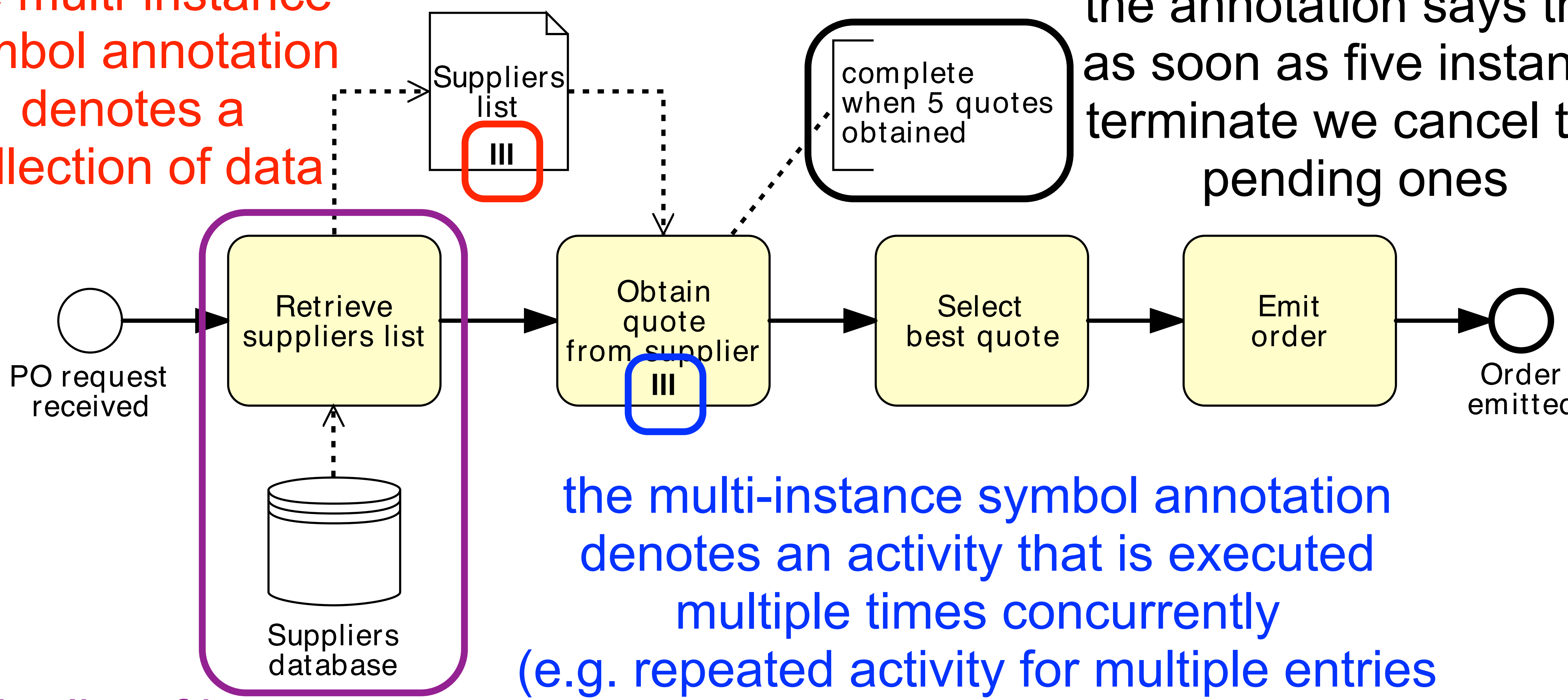
the larger the number of suppliers the larger the model!

we must revise the model if the suppliers change!

Multi-instance activities: procurement process

the multi-instance symbol annotation denotes a collection of data

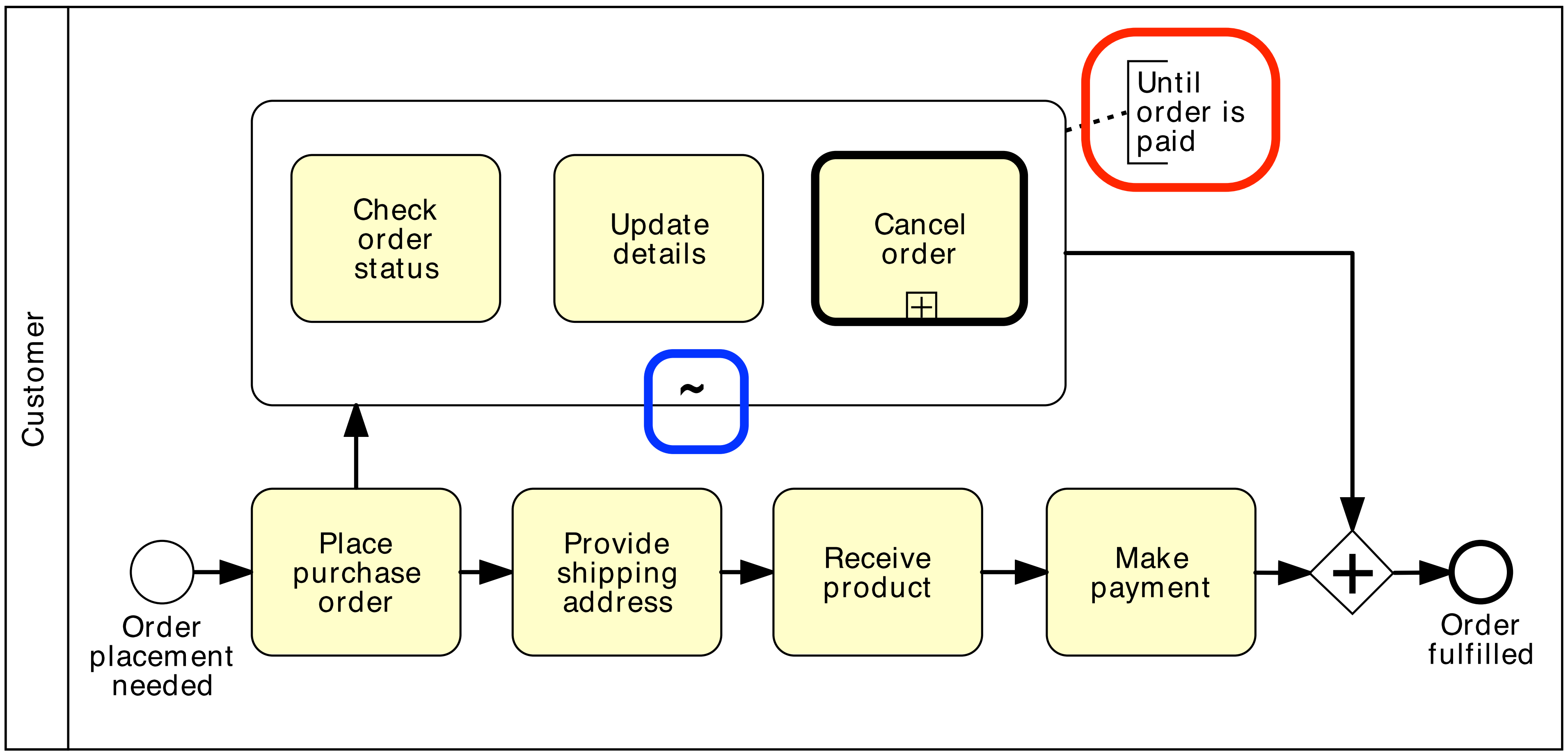
the annotation says that as soon as five instance terminate we cancel the pending ones



the multi-instance symbol annotation denotes an activity that is executed multiple times concurrently (e.g. repeated activity for multiple entries or data-items)

the list of instances is determined dynamically

Ad-hoc sub-processes: customer process



we can use annotations to specify loop conditions

the ad-hoc symbol annotation denotes an uncontrolled repetition of activities: they may be repeated multiple times with no specific order or not occur at all, until a condition is met

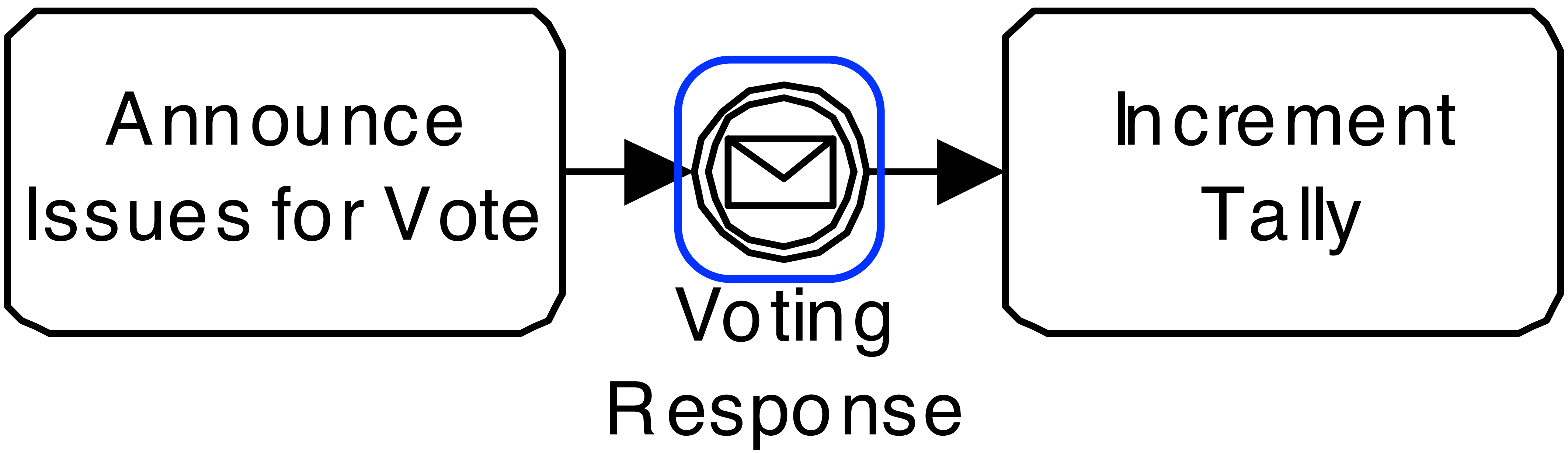
Message annotated events and activities

A start event can be annotated with a white-envelope:
a process instance is created
when a certain message is received

An end event can be annotated with a black-filled envelope:
the process concludes by sending a message

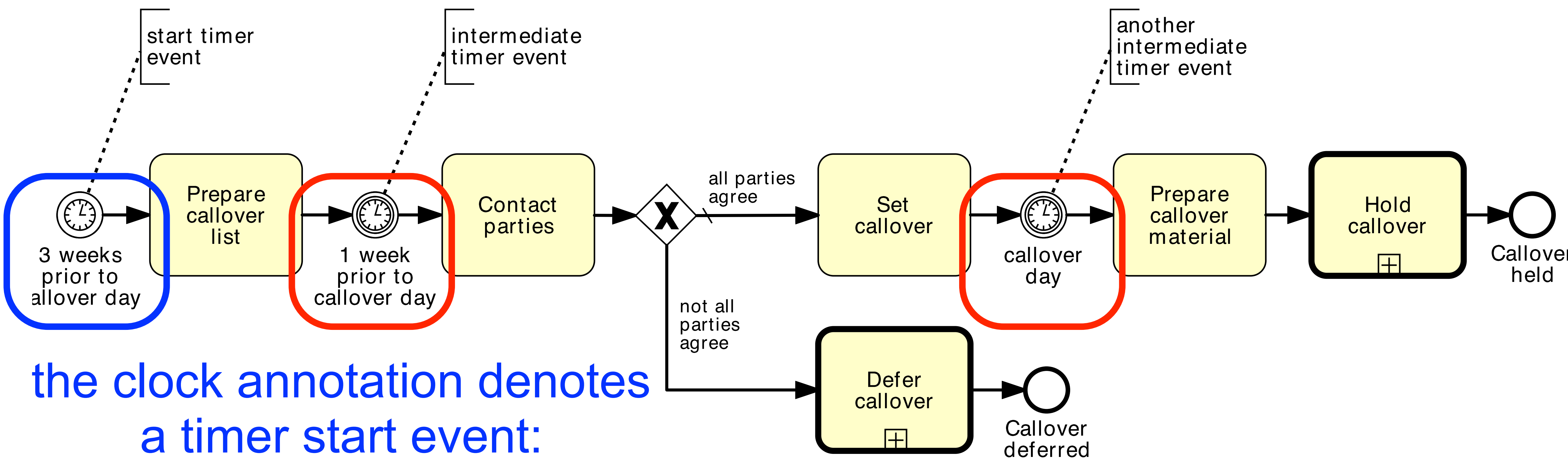
Intermediate events and activities can be annotated with
both kinds of envelope
(white = receipt of a message,
black = the sending of a message)

Process break (event waiting)



the envelope annotation denotes an intermediate message event: it signals the receipt of a message

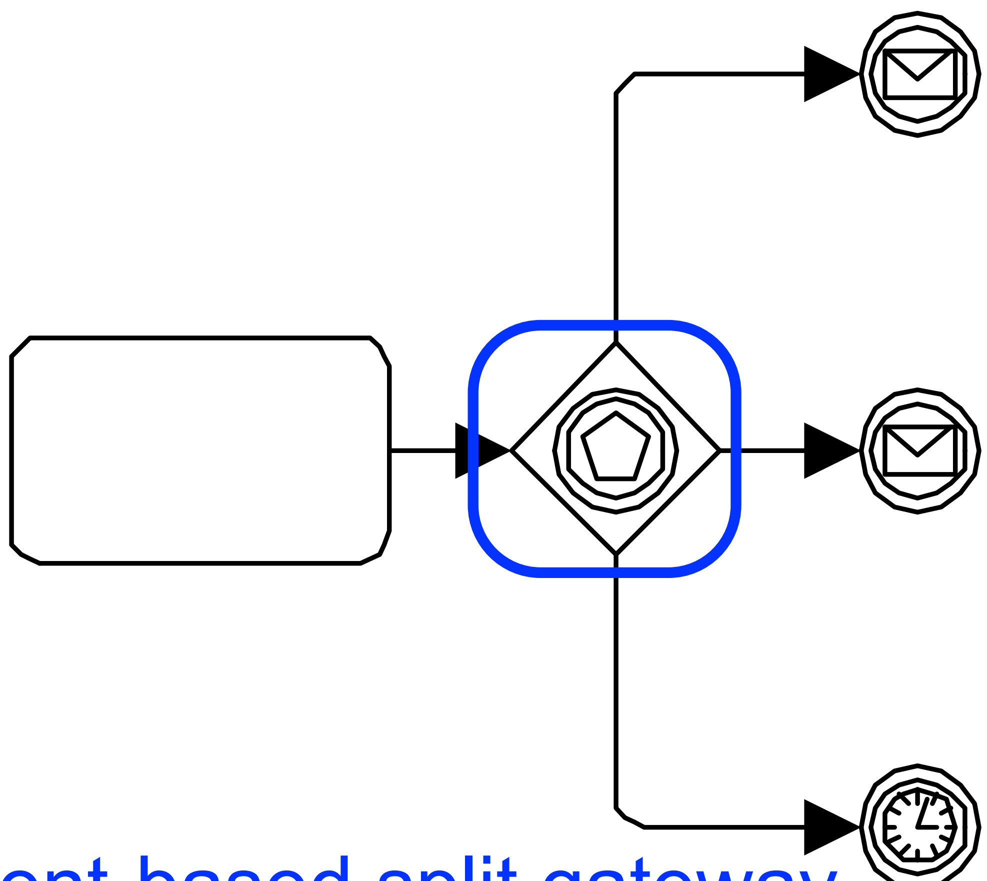
Timer events: small claims tribunal

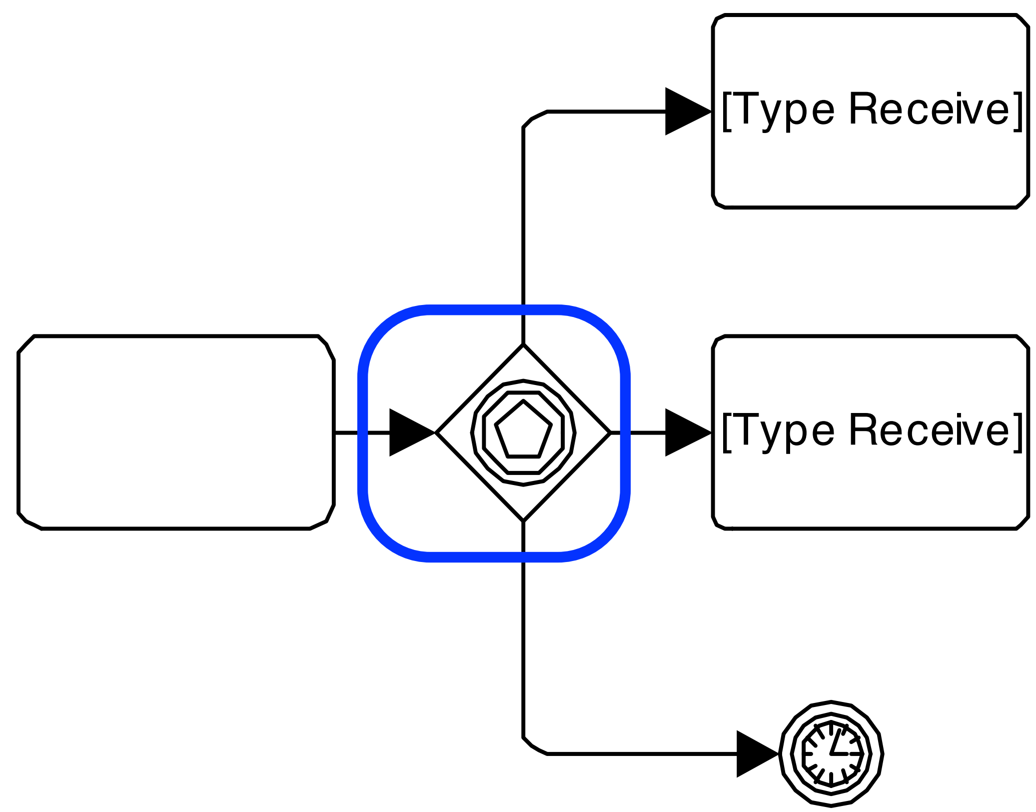


the clock annotation denotes a timer start event: an instance of the process is created when some temporal event happens

the clock annotation denotes a timer intermediate event: the process is blocked until a time-out expires

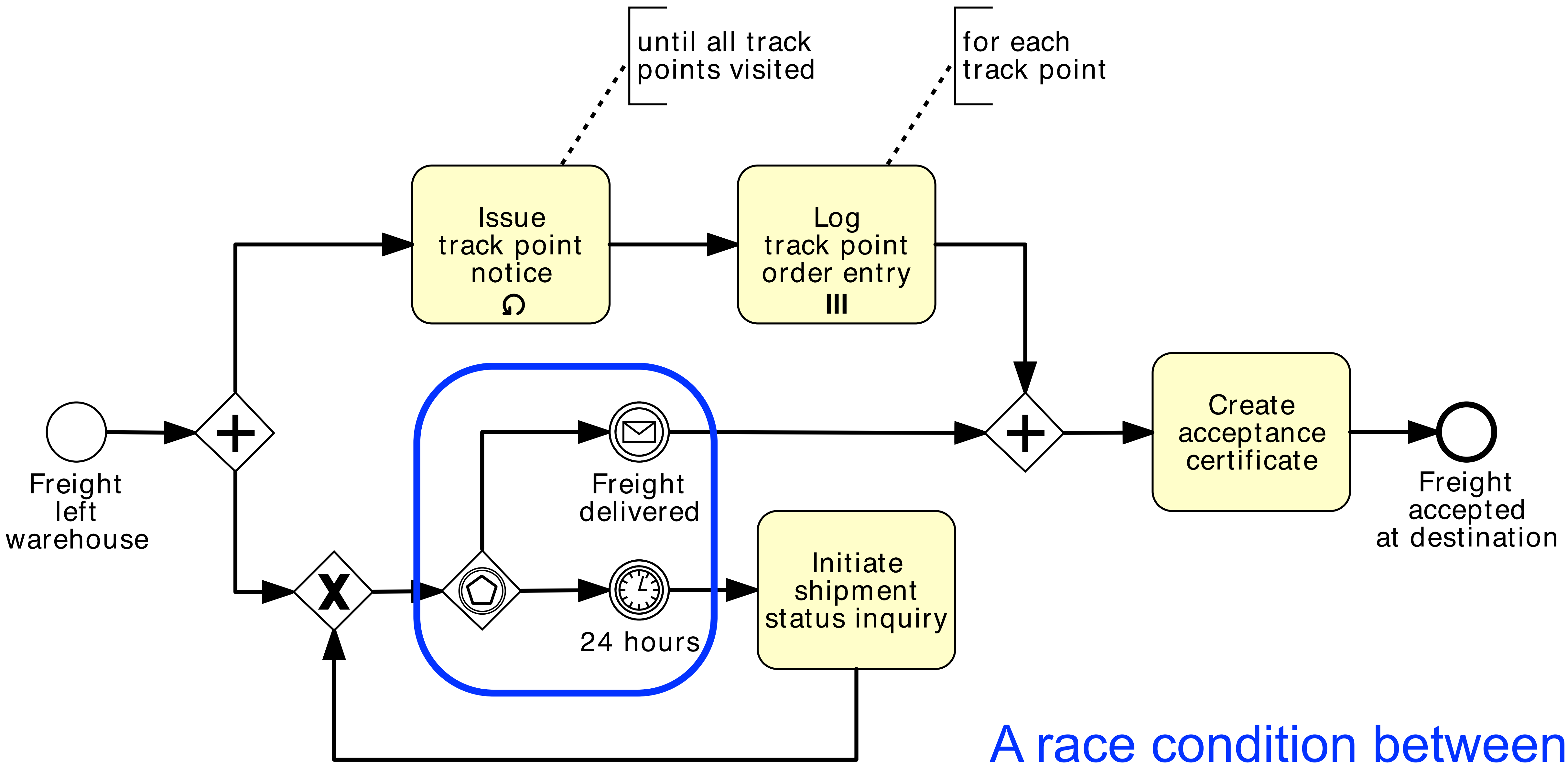
Event-based decisions (also deferred choice)





Event-based split gateway can be used to select a branch based on some external event

Deferred choice



A race condition between an incoming message and a timer

Exceptions: rainy-days vs sunny-days

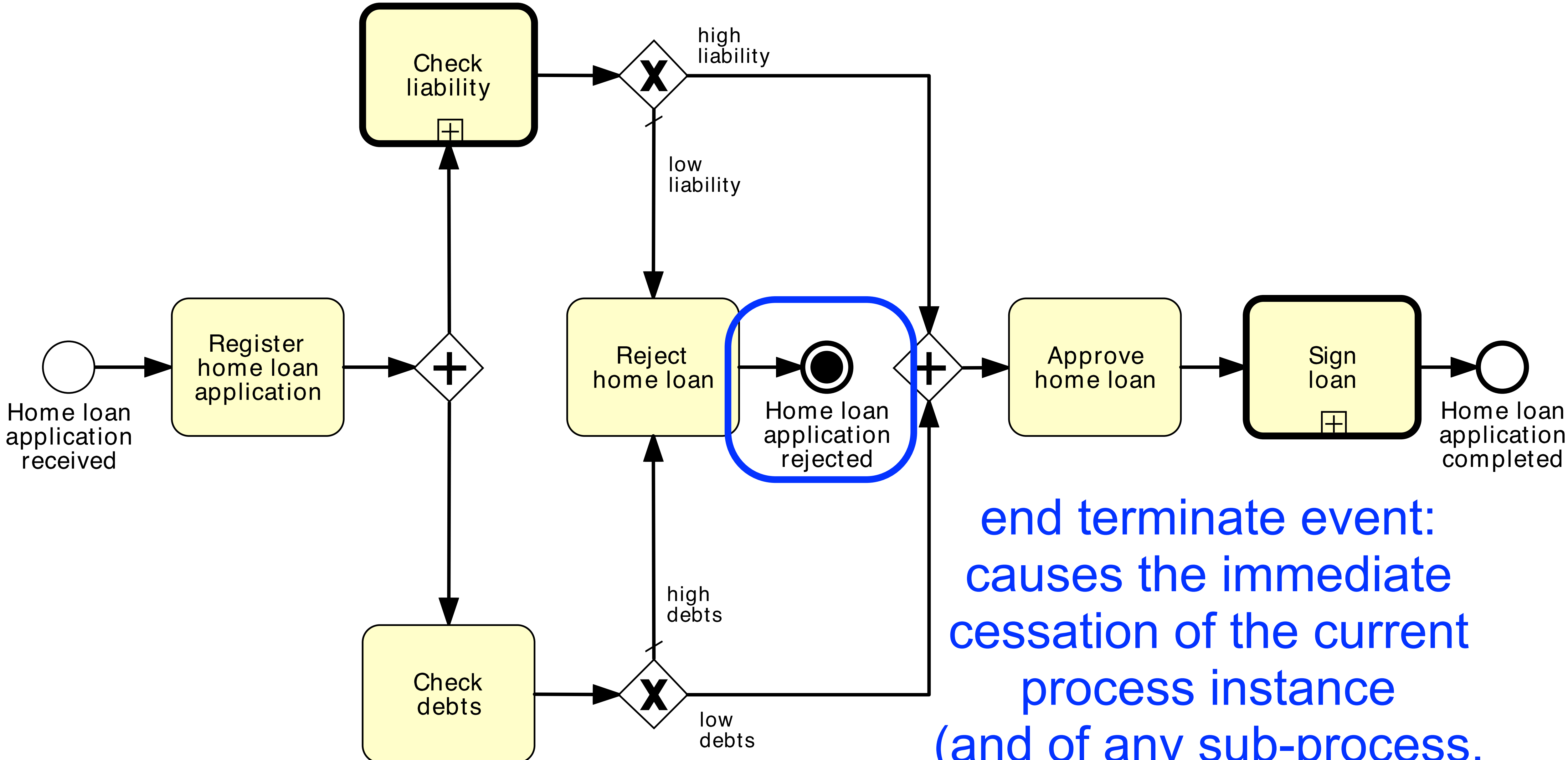
Exceptions are events
that deviate a process from its normal course

They include: business faults (e.g., out of stock),
technology faults (e.g., database crash)

Exceptions provoke the interruption or abortion
of the running process instance

Before adding exceptions it is important to have
the sunny-day scenario well understood

Process abortion: home loan



end terminate event:
causes the immediate cessation of the current process instance
(and of any sub-process, but not of the parent process if any)

Handling exceptions: rainy-days vs sunny-days

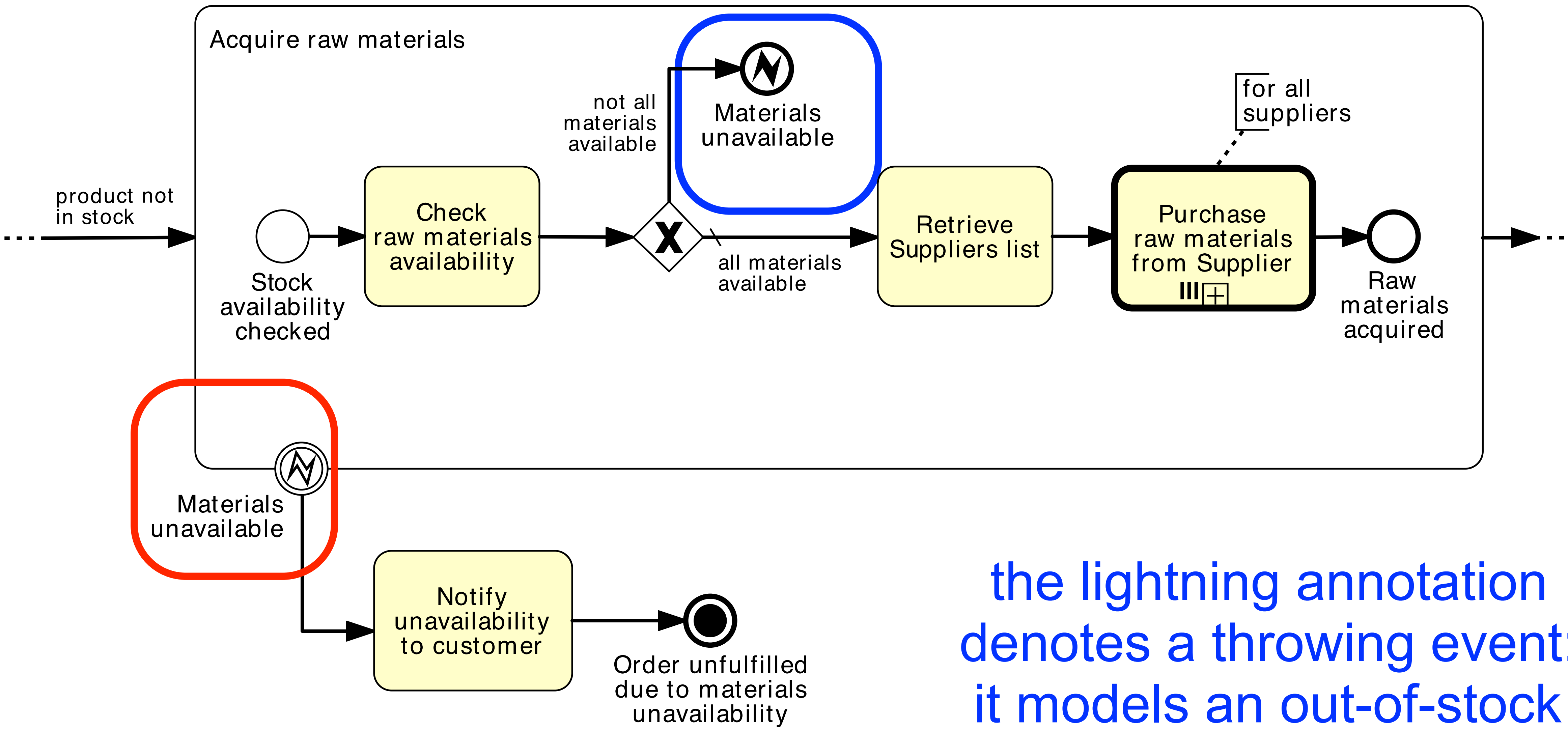
We can handle exceptions of sub-processes
by interrupting the activity that caused the exception
and moving the control flow to another process

The recovery procedure can try to bring the
process back to a consistent state

Error end events are used to interrupt the execution

Boundary events trigger the recovery procedure
(called exception flow)

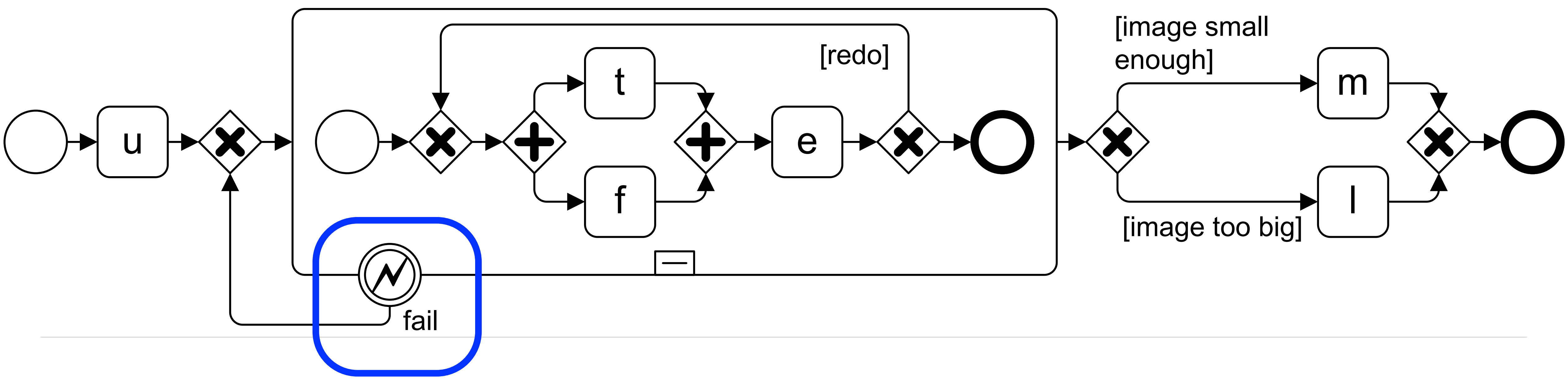
Throwing and catching: order fulfillment



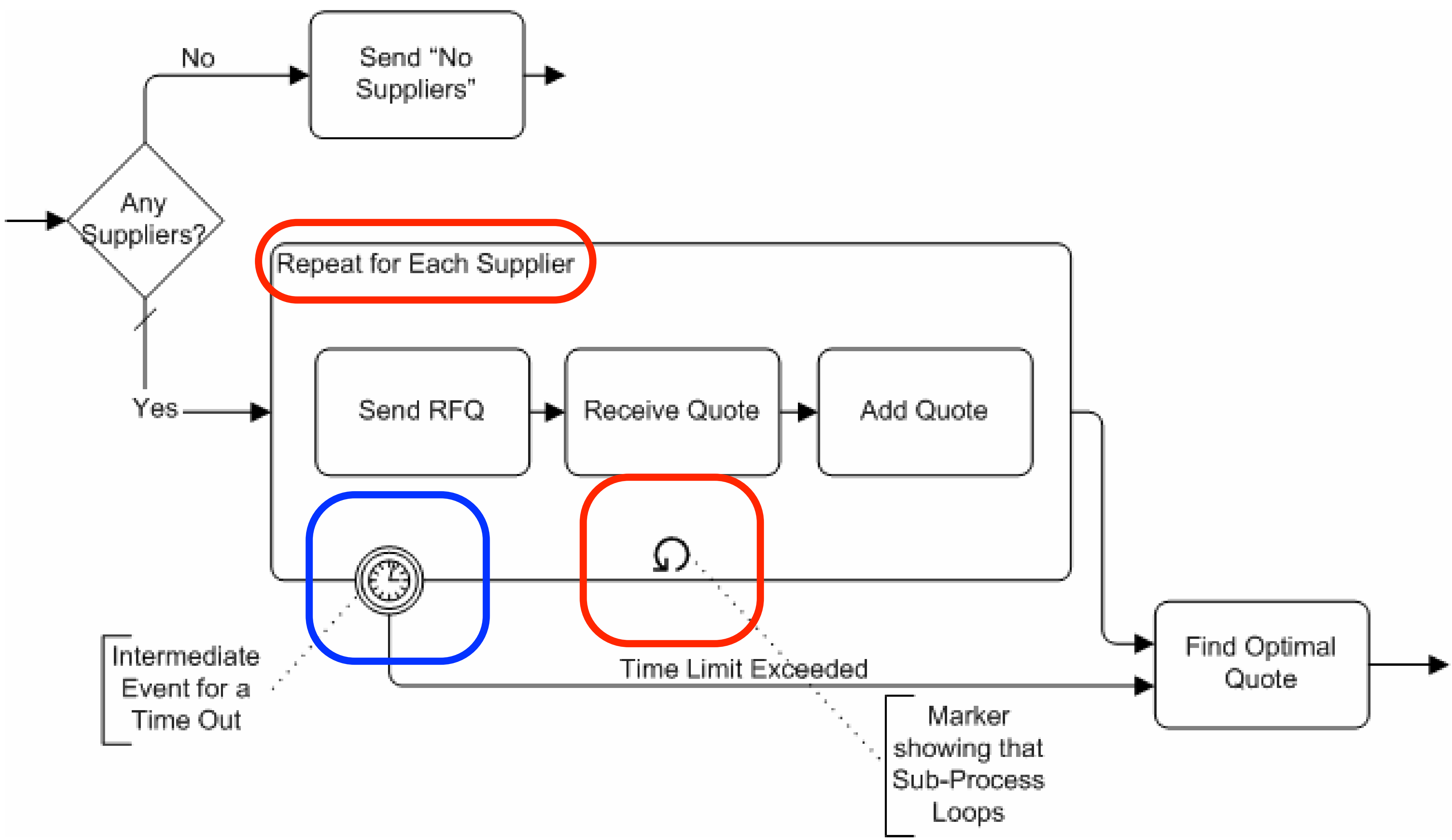
the lightning annotation denotes an error-catching event

the lightning annotation denotes a throwing event: it models an out-of-stock exception

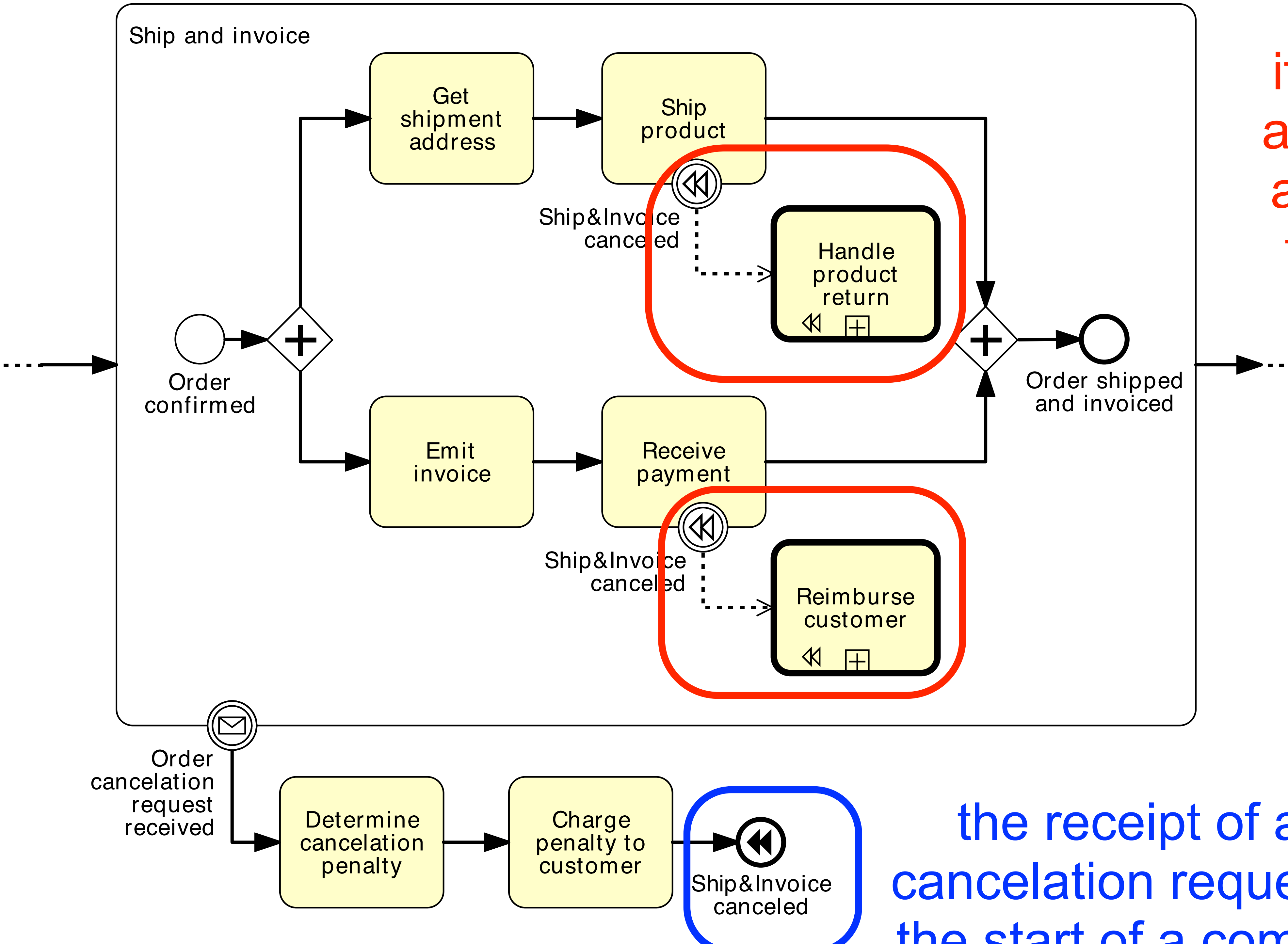
Recovery from faults: image manipulation



Intermediate time out and a loop



Compensations



if the compensable activities have been already completed, then they must be compensated

the receipt of an order cancelation request triggers the start of a compensation

Exercises

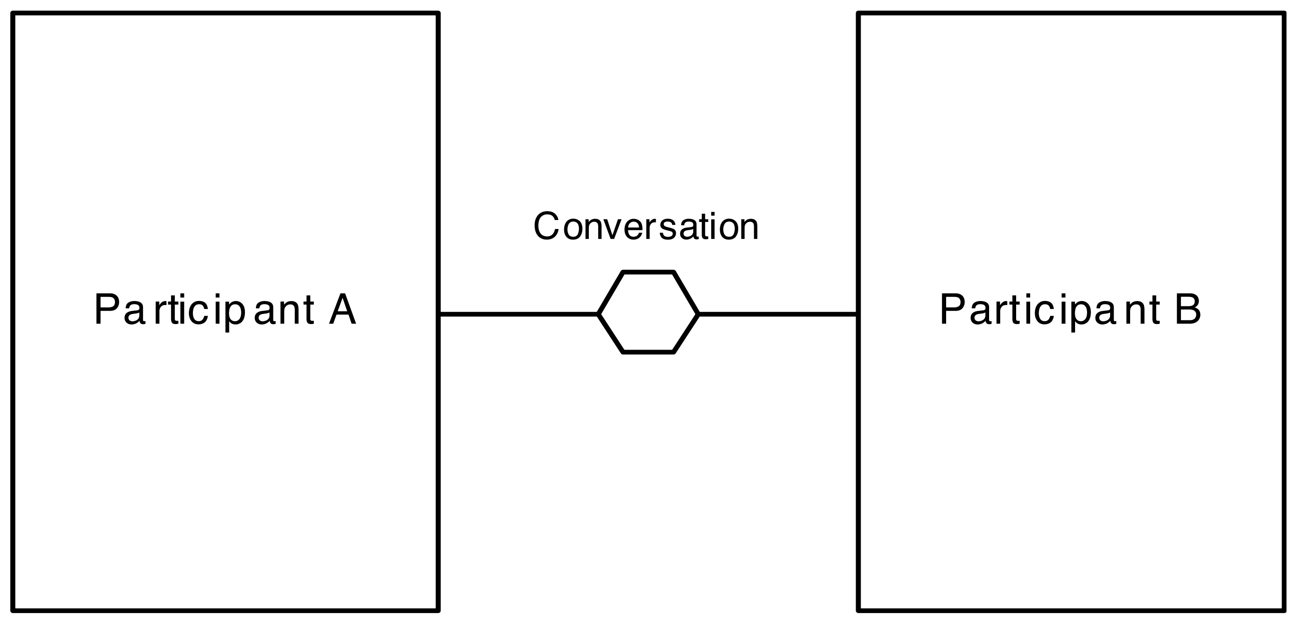
Model the following process fragment:

After a car accident, a statement is sought from two witnesses out of the five that were present, in order to lodge the insurance claim.
As soon as the first two statements are received, the claim can be lodged with the insurance company without waiting for the other statements.

Conversations, choreographies, and collaborations

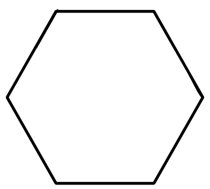
Conversation

A **Conversation** is the logical relation of (correlated) Message exchanges

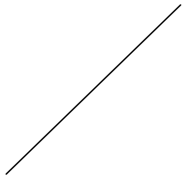


communication element

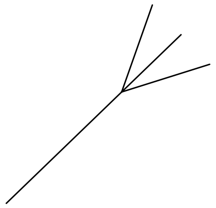
Conversation diagram



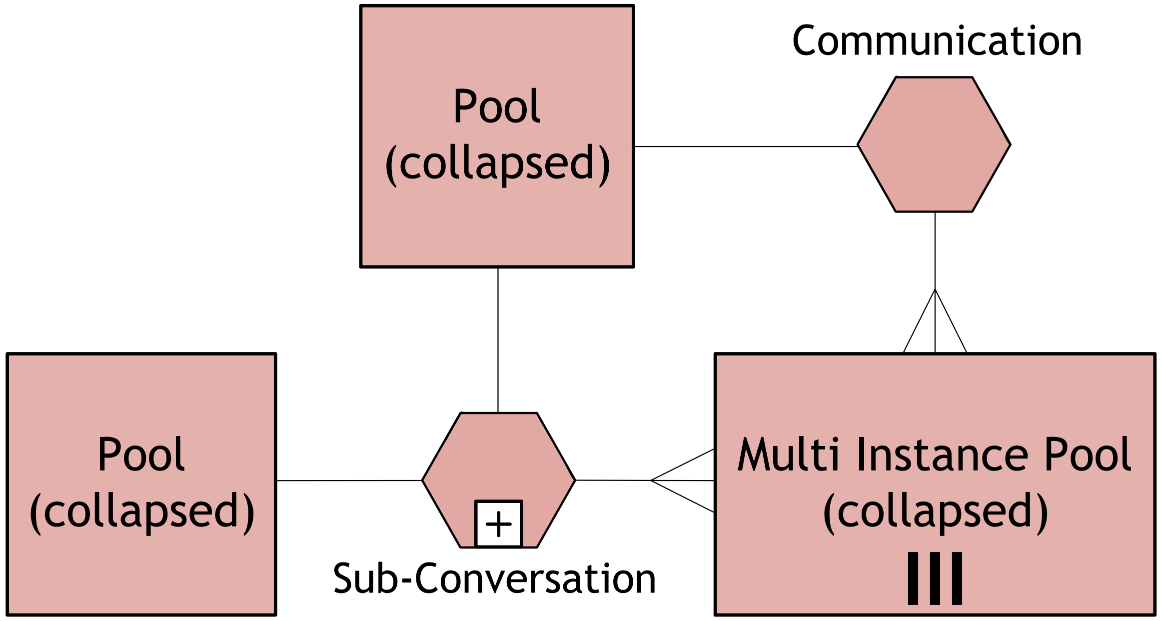
A **Communication** defines a set of logically related message exchanges. When marked with a $\boxed{+}$ symbol it indicates a Sub-Conversation, a compound conversation element.



A **Conversation Link** connects Communications and Participants.



A **Forked Conversation Link** connect Communications and multiple Participants.



Choreography

The behaviour of different Conversations is modelled through separate Choreographies

A **Choreography** defines the sequence of interaction between participants

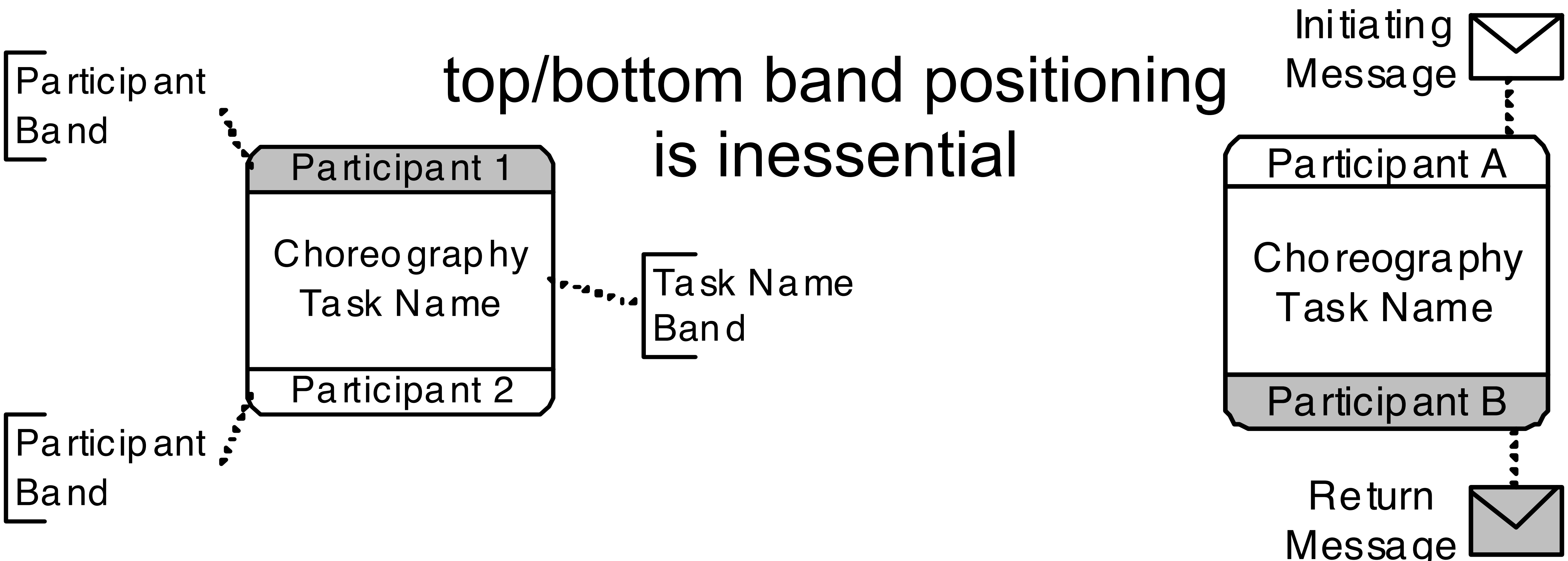
A choreography does not exists in a pool and it is not executable

It describes how the participants are supposed to behave

Choreography task

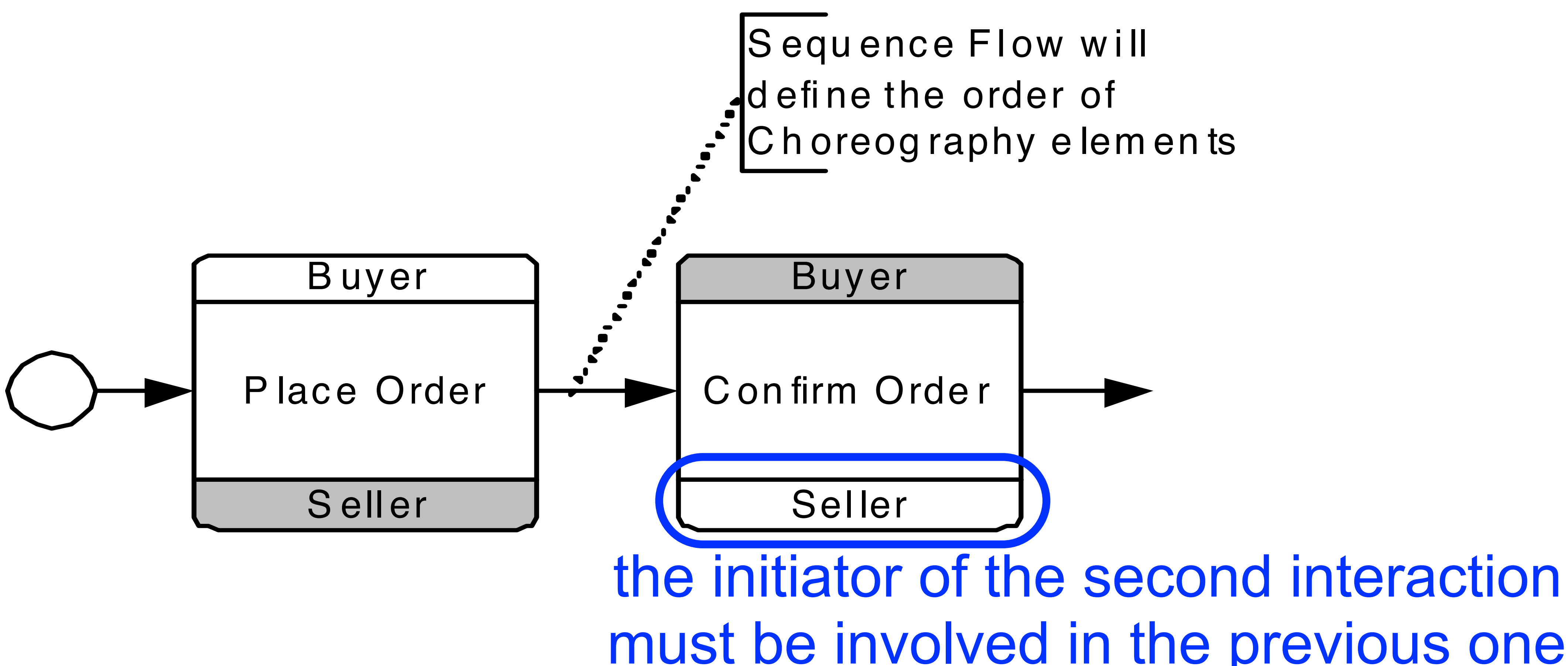
A **Choreography task** is an activity in a choreography that consists of a set (one or more) Message exchanges

A choreography task involves two or more participants that are displayed in different bands



Sequence flow in a choreography

Sequence Flow are used within Choreographies to show the sequence of the Choreography Activities, Events, and Gateways



Collaboration

A **Collaboration** contains two or more Pools, representing the Participants in the Collaboration

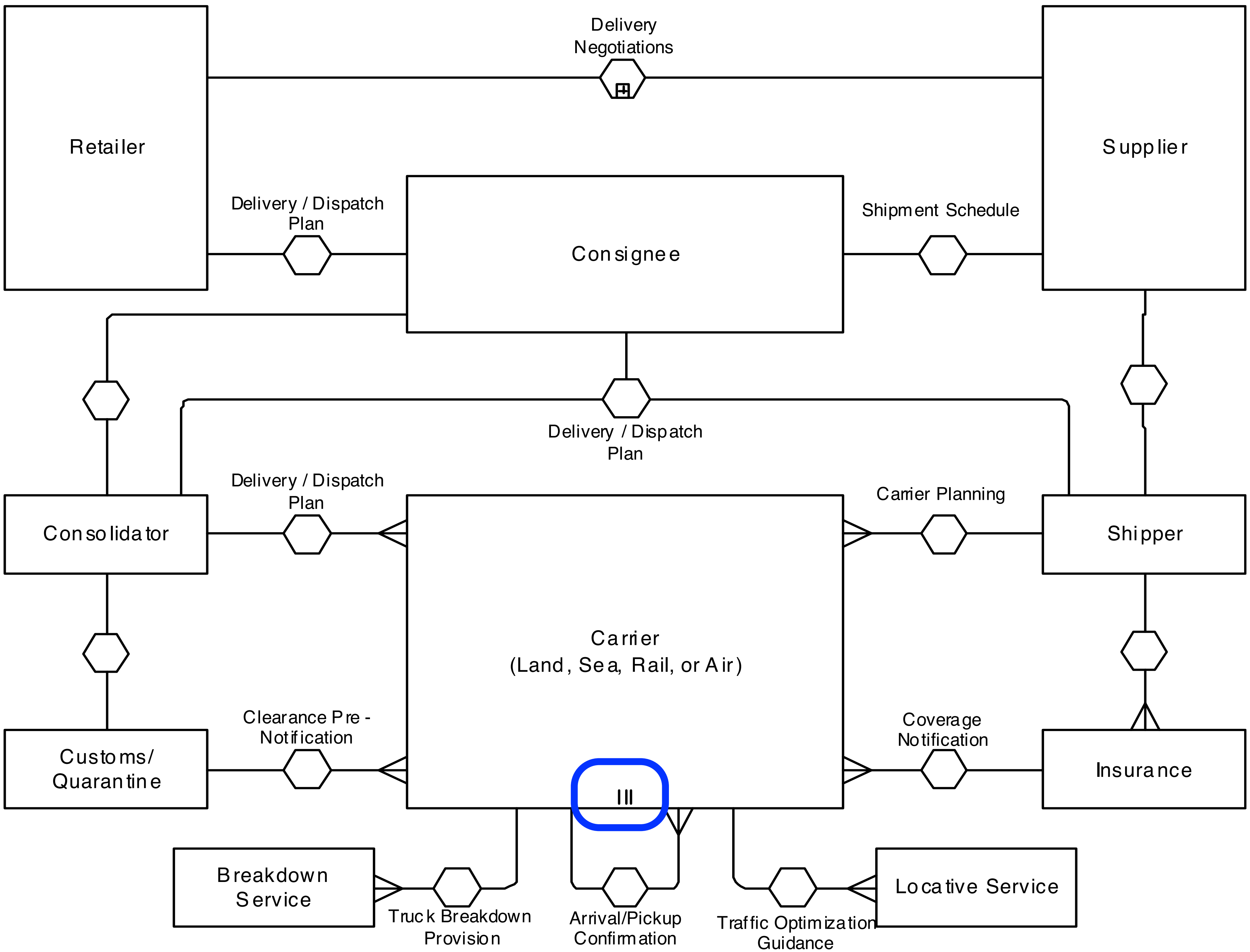
A Pool may be empty or show a Process within

The Message exchange is shown by a Message Flow that connects Pools or the objects within the Pools The Messages associated with the Message Flow may also be shown

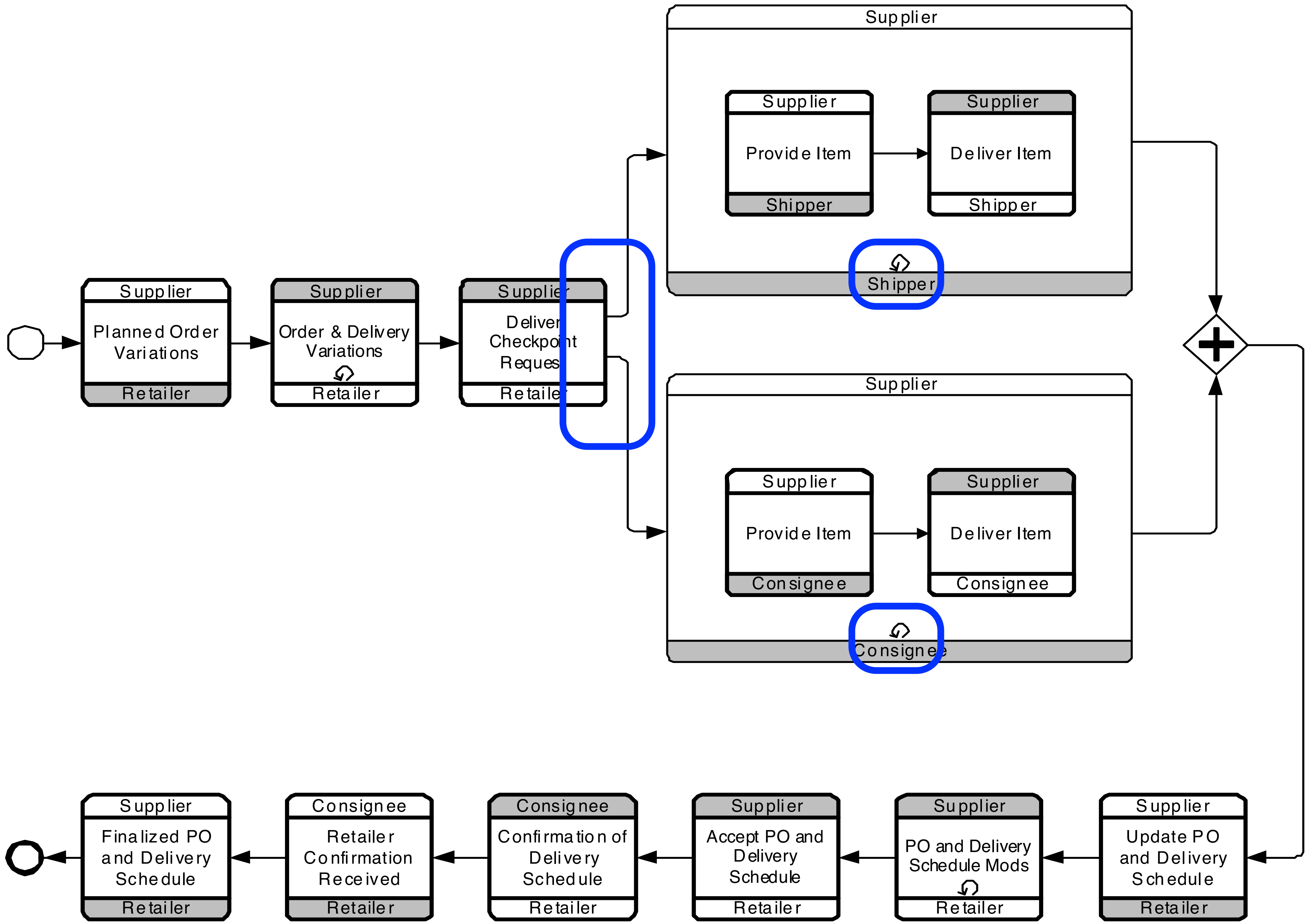
Choreographies may be shown “in between” the Pools as they bisect the Message Flow

Examples
(a taste of BPMN)

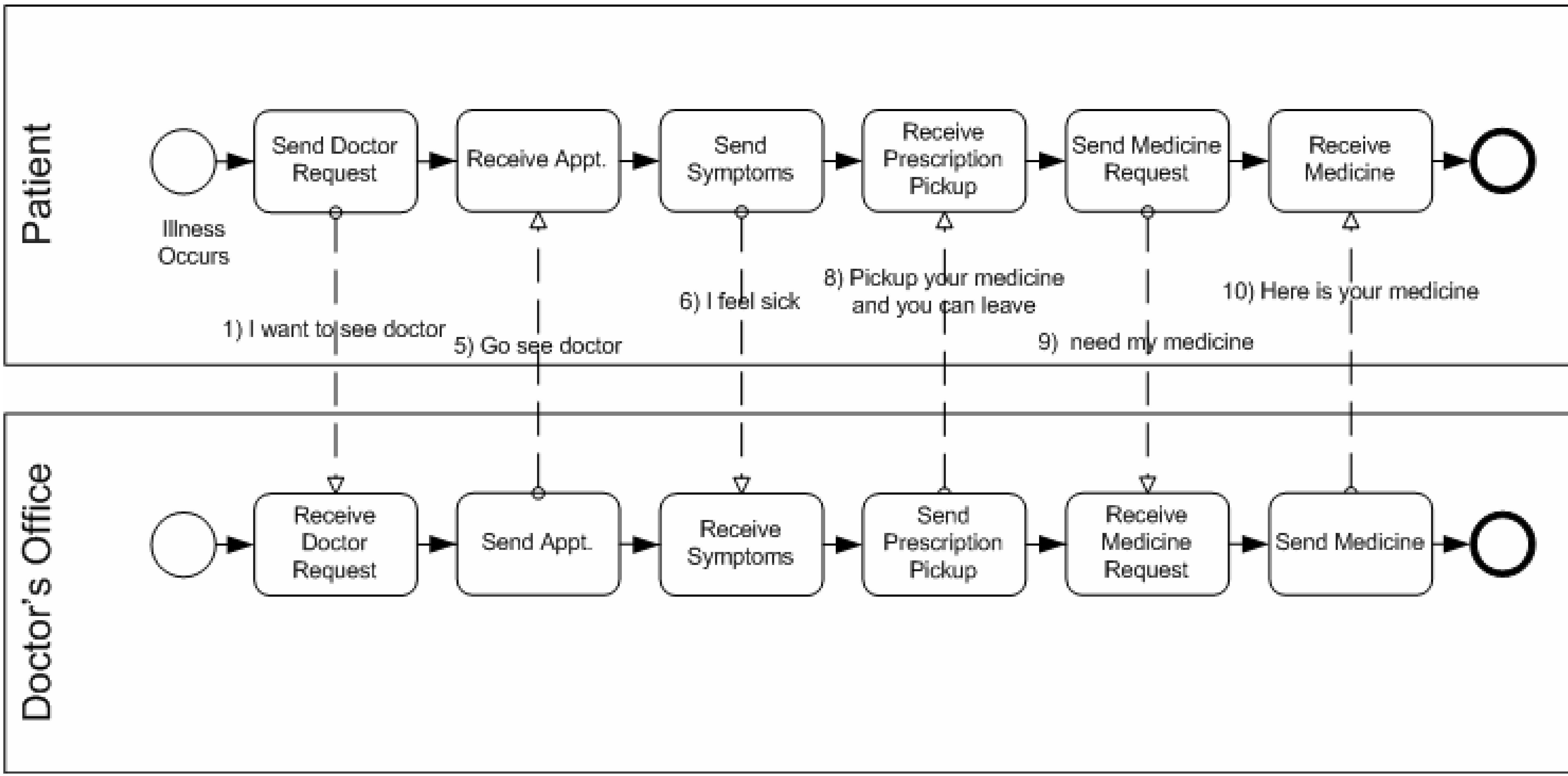
A conversation



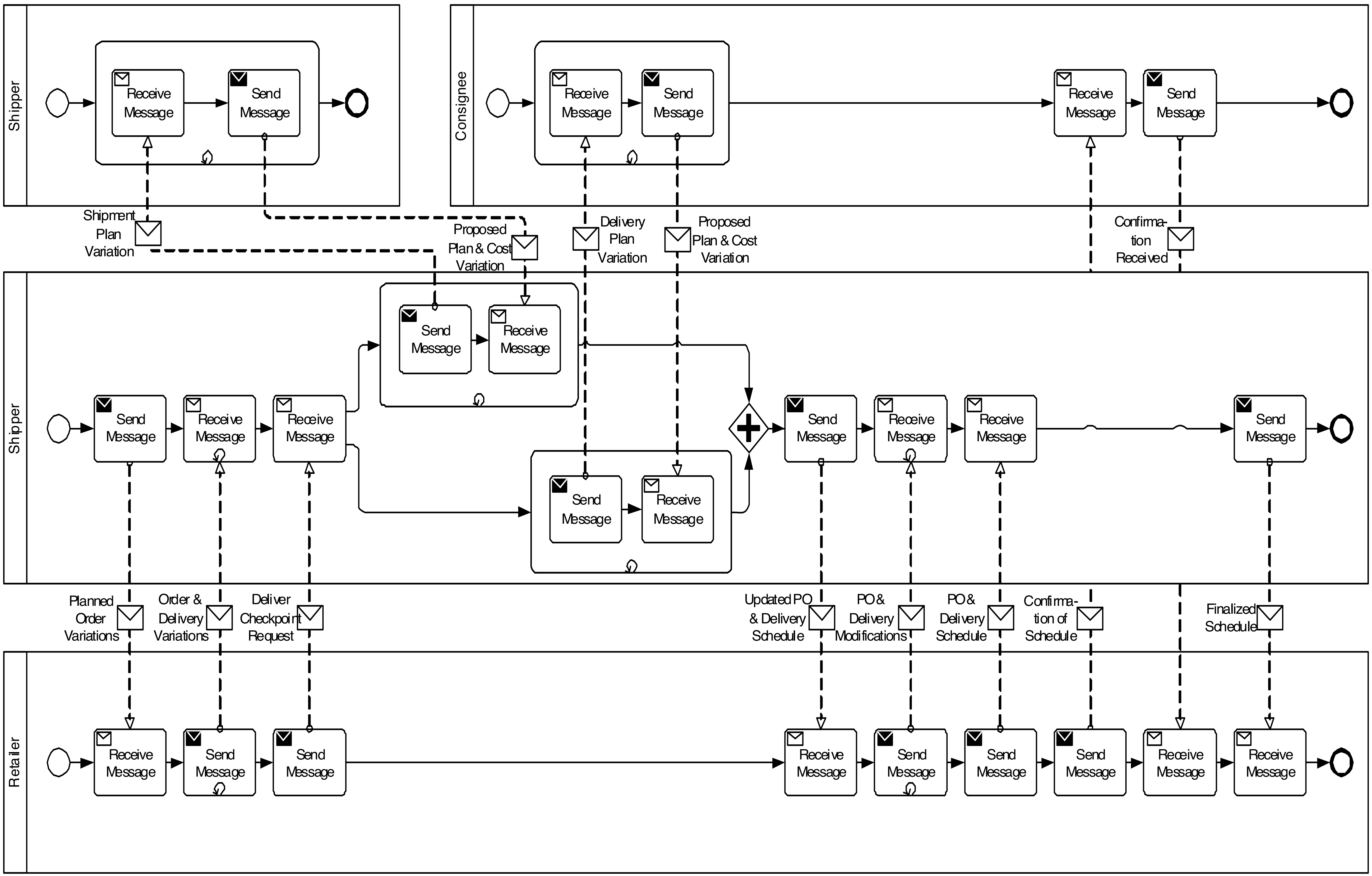
A choreography



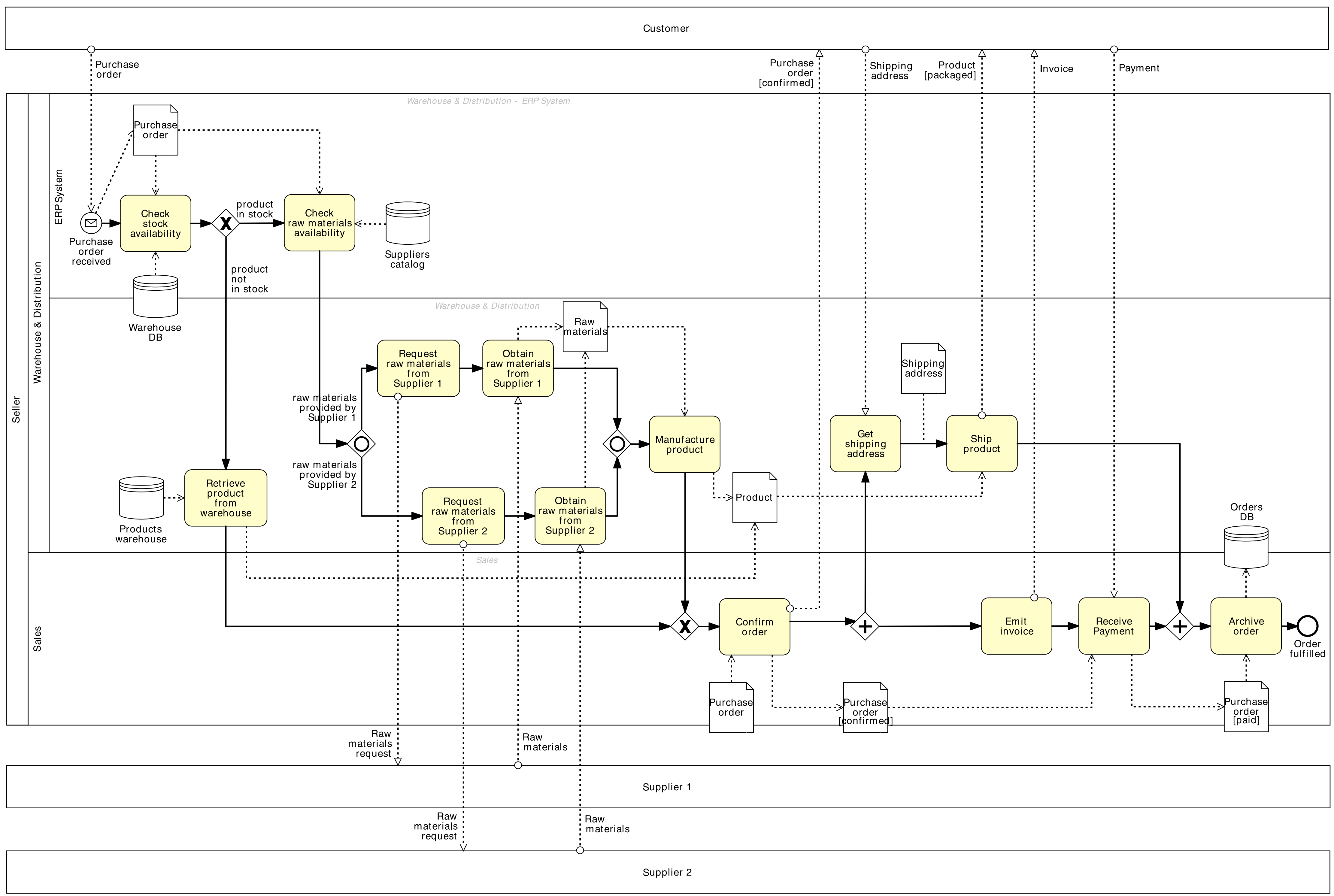
A collaboration with two pools



A collaboration diagram



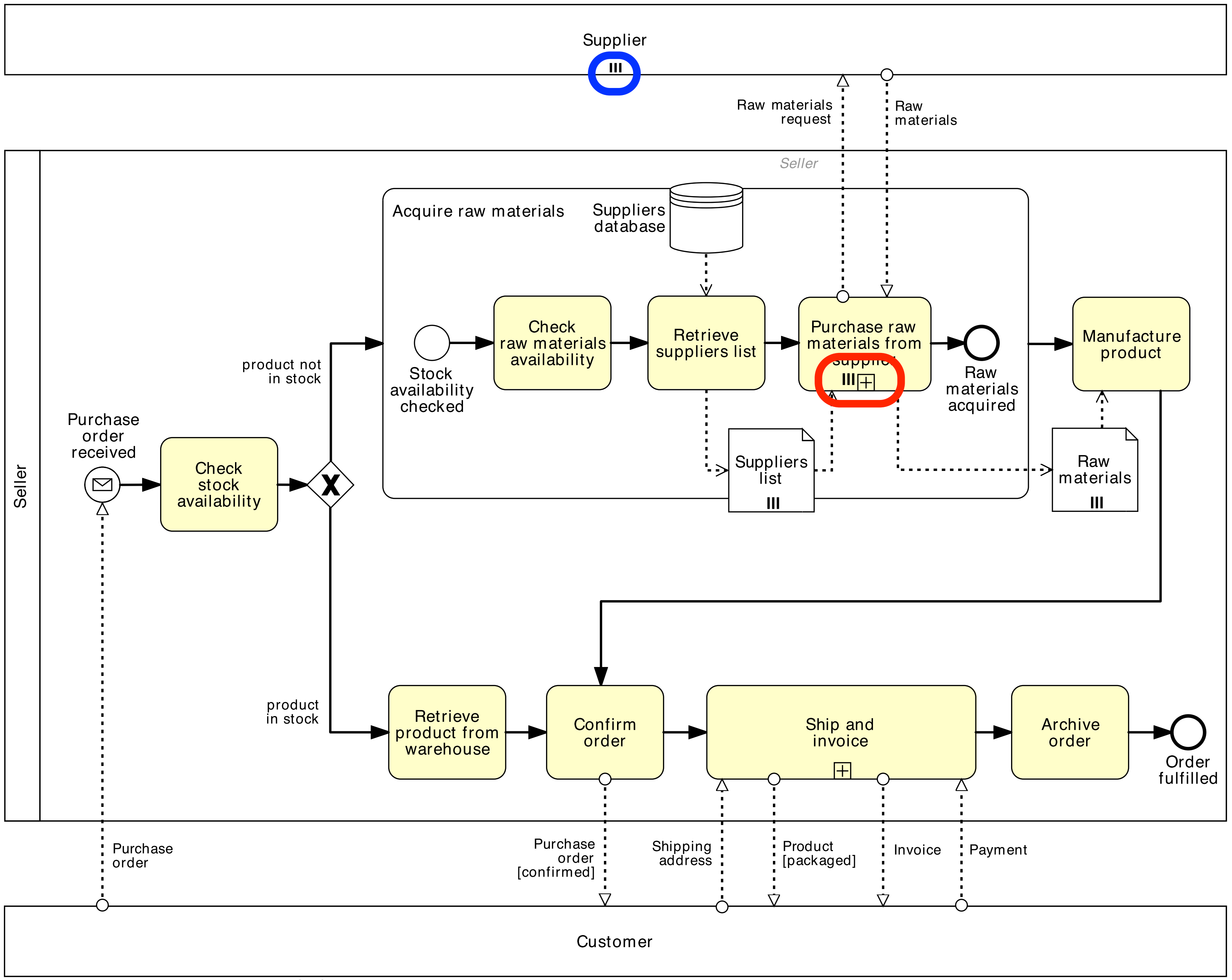
A collaboration diagram: order fulfillment



Multi-instance pools: order fulfillment

the multi-instance
symbol annotation
denotes
a set of resources
with similar characteristics

multi-instance
sub-process



Exercises: loan application

Extend the loan application model by representing the interactions between the loan provider and the applicant.

From BPMN to Petri nets

ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Information and Software Technology 50 (2008) 1281–1294

INFORMATION AND SOFTWARE TECHNOLOGY

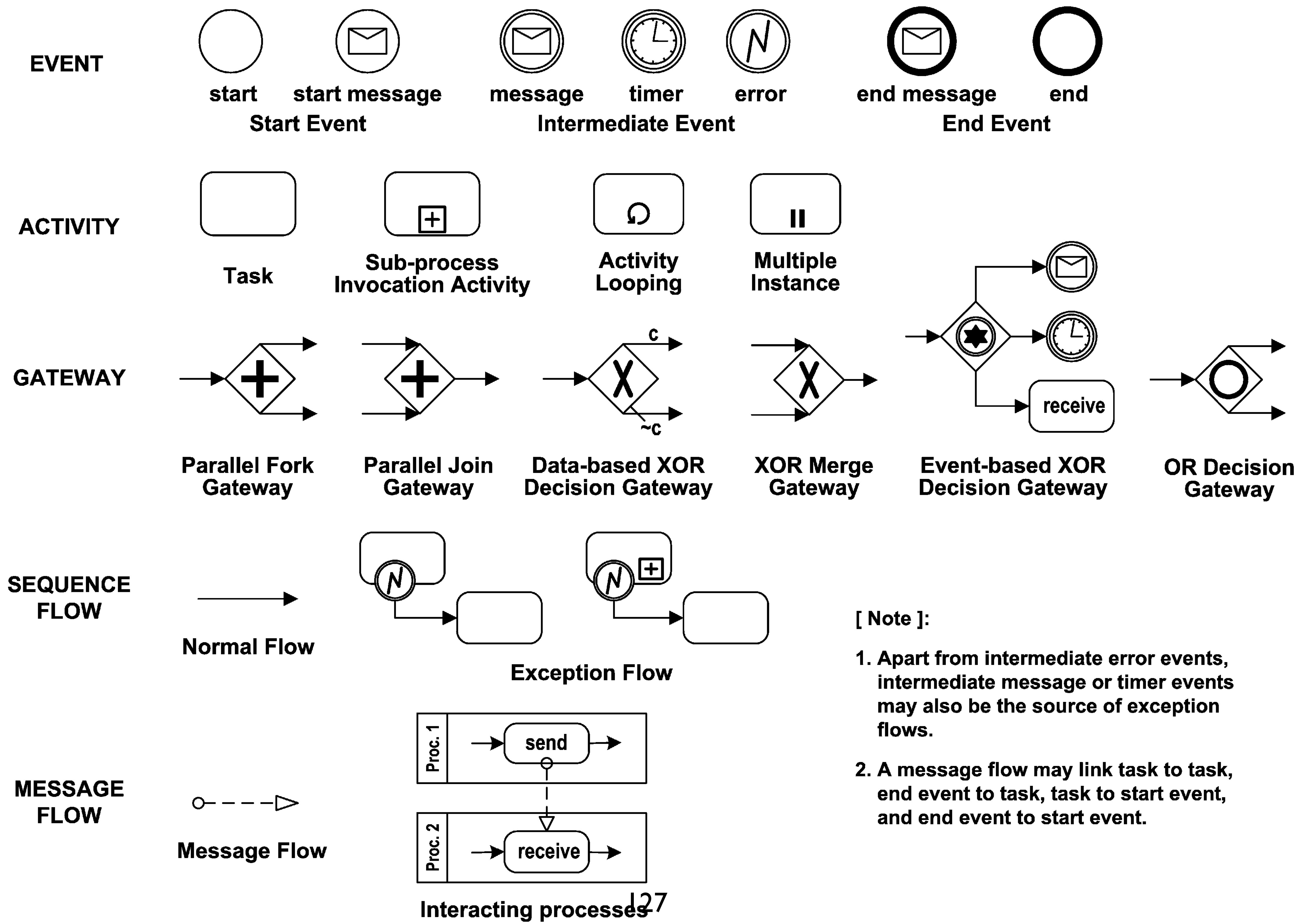
www.elsevier.com/locate/infsof

Semantics and analysis of business process models in BPMN

Remco M. Dijkman [a], Marlon Dumas [b,c], Chun Ouyang [c,*]

[a] *Department of Technology Management, Eindhoven University of Technology, P.O. Box 513, 5600 MB, The Netherlands*
[b] *Institute of Computer Science, University of Tartu, J Liivi 2, Tartu 50409, Estonia*
[c] *Faculty of Information Technology, Queensland University of Technology, G.P.O. Box 2434, Brisbane, Qld 4001, Australia*

Overview



[Note]:

1. Apart from intermediate error events, intermediate message or timer events may also be the source of exception flows.
2. A message flow may link task to task, end event to task, task to start event, and end event to start event.

Simplified BPMN

a start / exception event has just one outgoing flow
and no incoming flow

an end event has just one incoming flow
and no outgoing flow

all activities and intermediate events have exactly
one incoming flow and one outgoing flow

all gateways have either
one incoming flow (and multiple outgoing)
or one outgoing flow (and multiple incoming)

Simplified BPMN

The previous constraints are no real limitation:

events or activities with multiple incoming flows:
insert a preceding XOR-join gateway

events or activities with multiple outgoing flows:
insert a following AND-split gateway

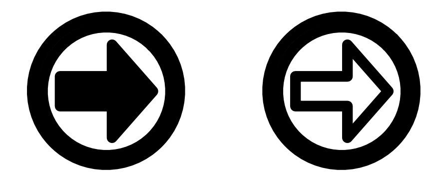
gateways with multiple incoming and outgoing flows:
decompose in two gateways

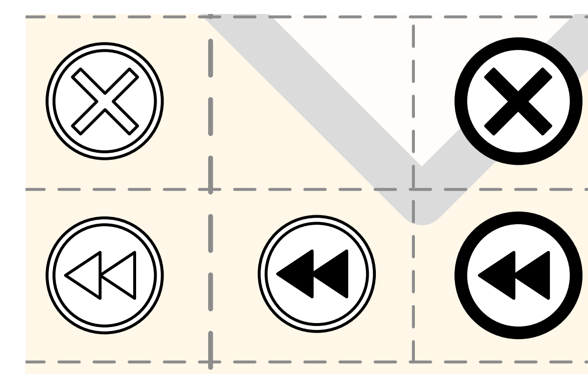
insert start / end event if needed

Simplified BPMN

No link events

they are just a notational convenience
to spread a model into several pages
(no effect on the semantics)



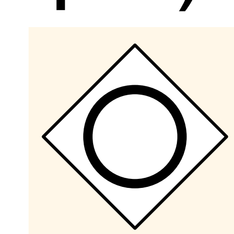


No transactions and compensations

Limited form of sub-processing

no OR-split

(can be expressed in terms of AND-split and XOR-split)



no OR-join

Roughly

A place for each arc

one transitions for each event

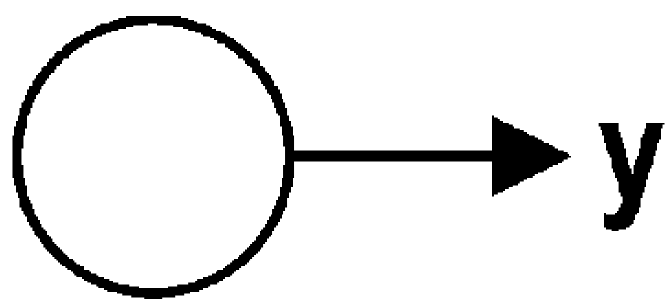
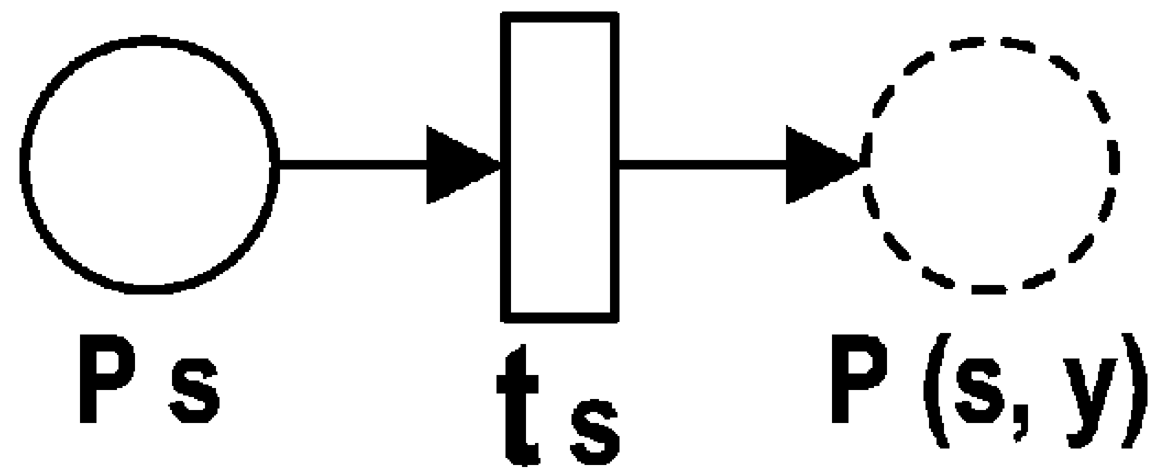
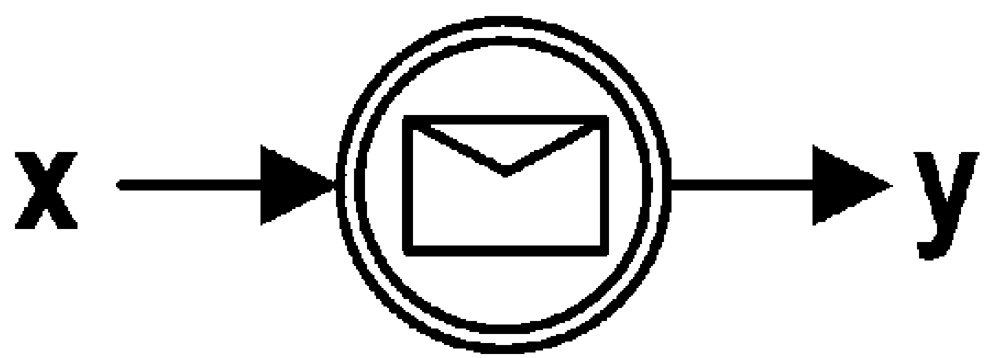
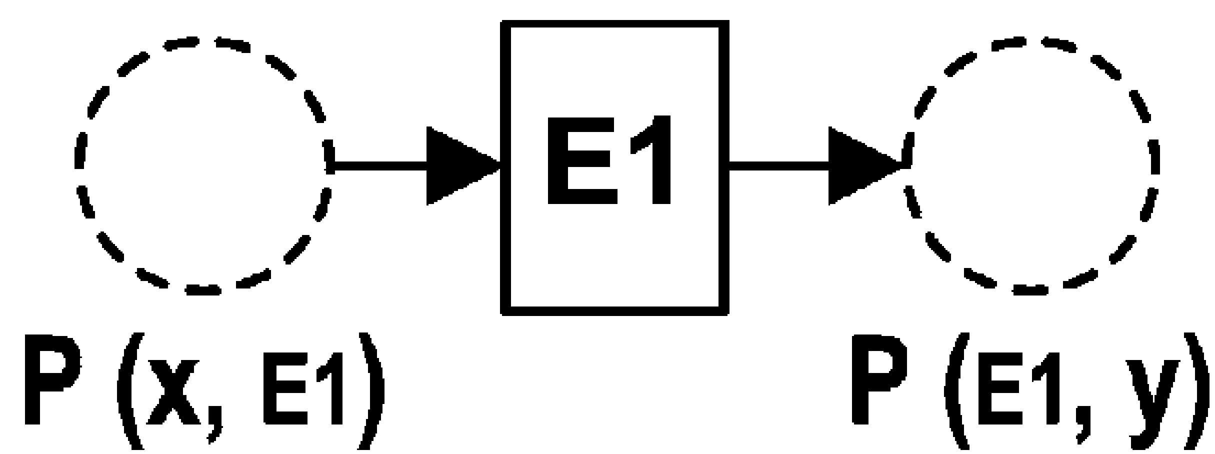
one transition for each activity

one or two transitions for each gateway

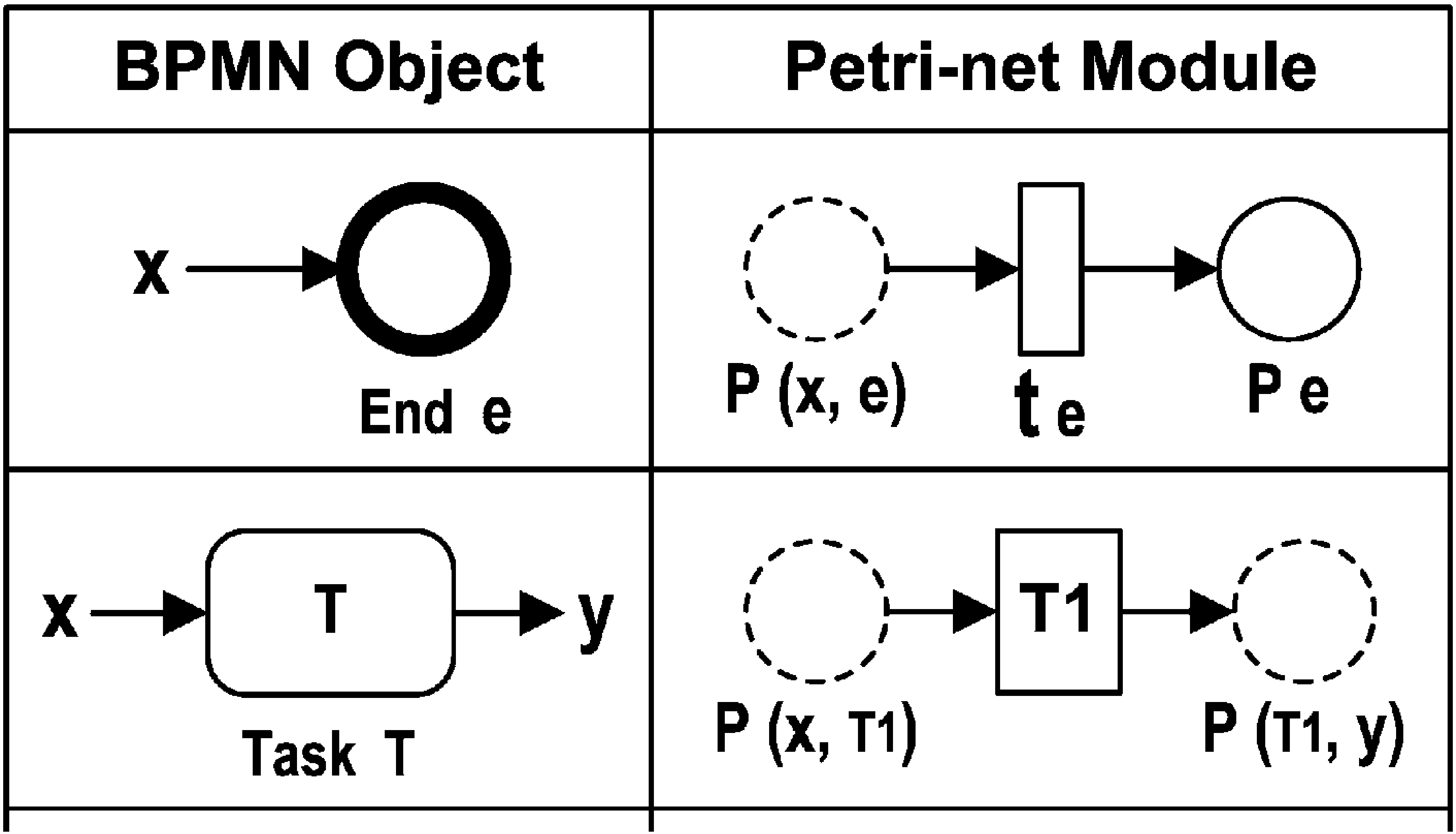
…

with some exceptions!
(start event, end event, event-based gateways, loops, …)

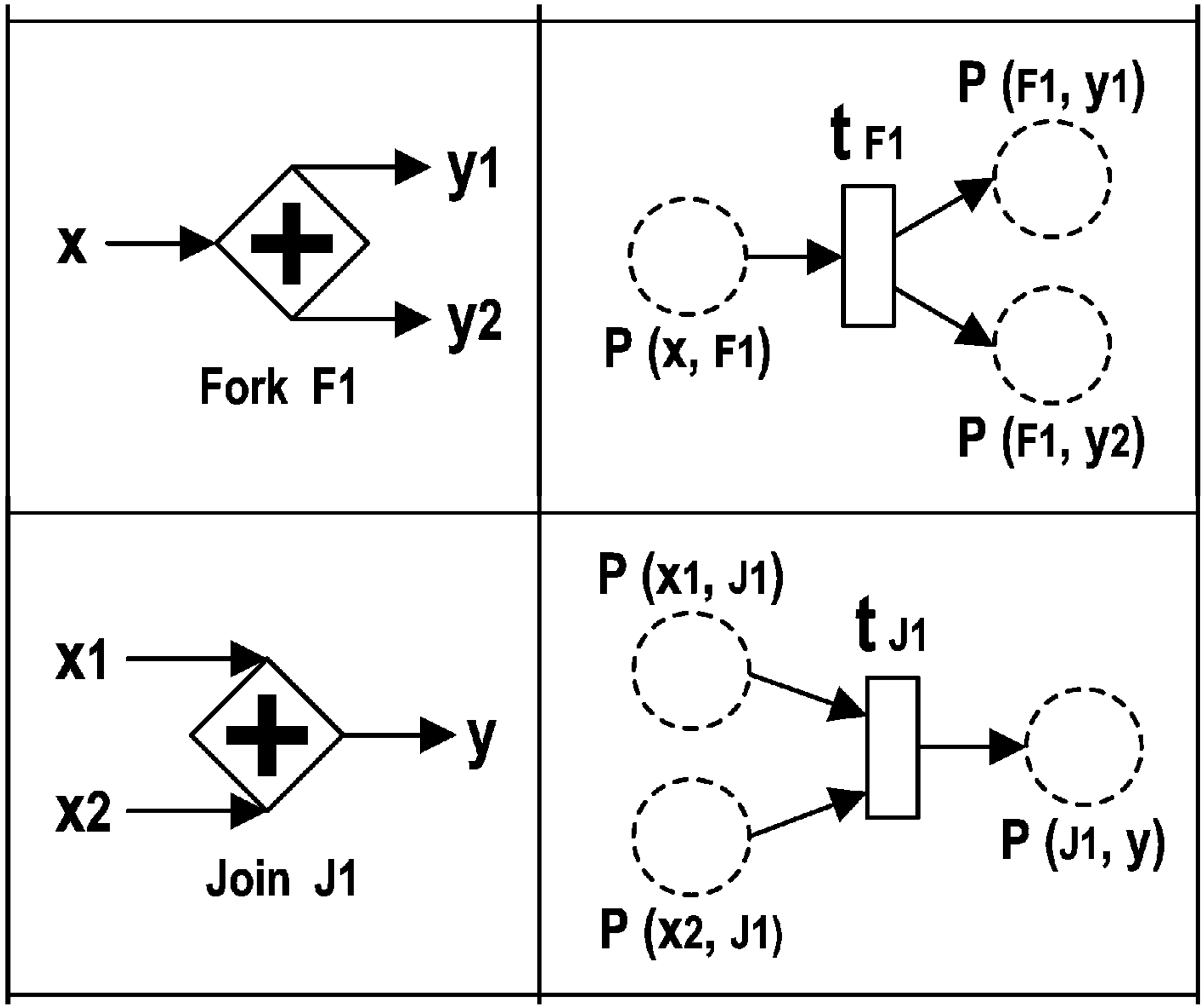
Zoom in: start, intermediate

BPMN Object	Petri-net Module
 Start s	
 Message E	

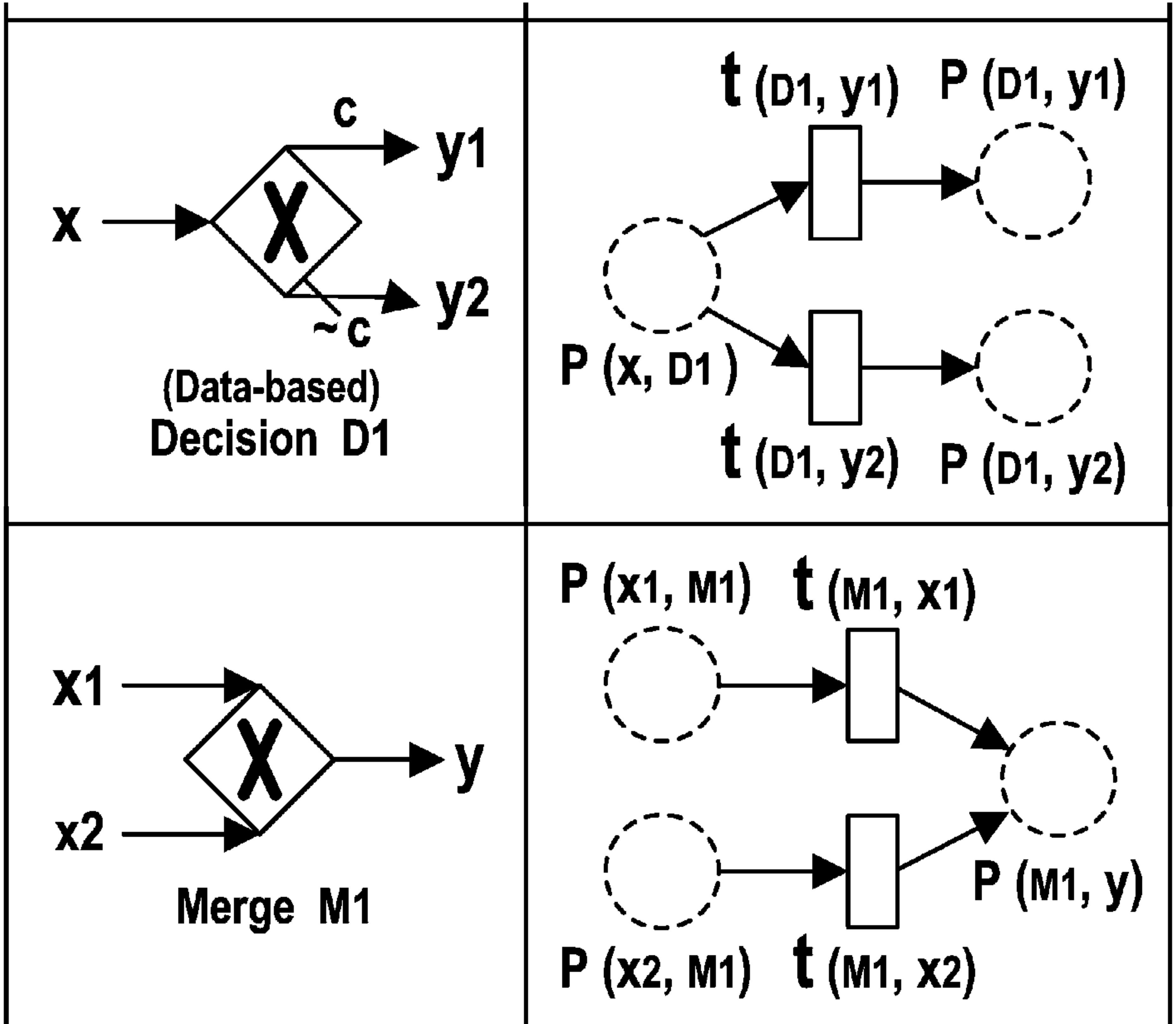
Zoom in: end event, task



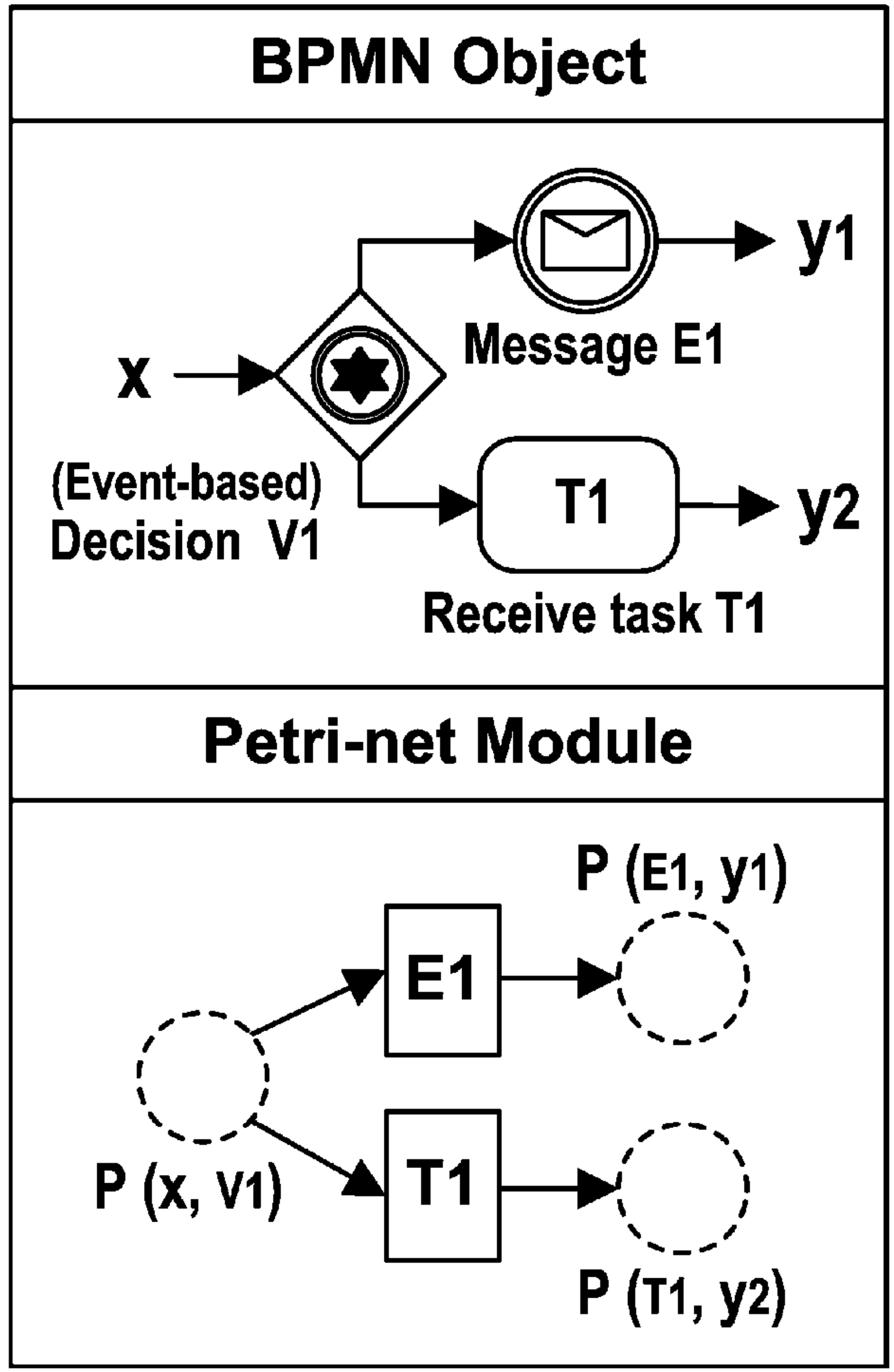
Zoom in: parallel gateway



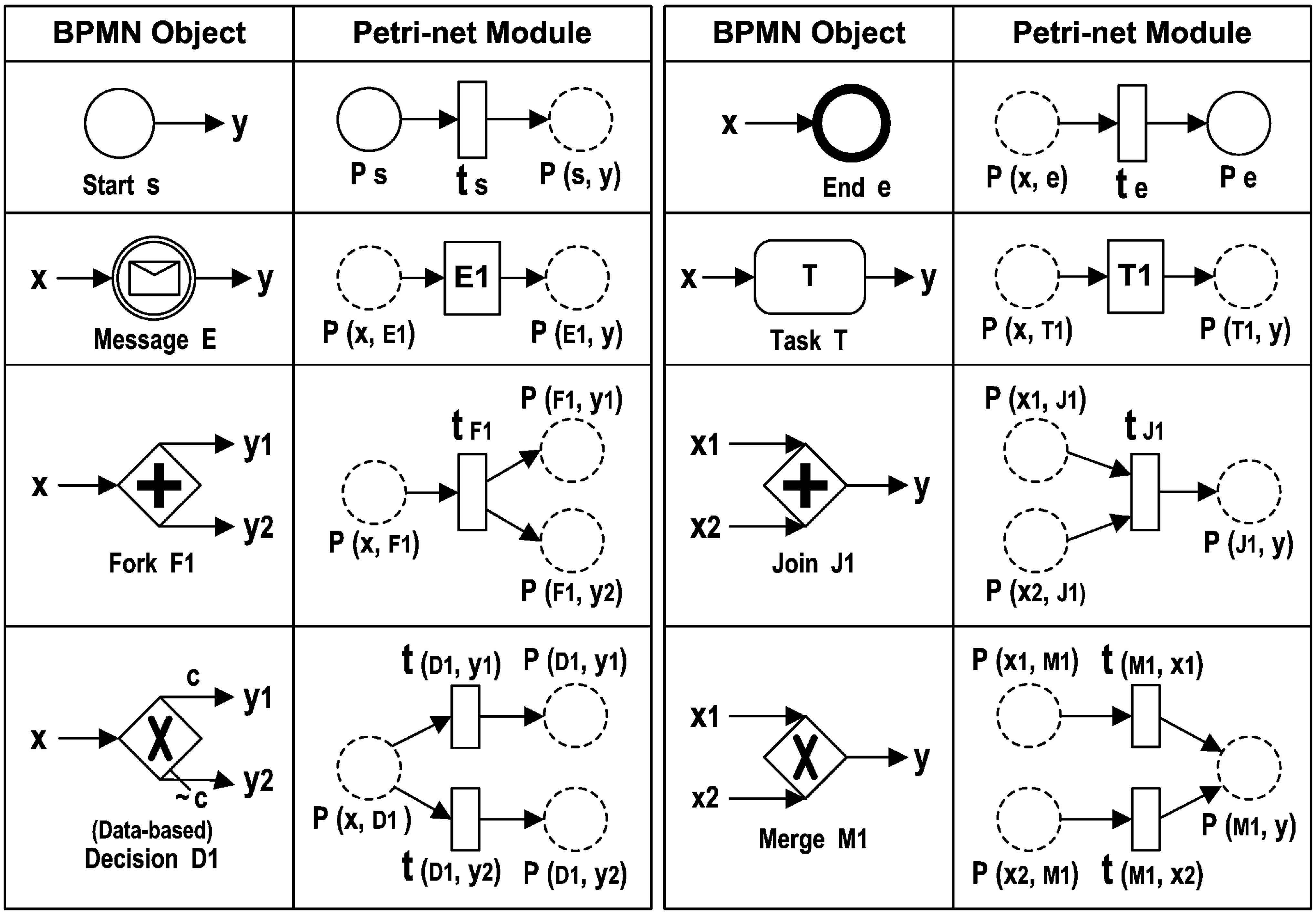
Zoom in: choice gateway

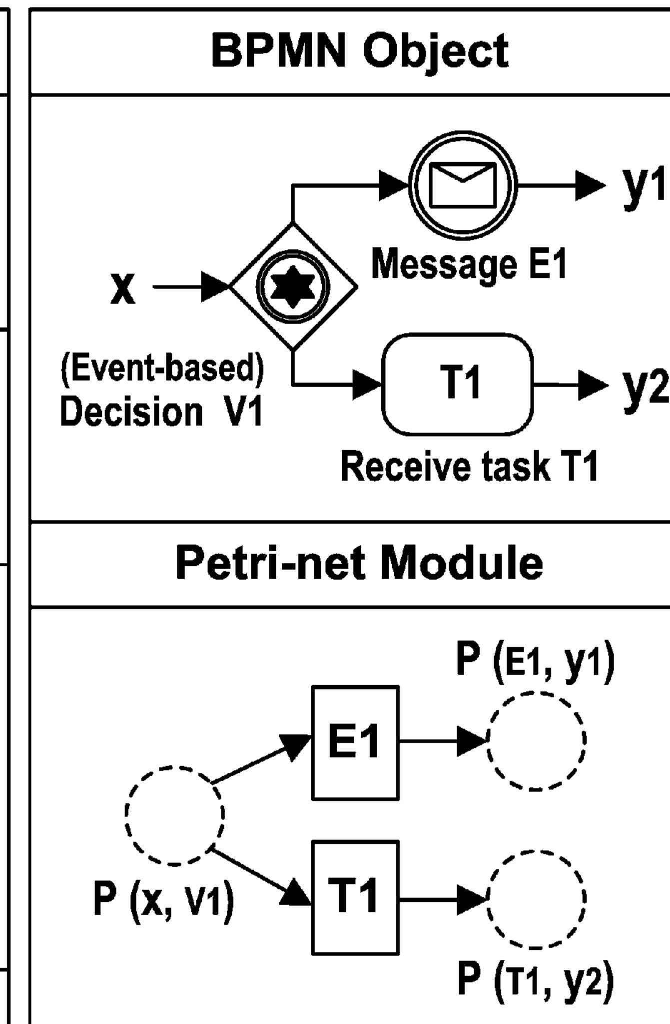


Zoom in: event based gateway



Task, events and gateways as nets

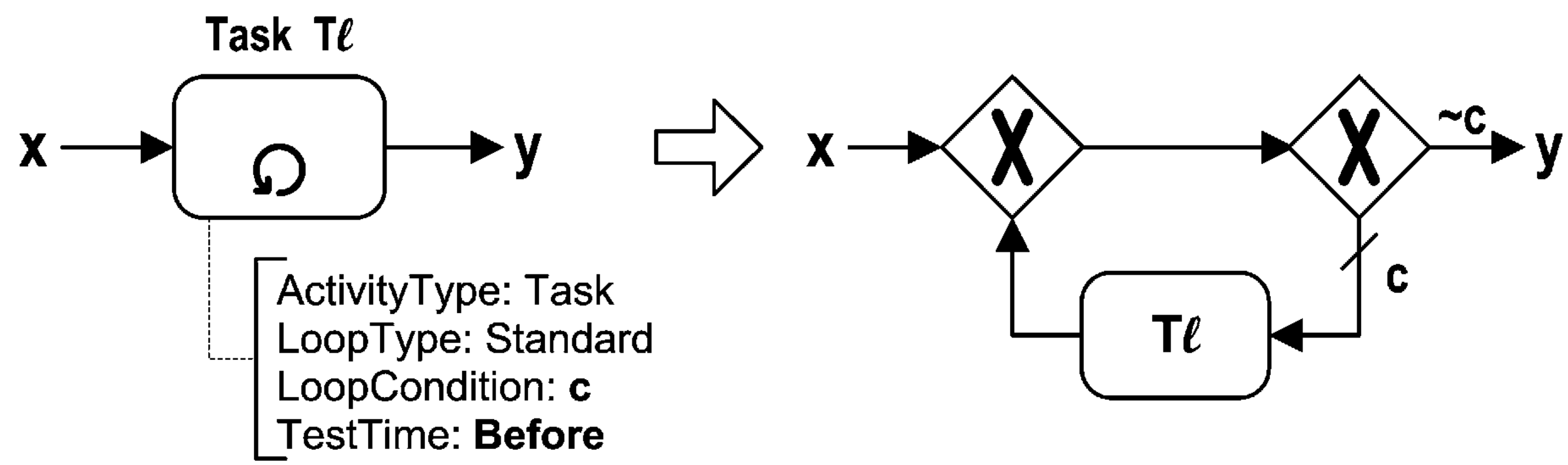




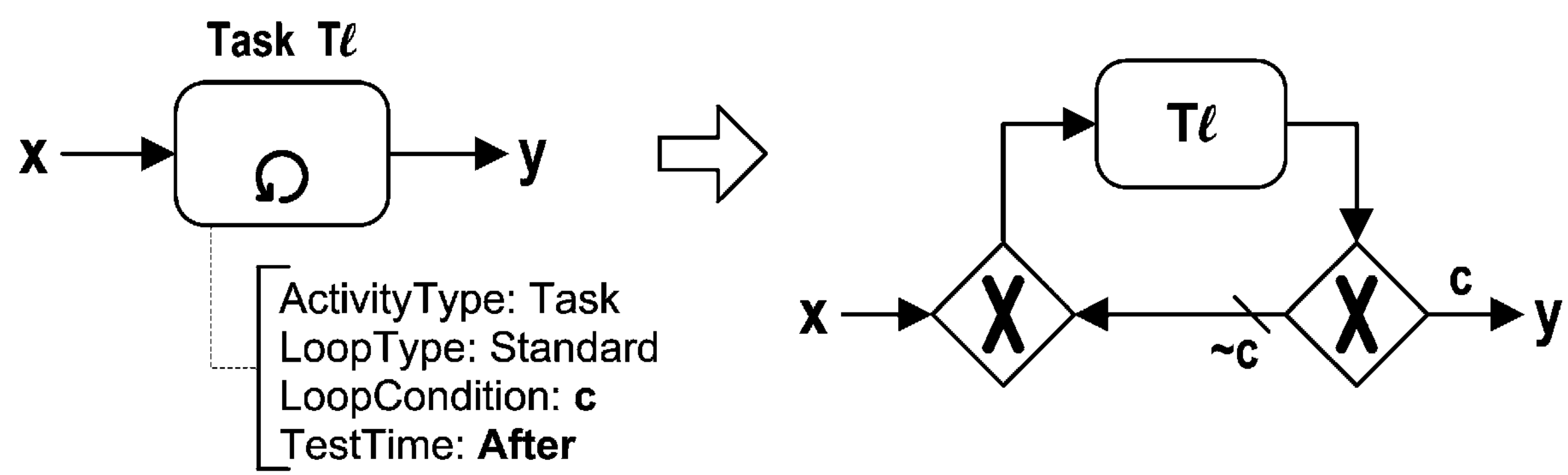
[Note]:

x, x1 or x2 represents an input object, and y, y1 or y2 represents an output object.

Activity looping

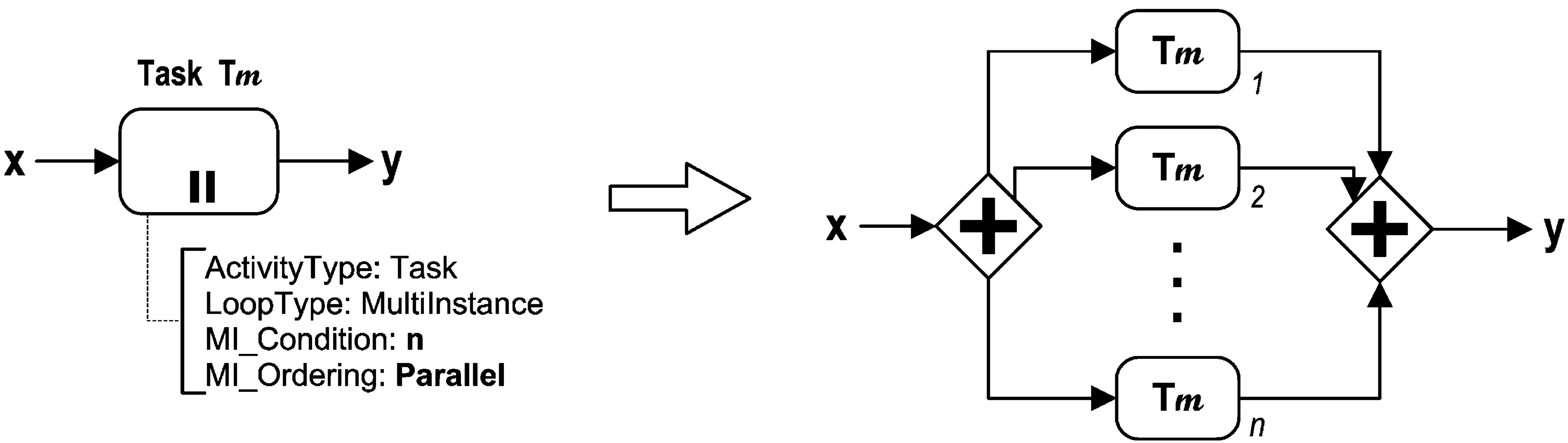


(a) “while-do” loop

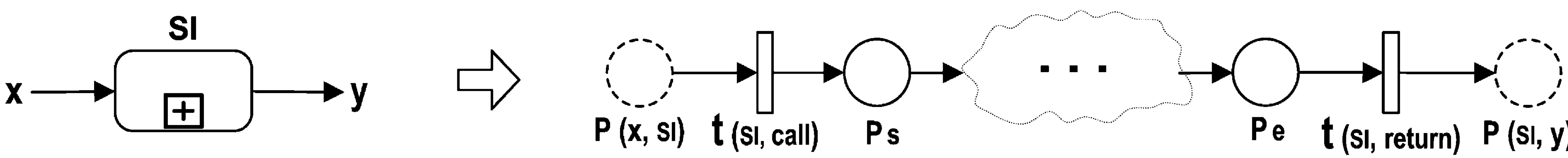


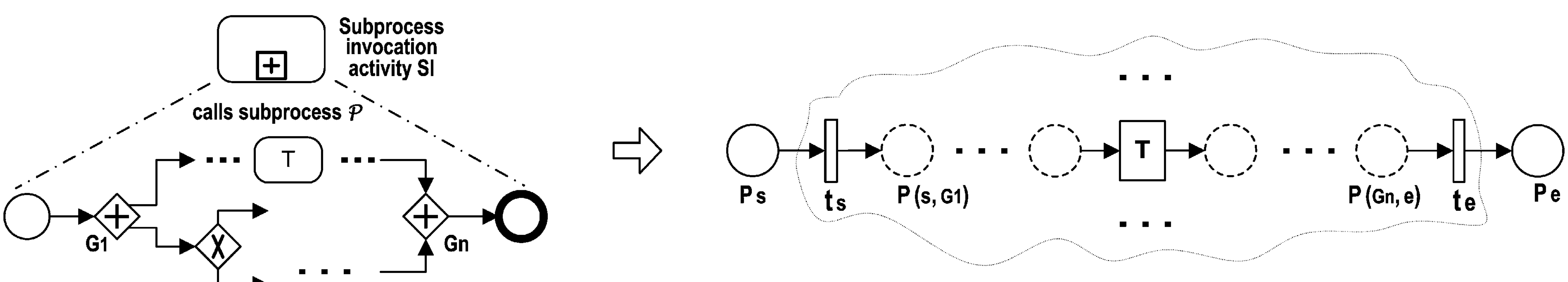
(b) “do-until” loop

Multiple instances (design-time bounded)

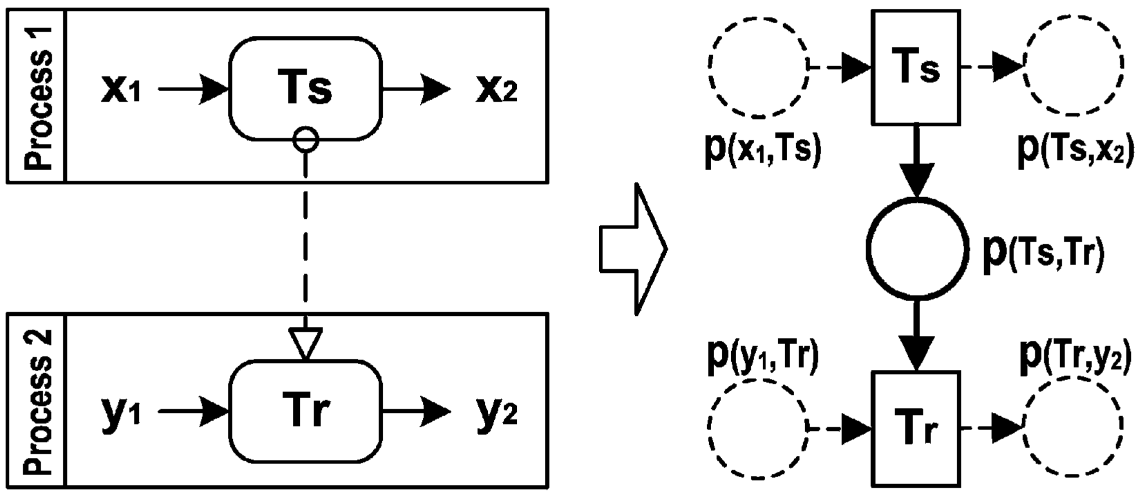


Sub-processes

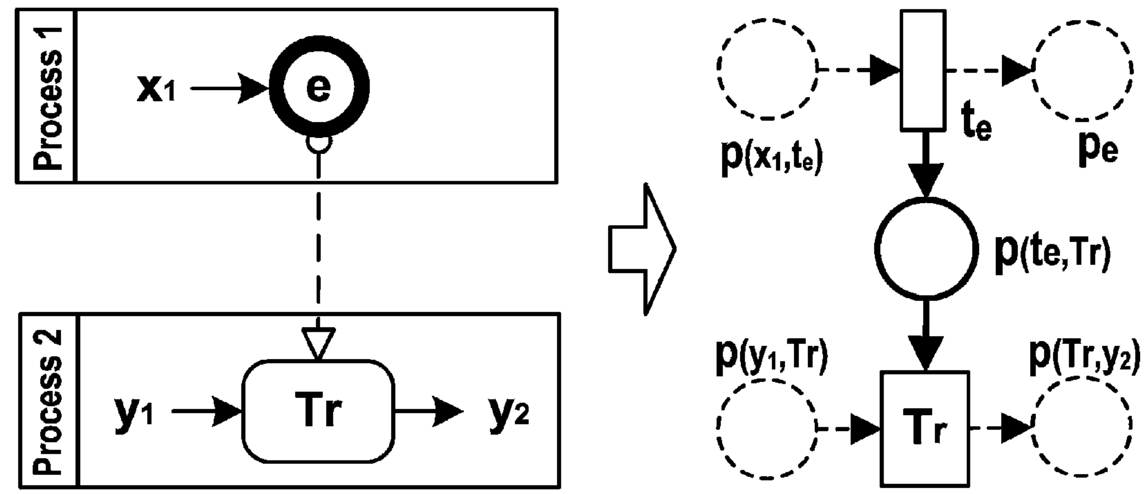




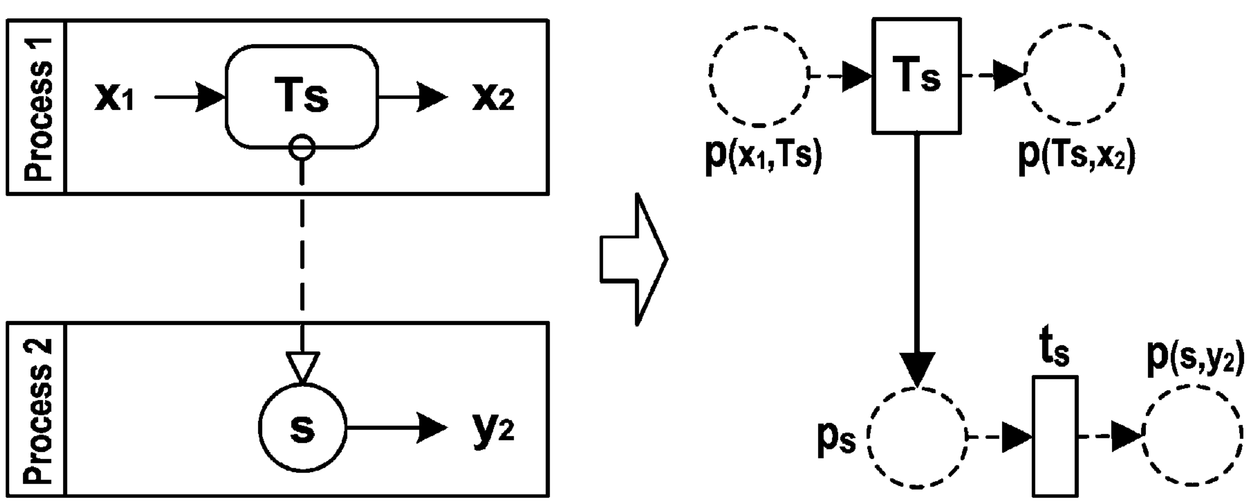
Message flow



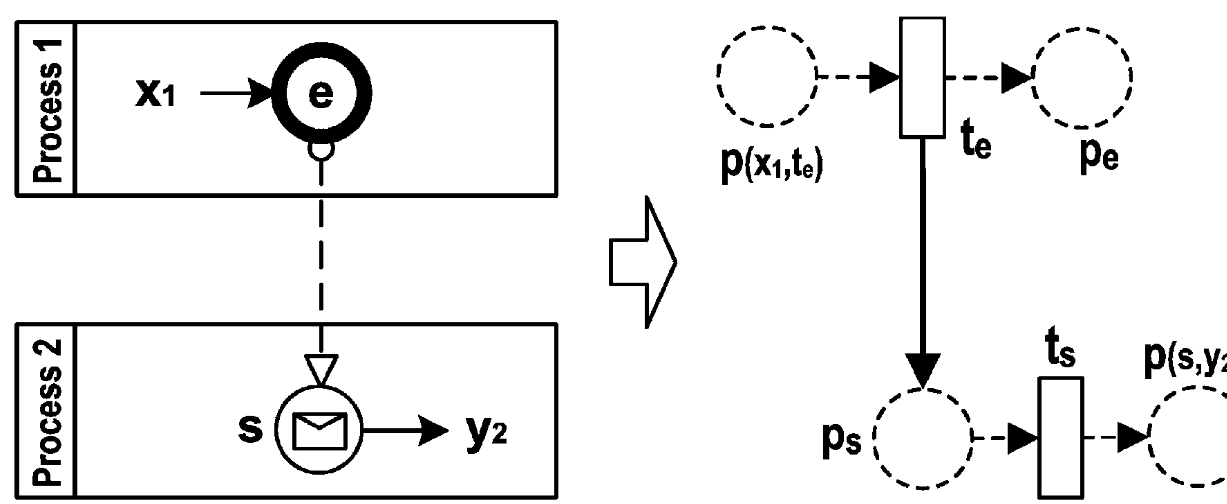
(a) task to task



(b) end event to task

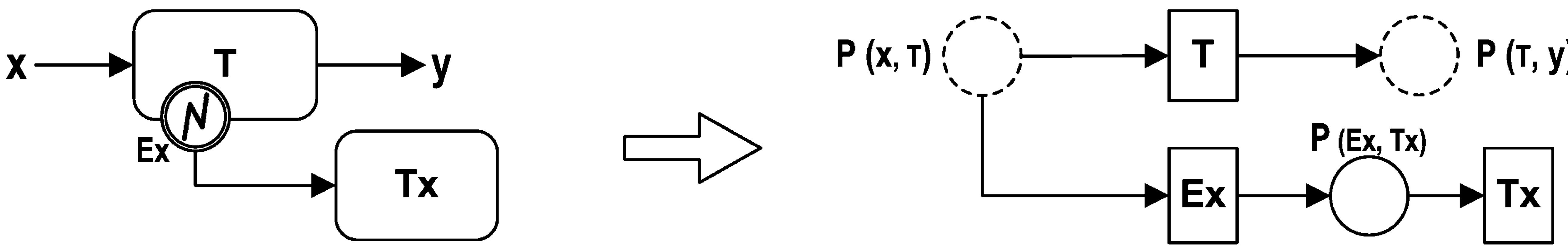


(c) task to start event

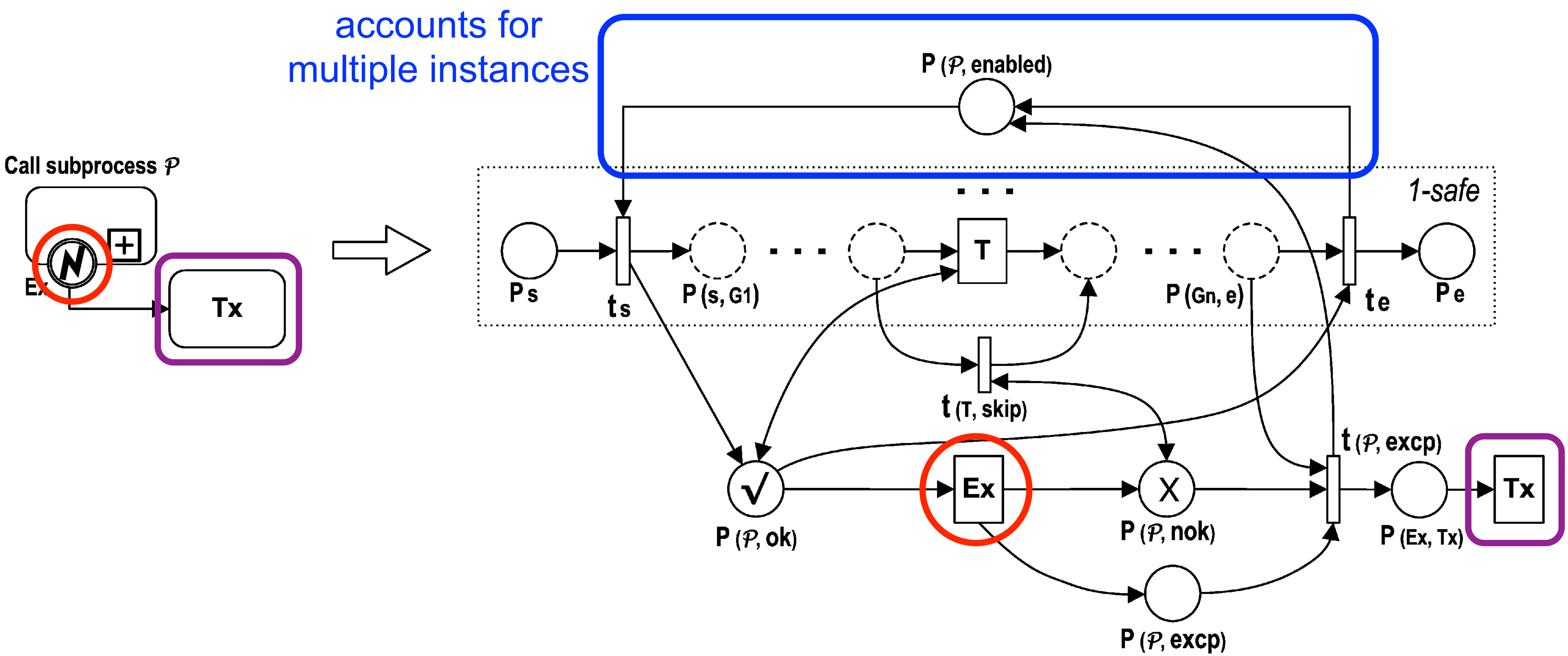


(d) end event to start event

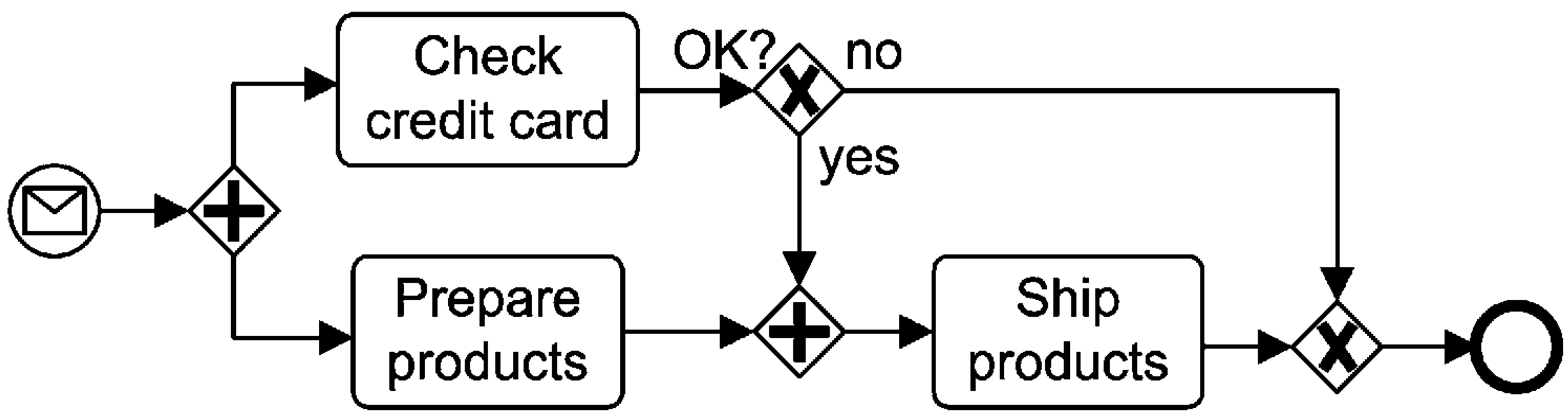
Exception handling: single task

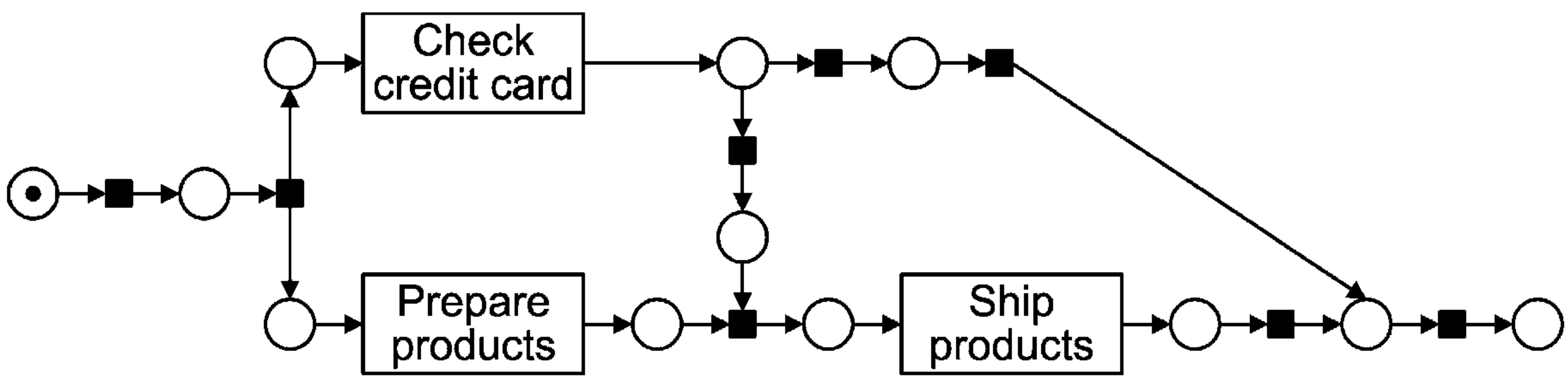


Exception handling: sub-processes

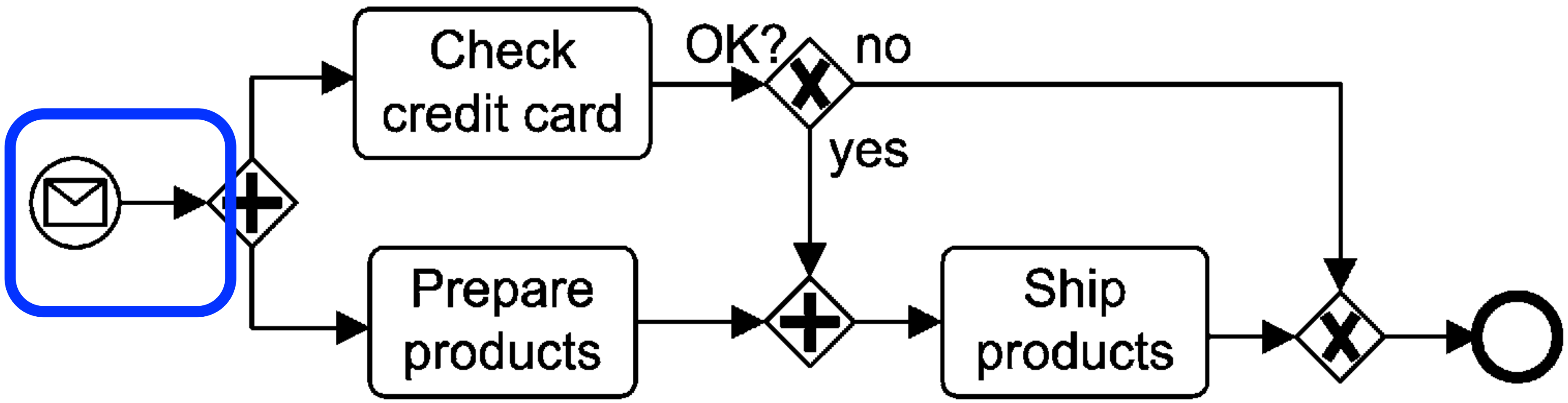


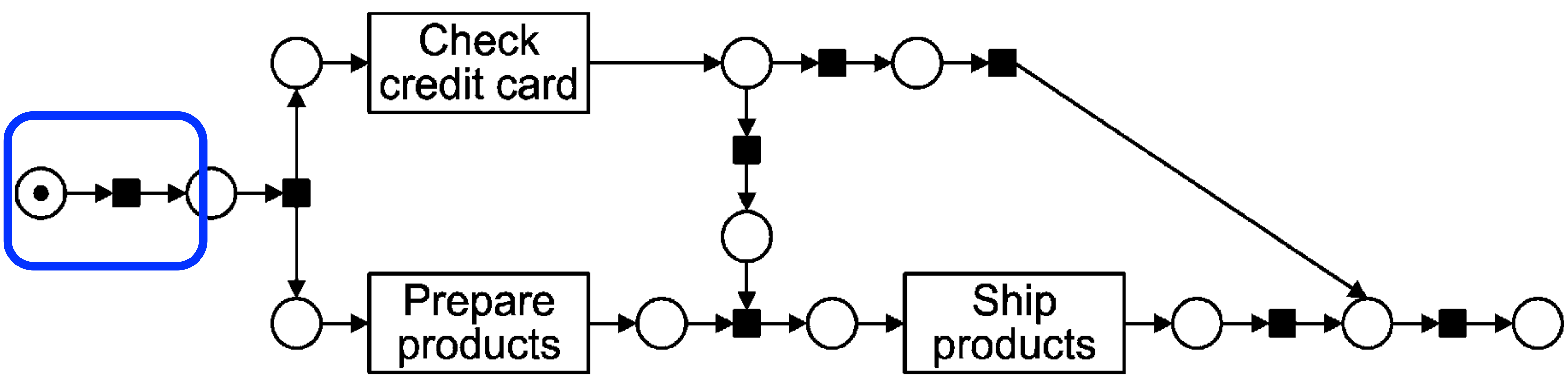
Example: Order process



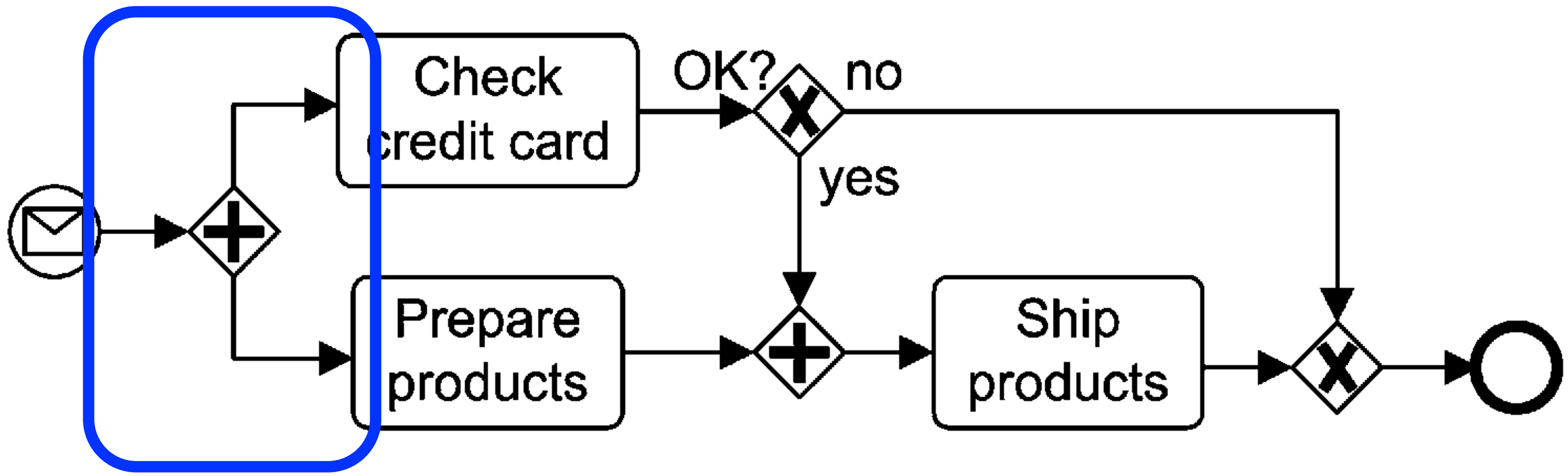


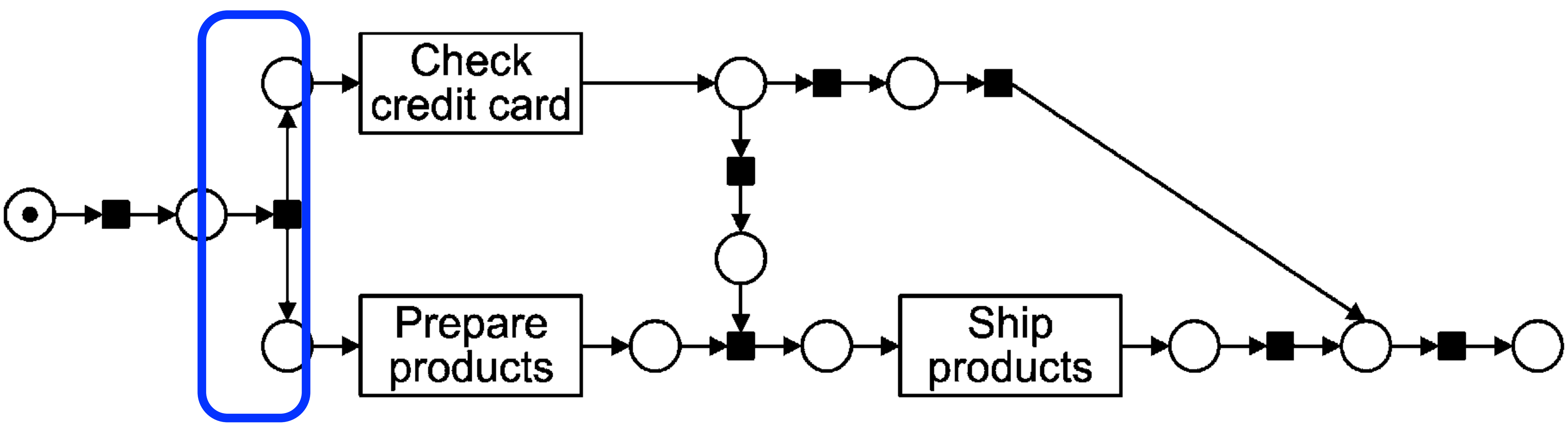
Example: Order process



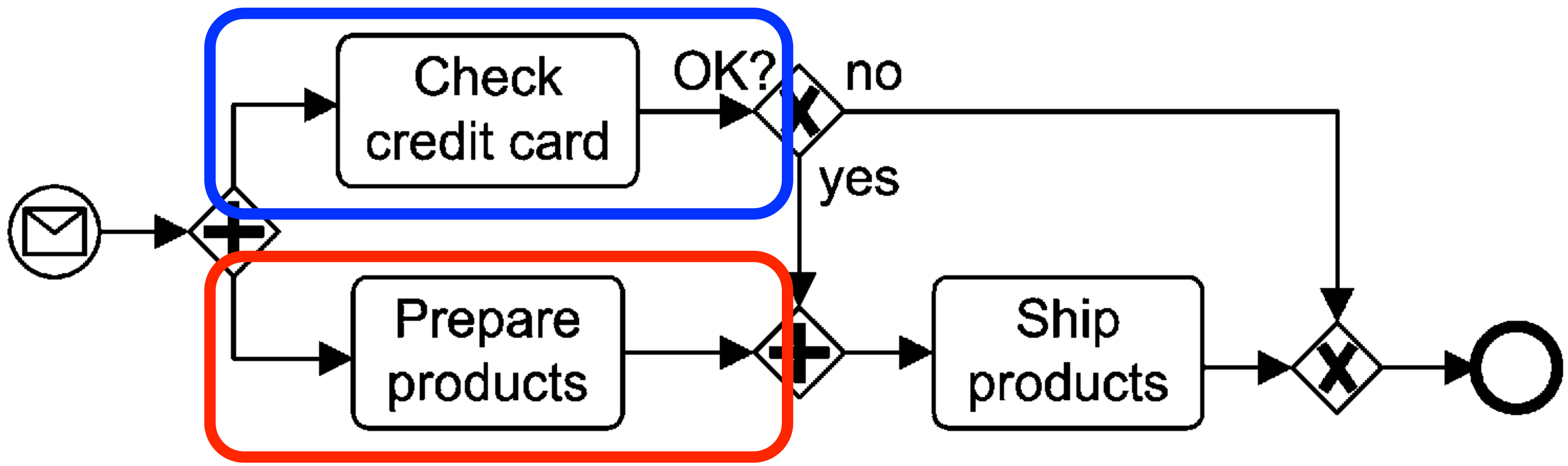


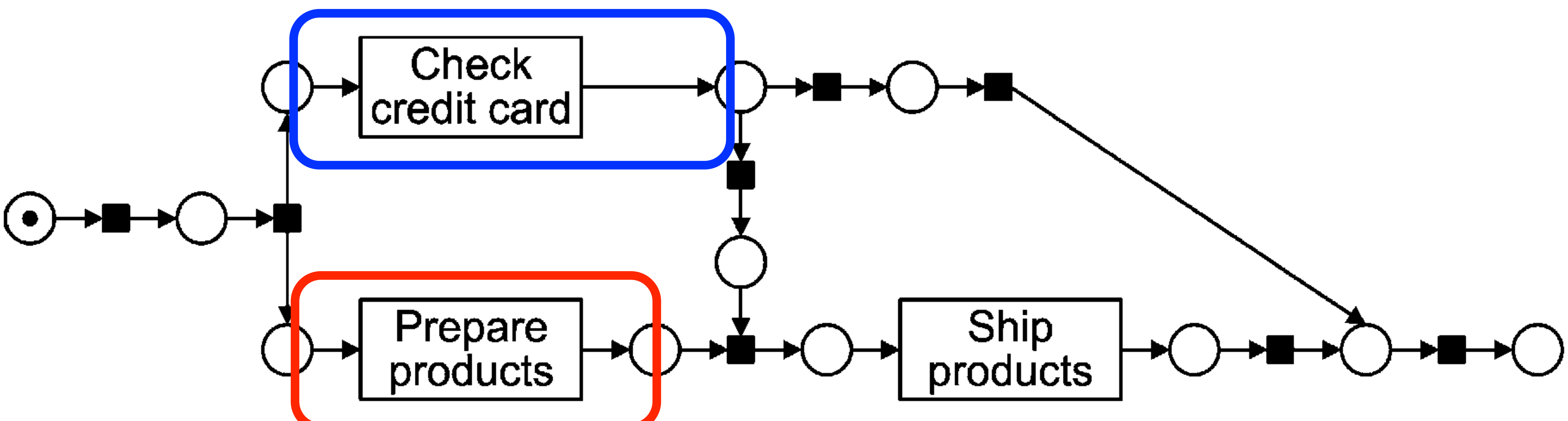
Example: Order process



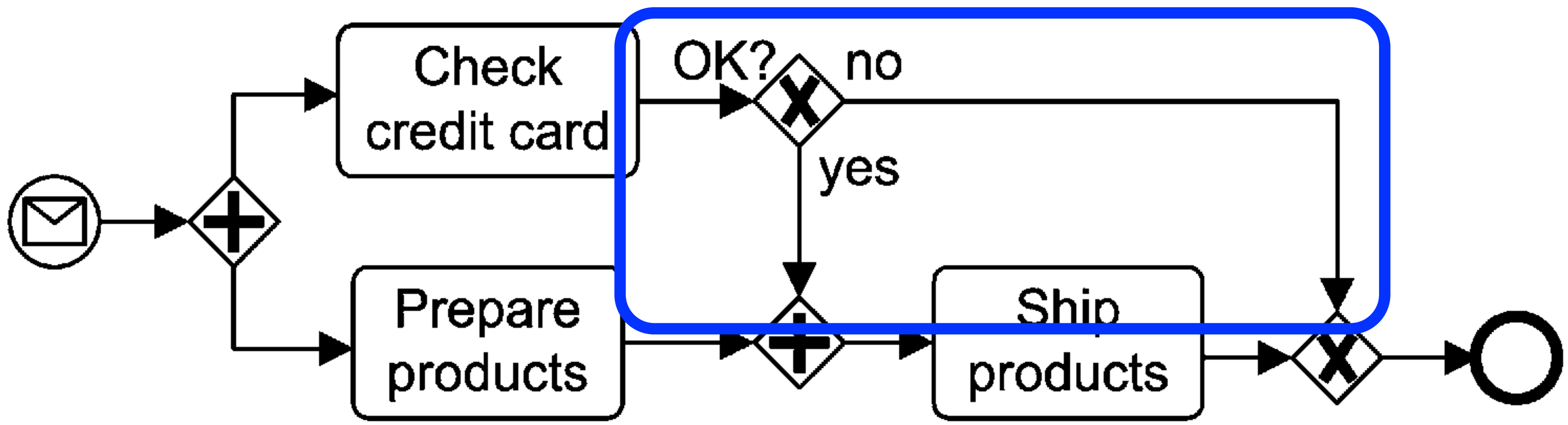


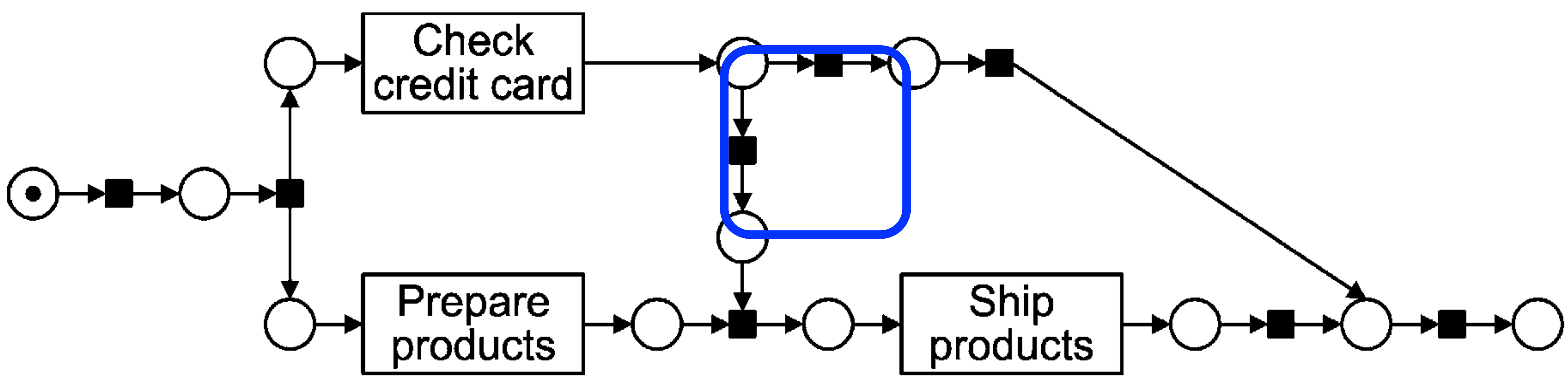
Example: Order process



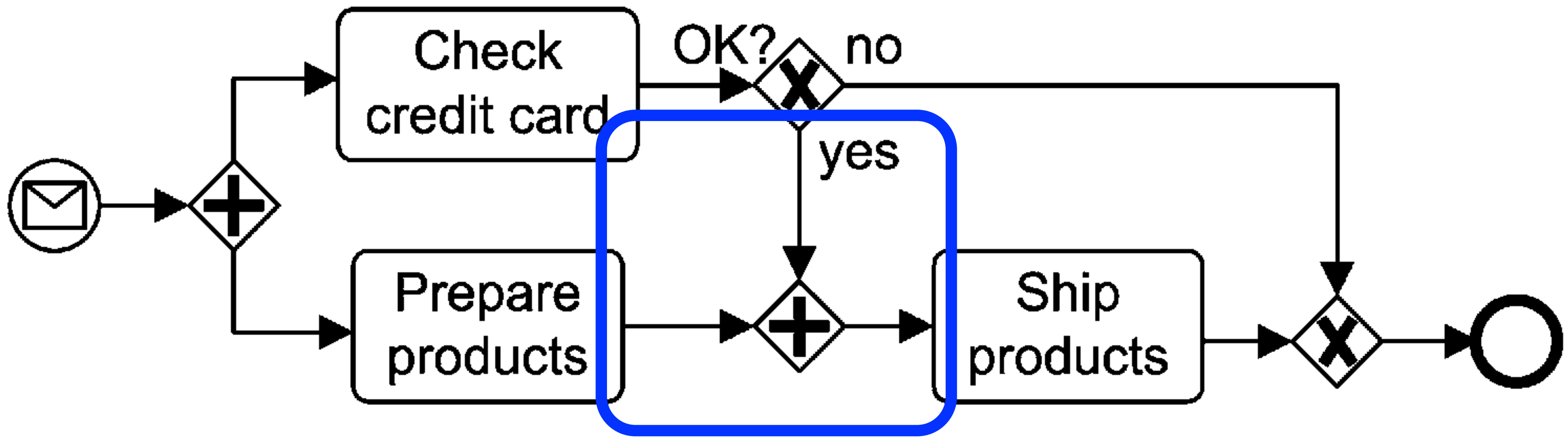


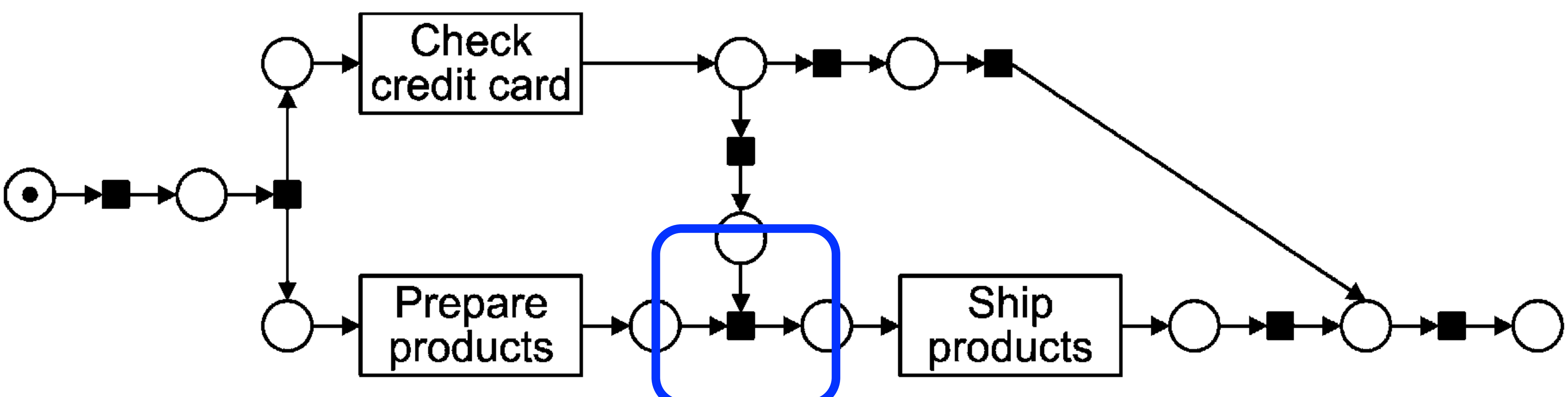
Example: Order process



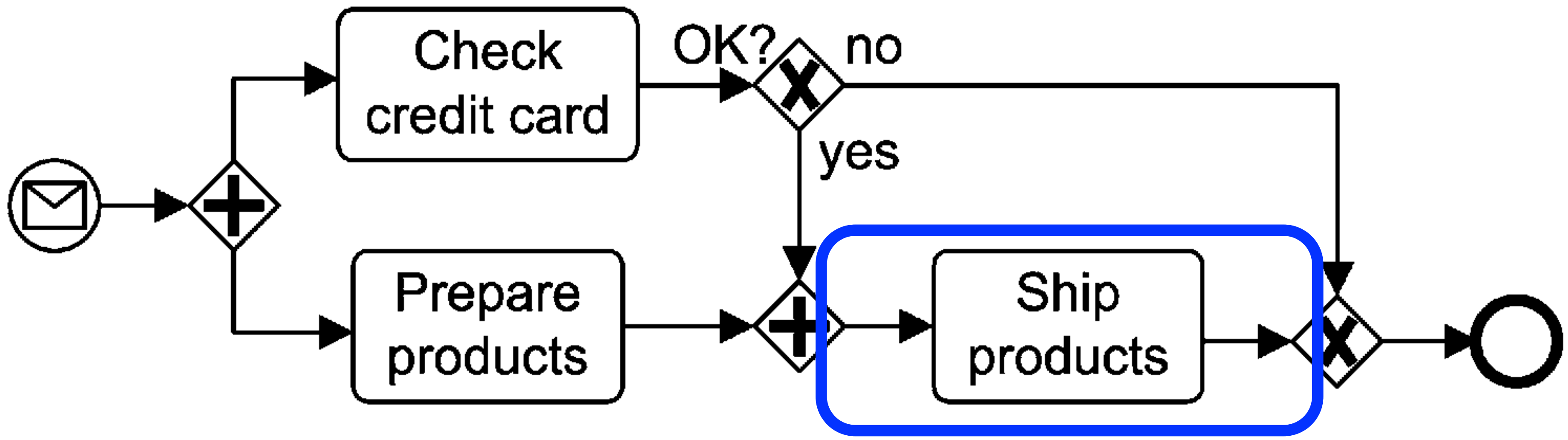


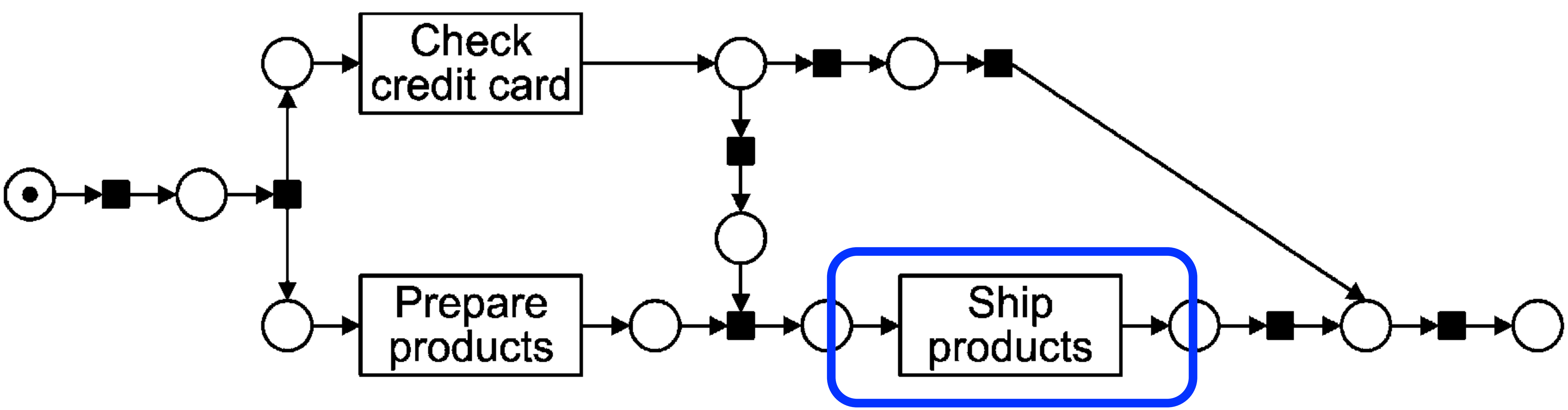
Example: Order process



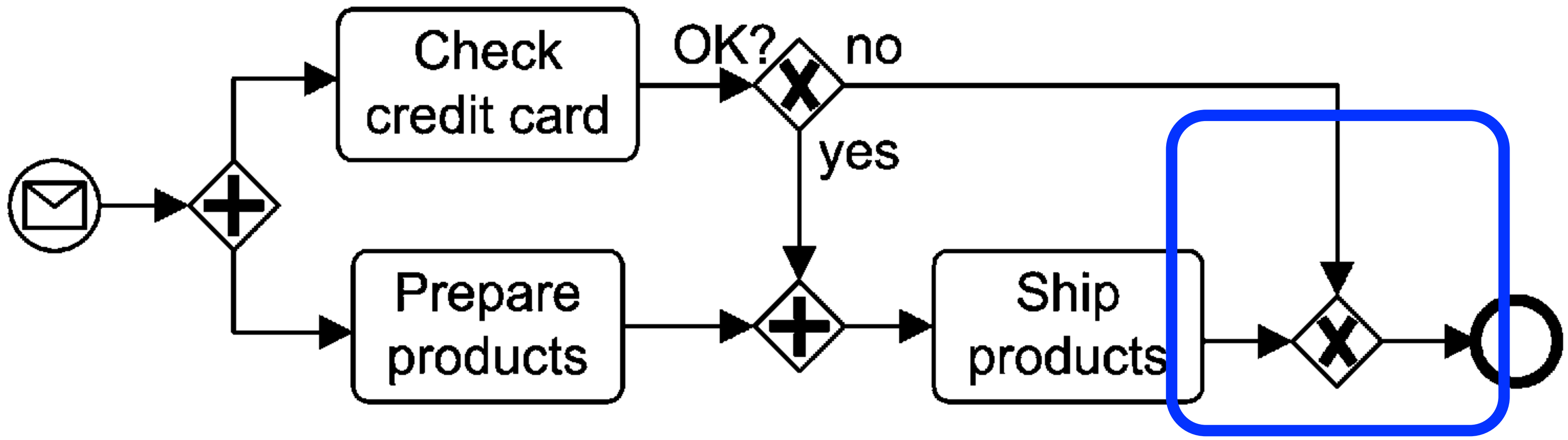


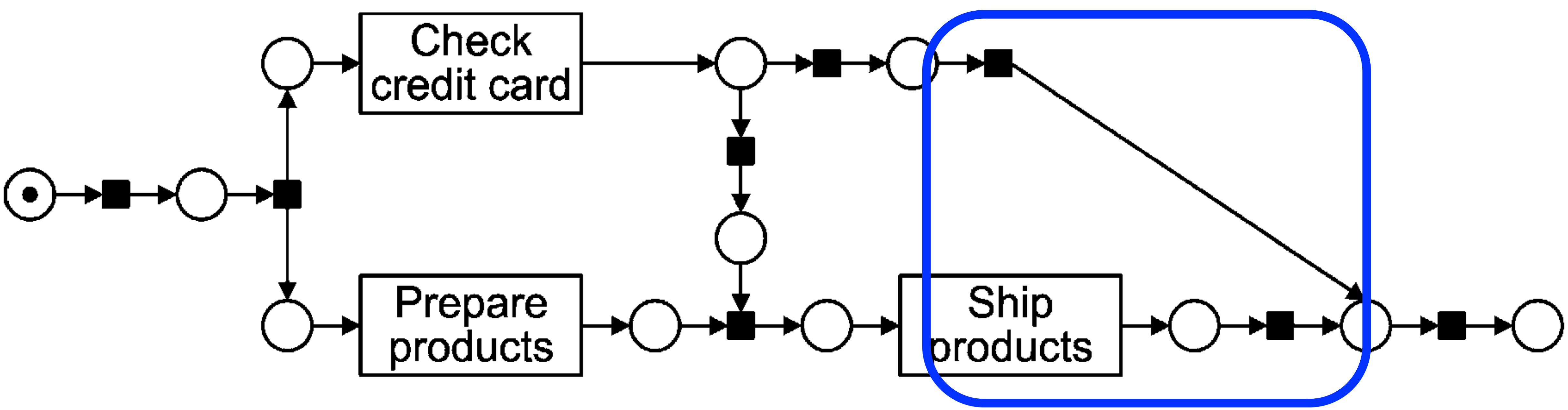
Example: Order process



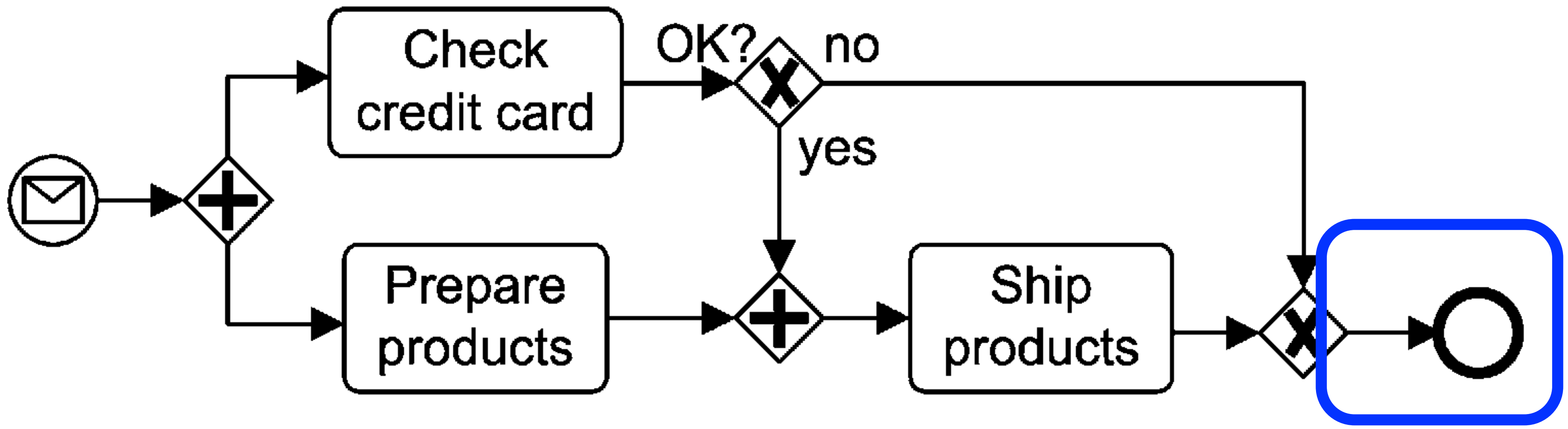


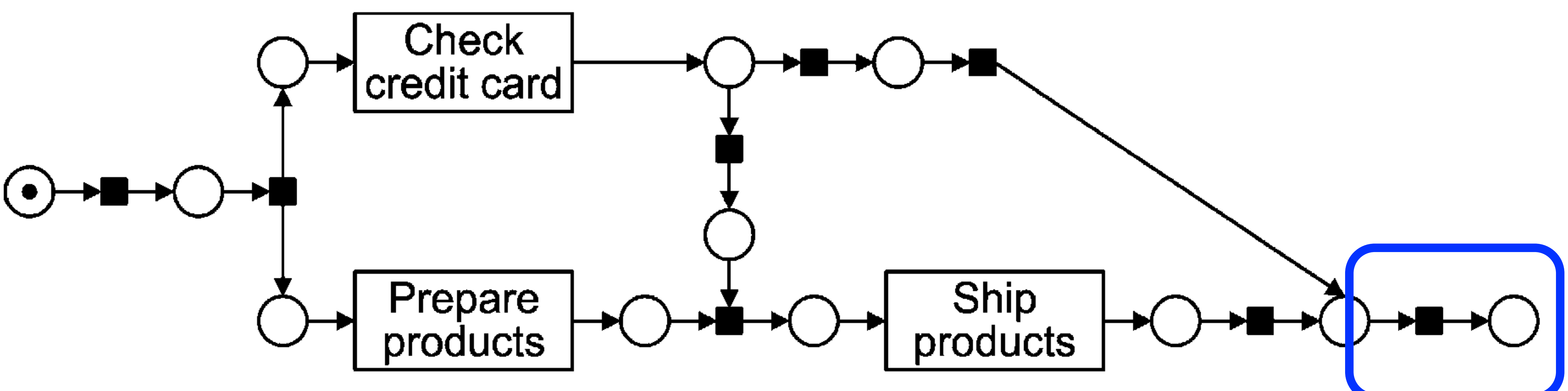
Example: Order process



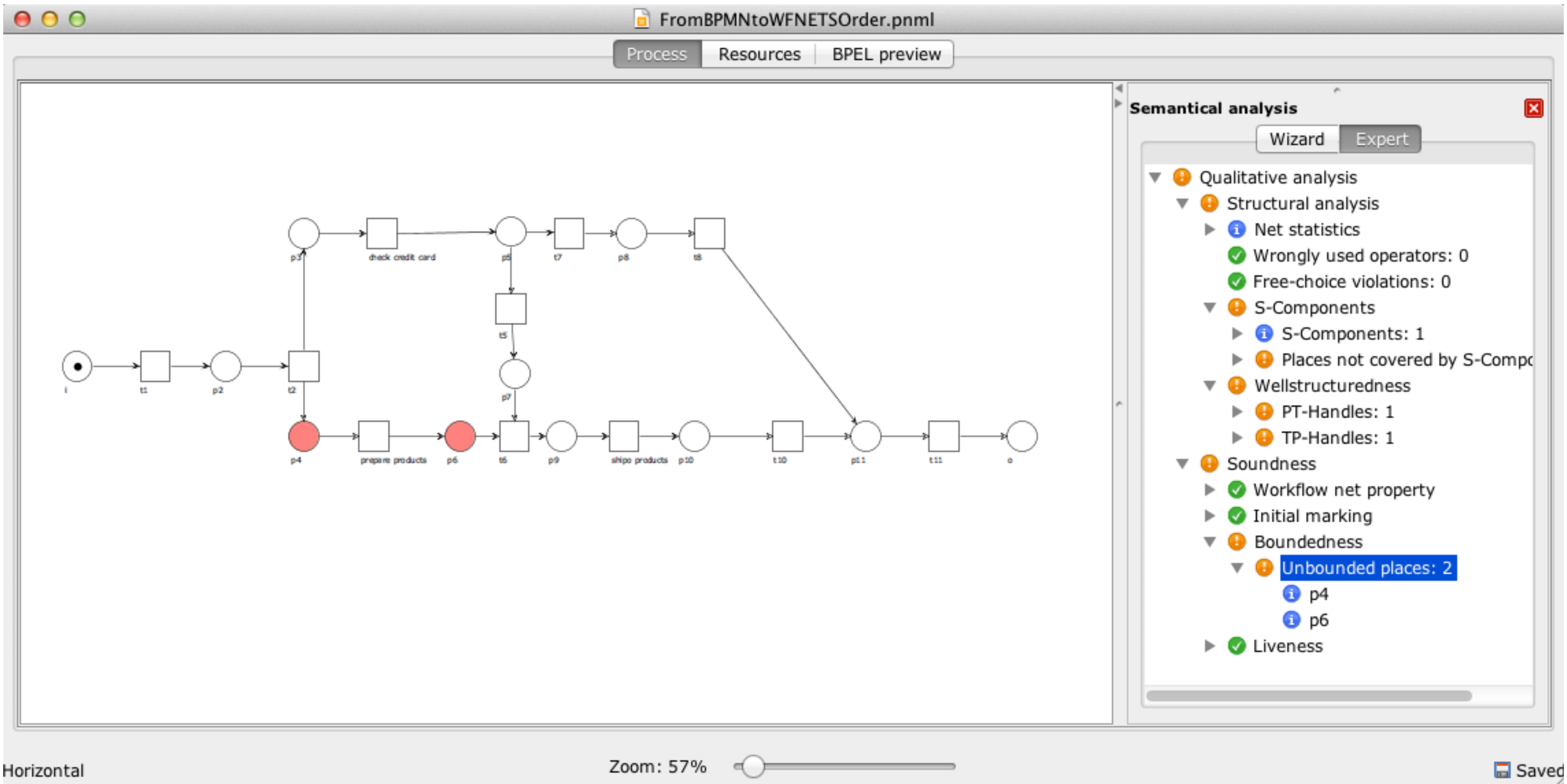


Example: Order process

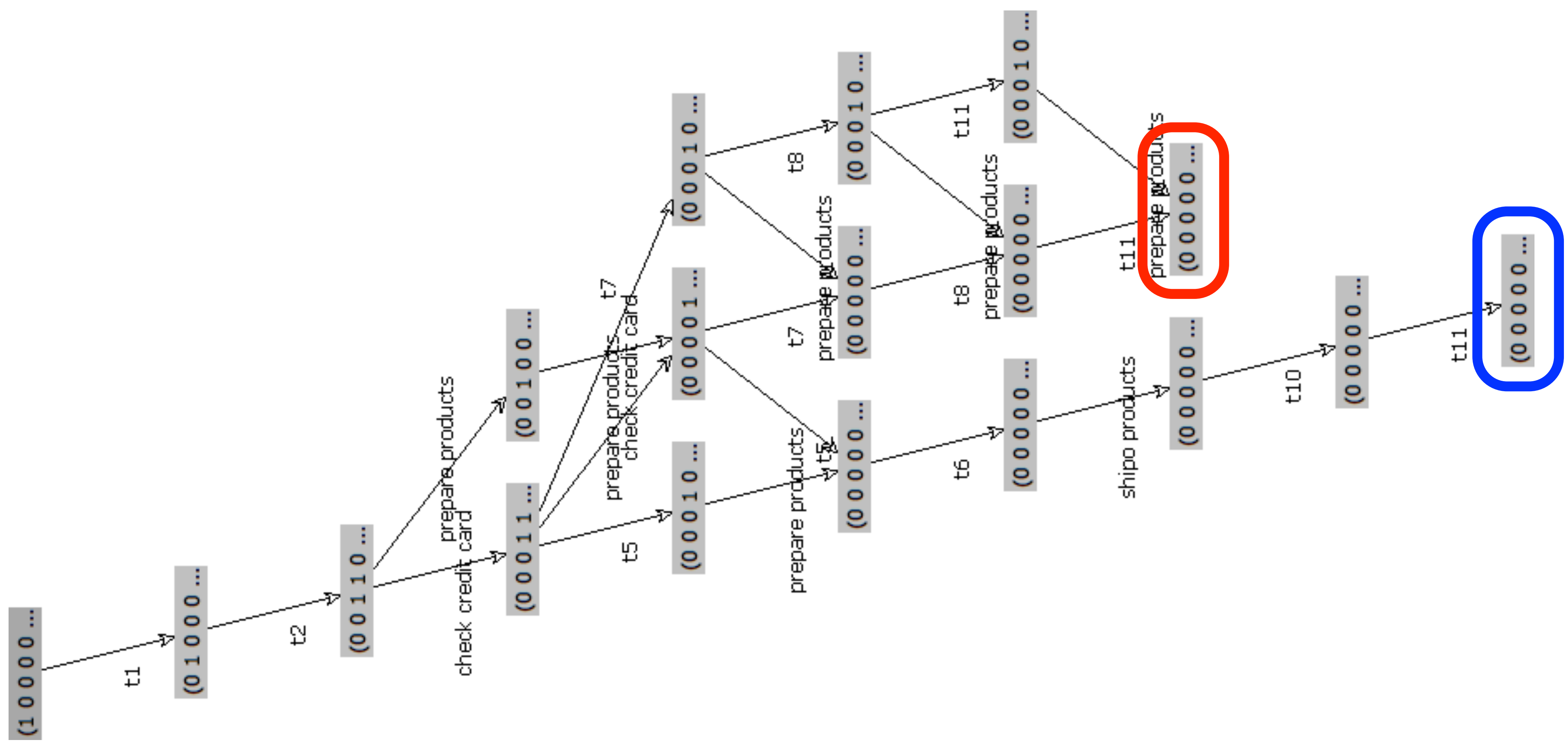




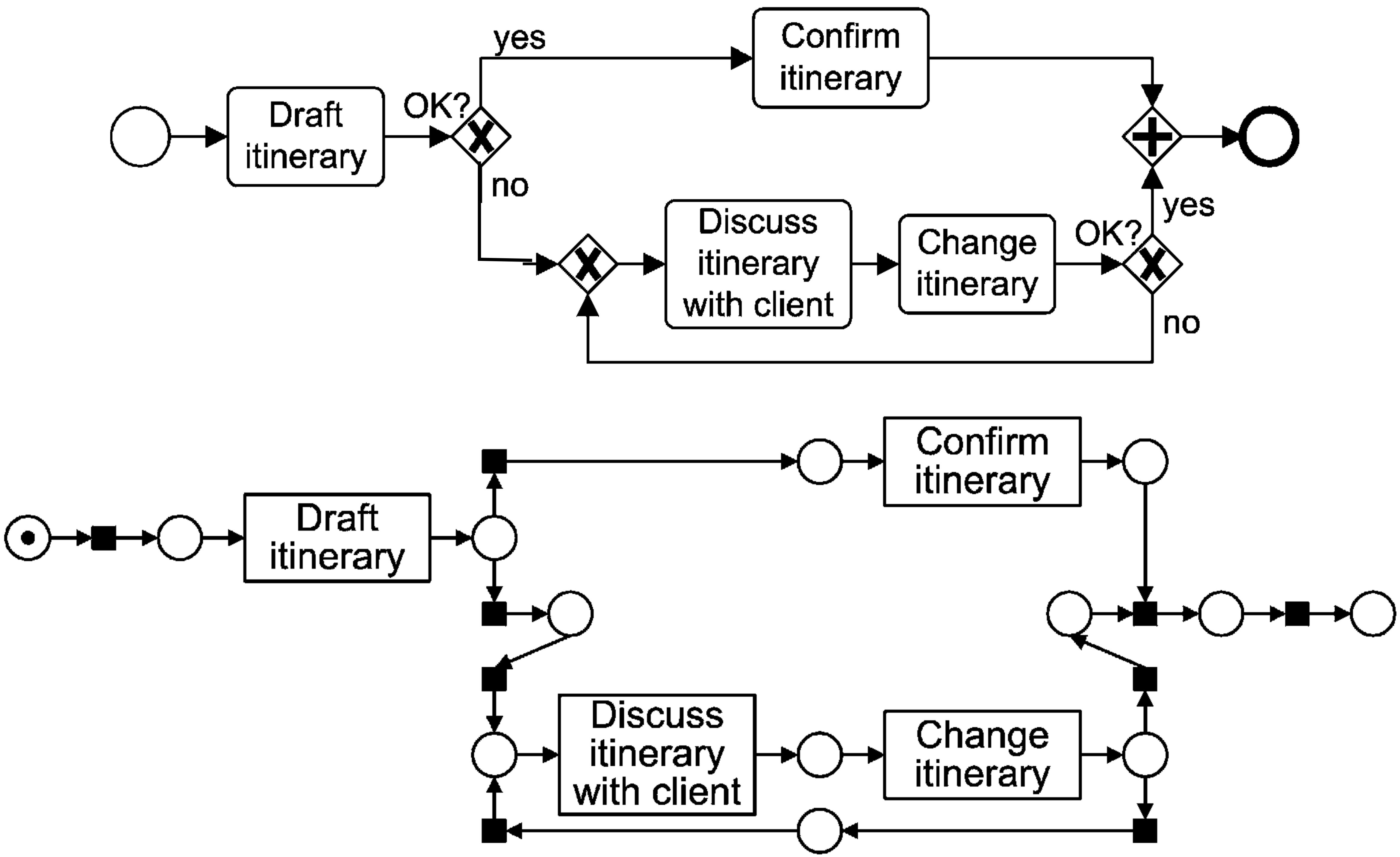
Example: Order process



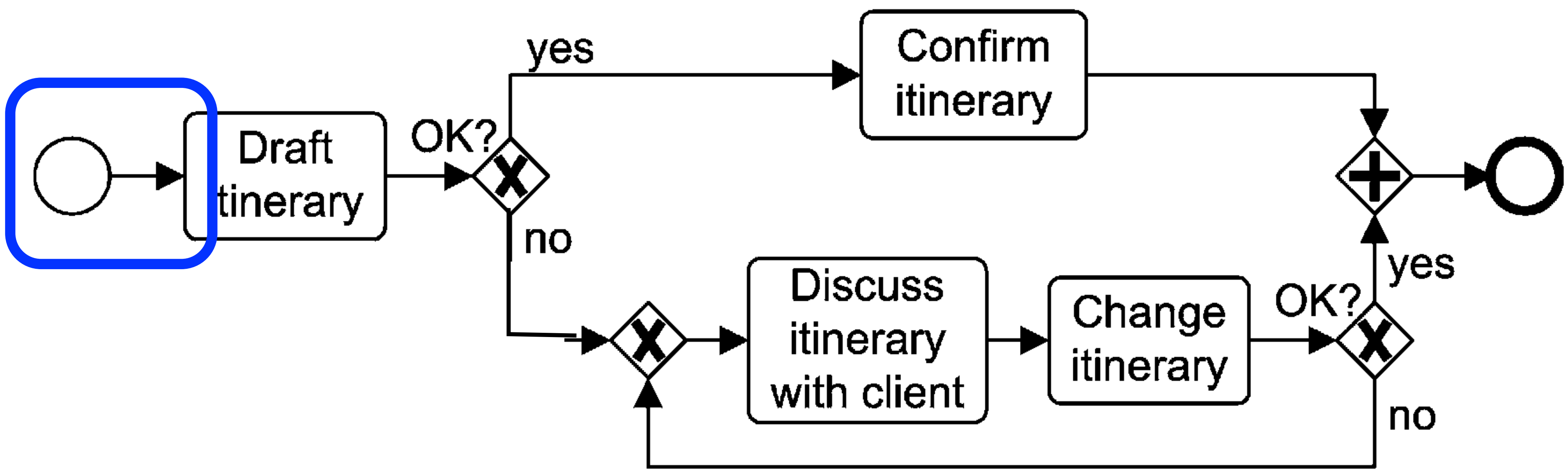
Example: Order process

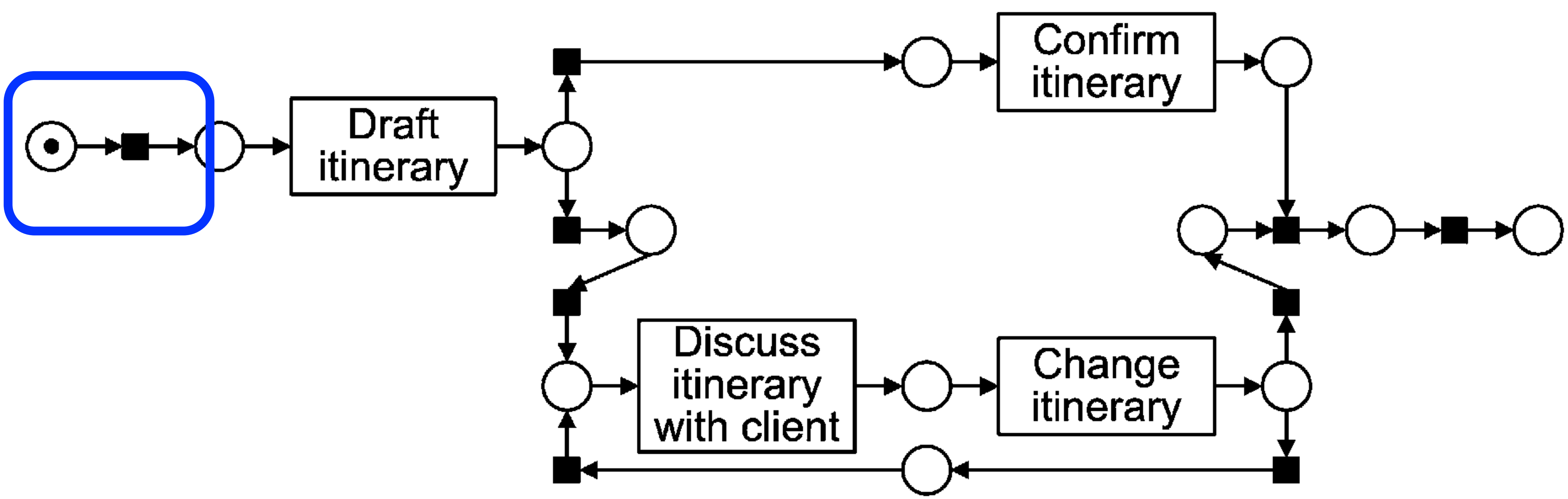


Example: Travel itinerary

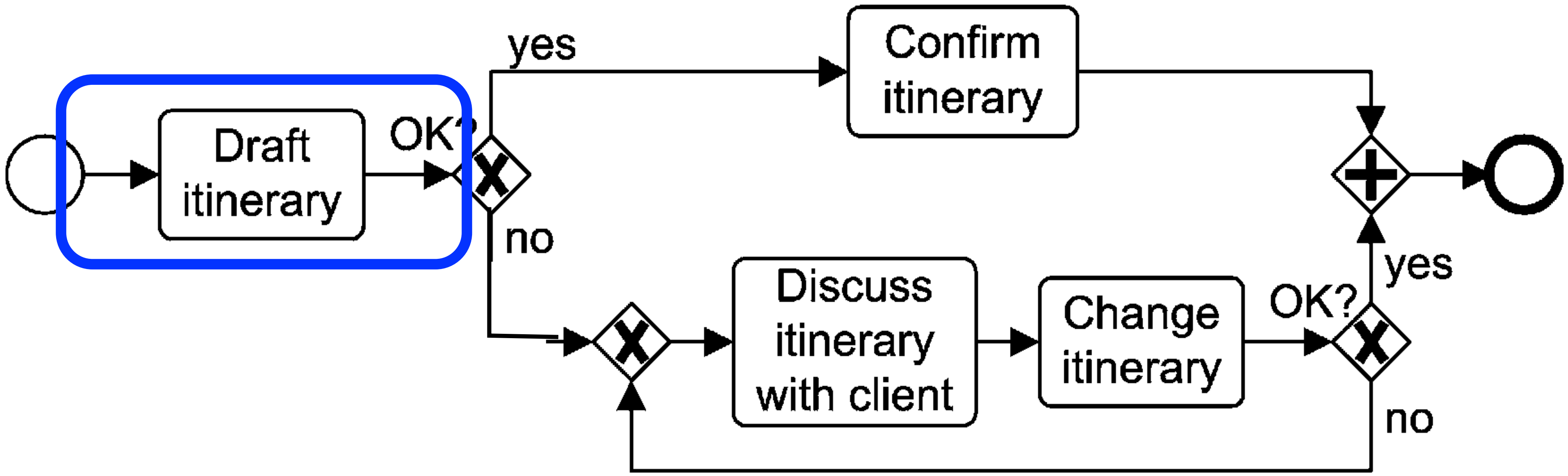


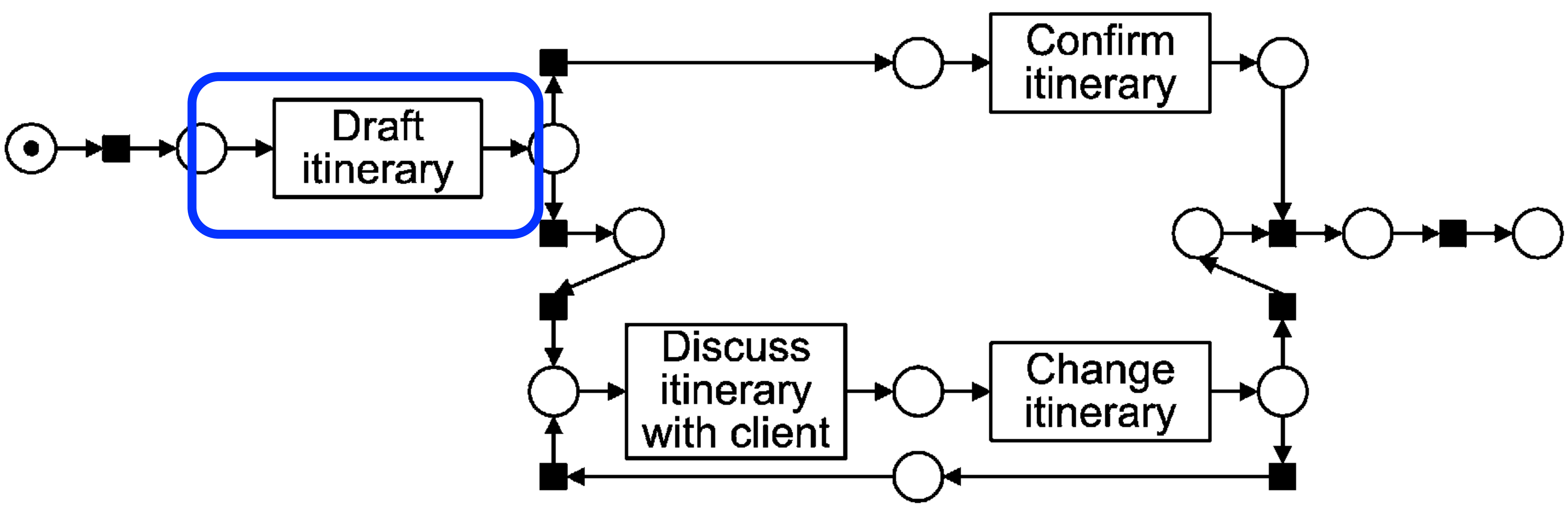
Example: Travel itinerary



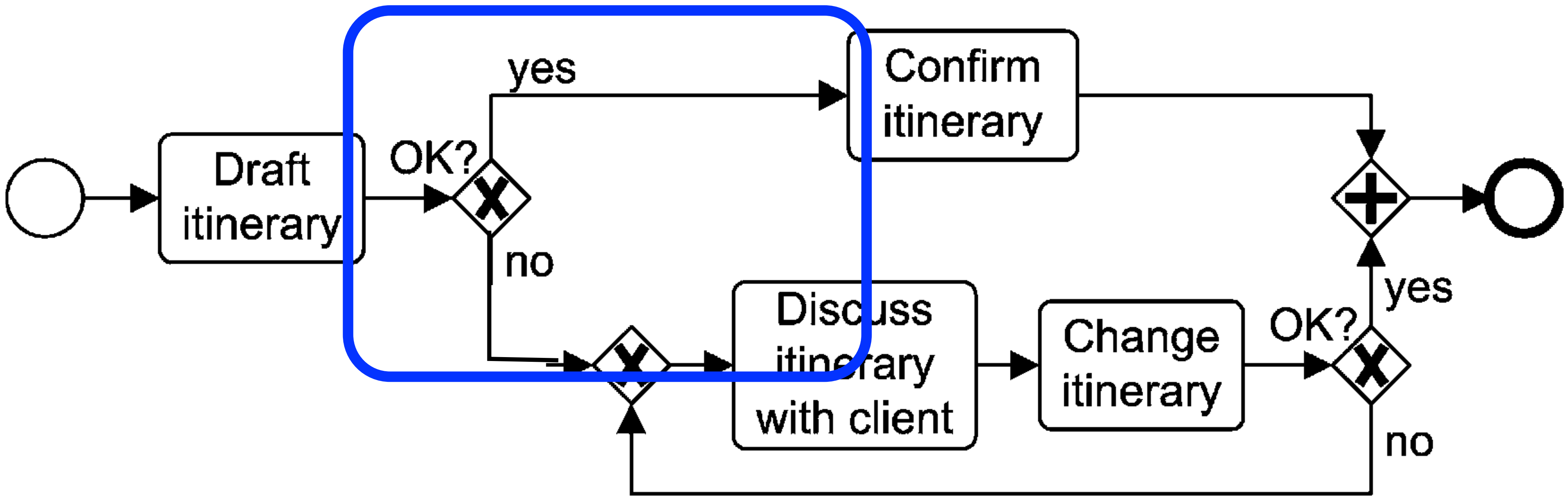


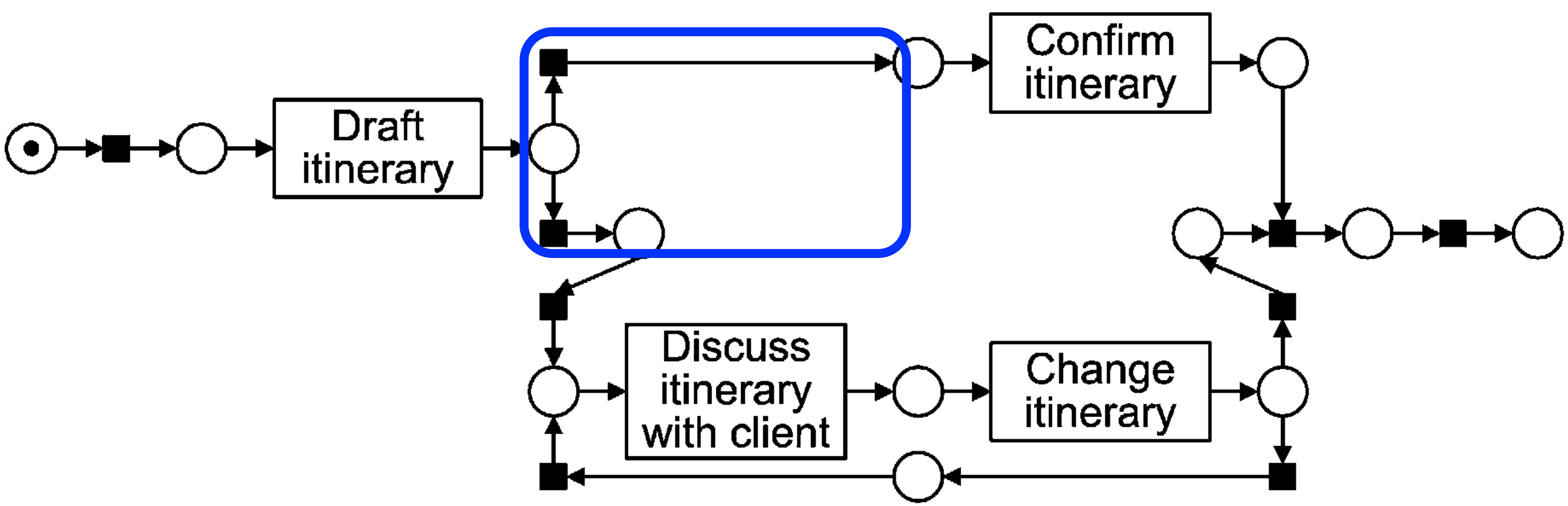
Example: Travel itinerary



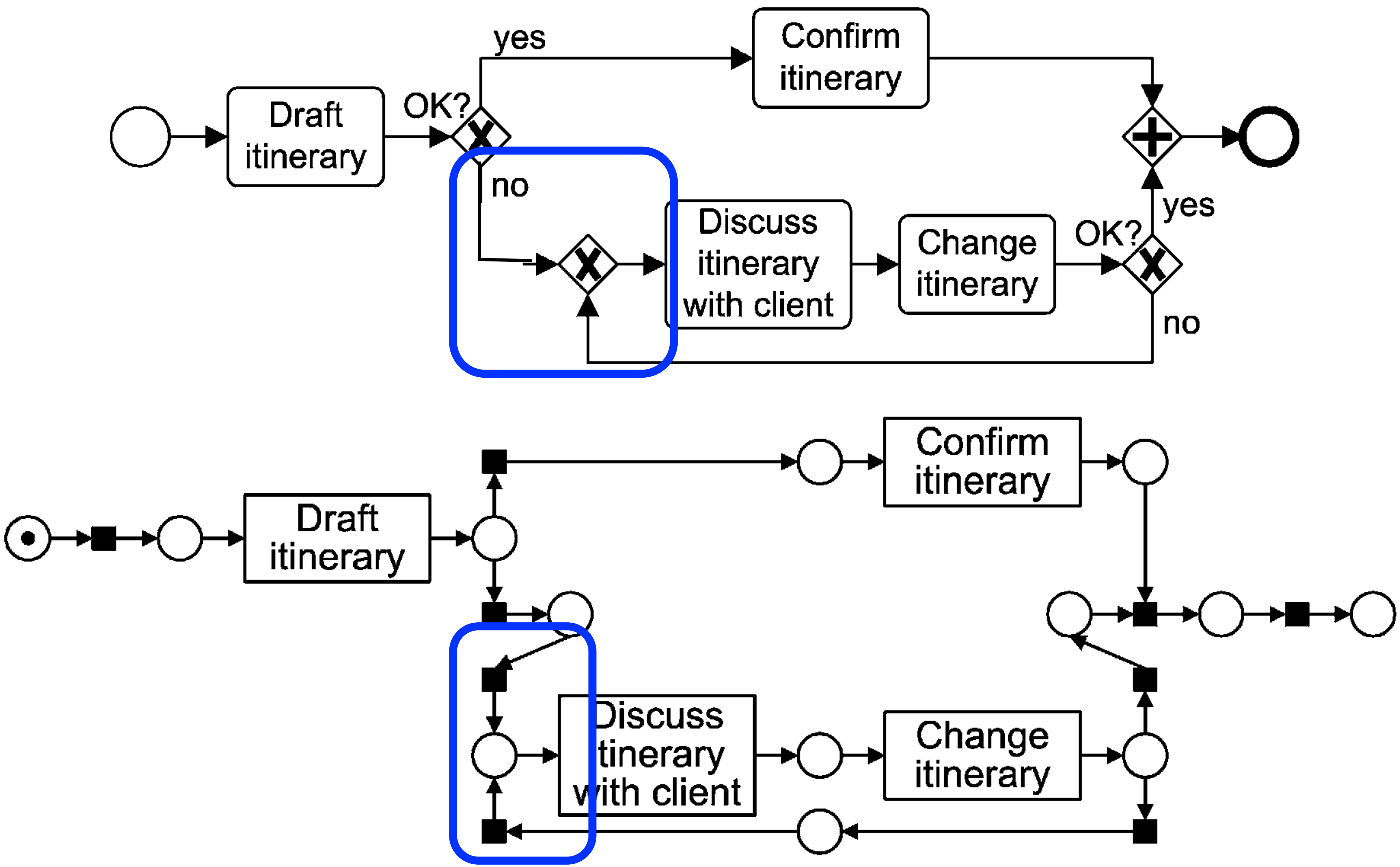


Example: Travel itinerary

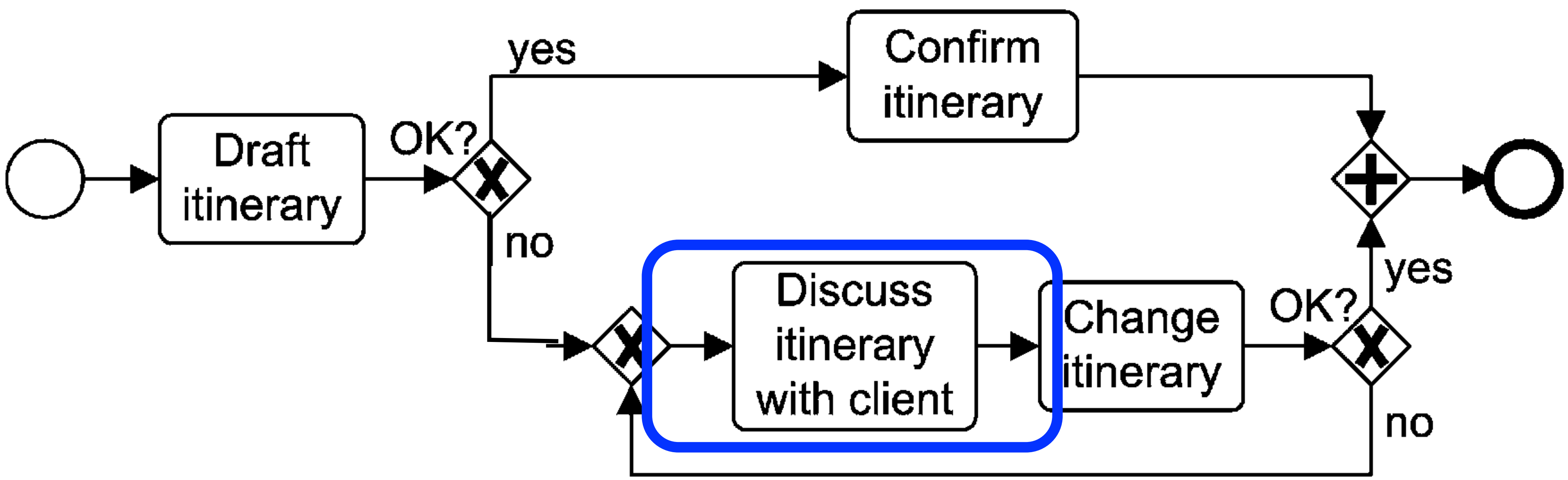


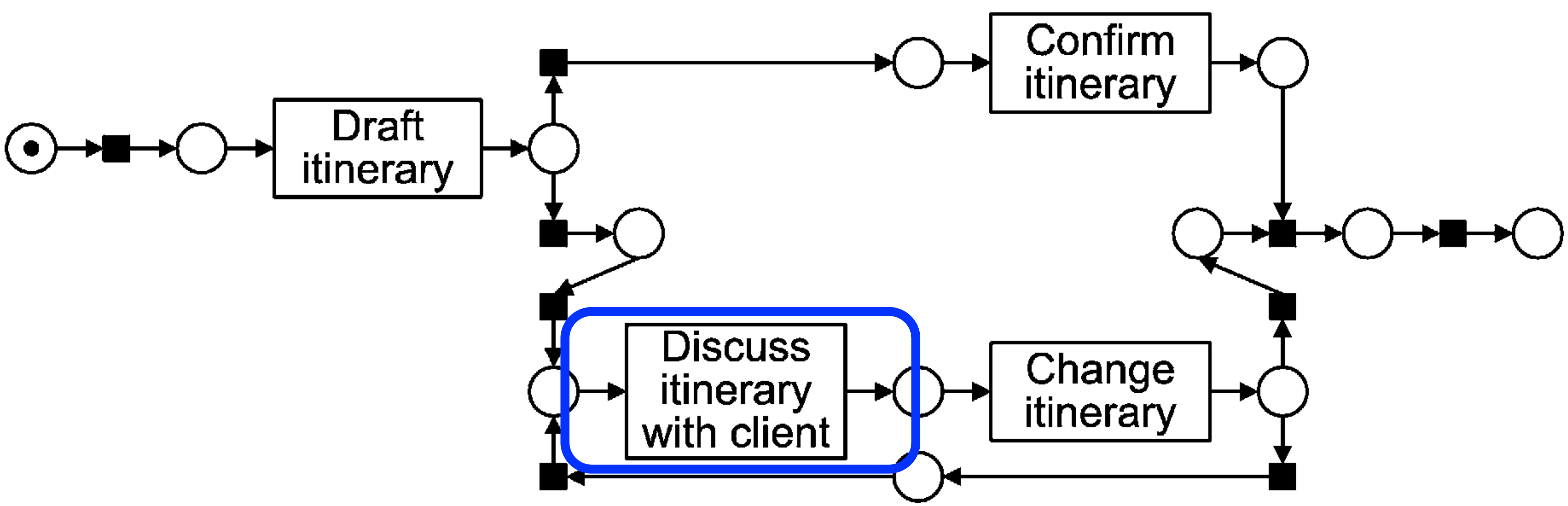


Example: Travel itinerary

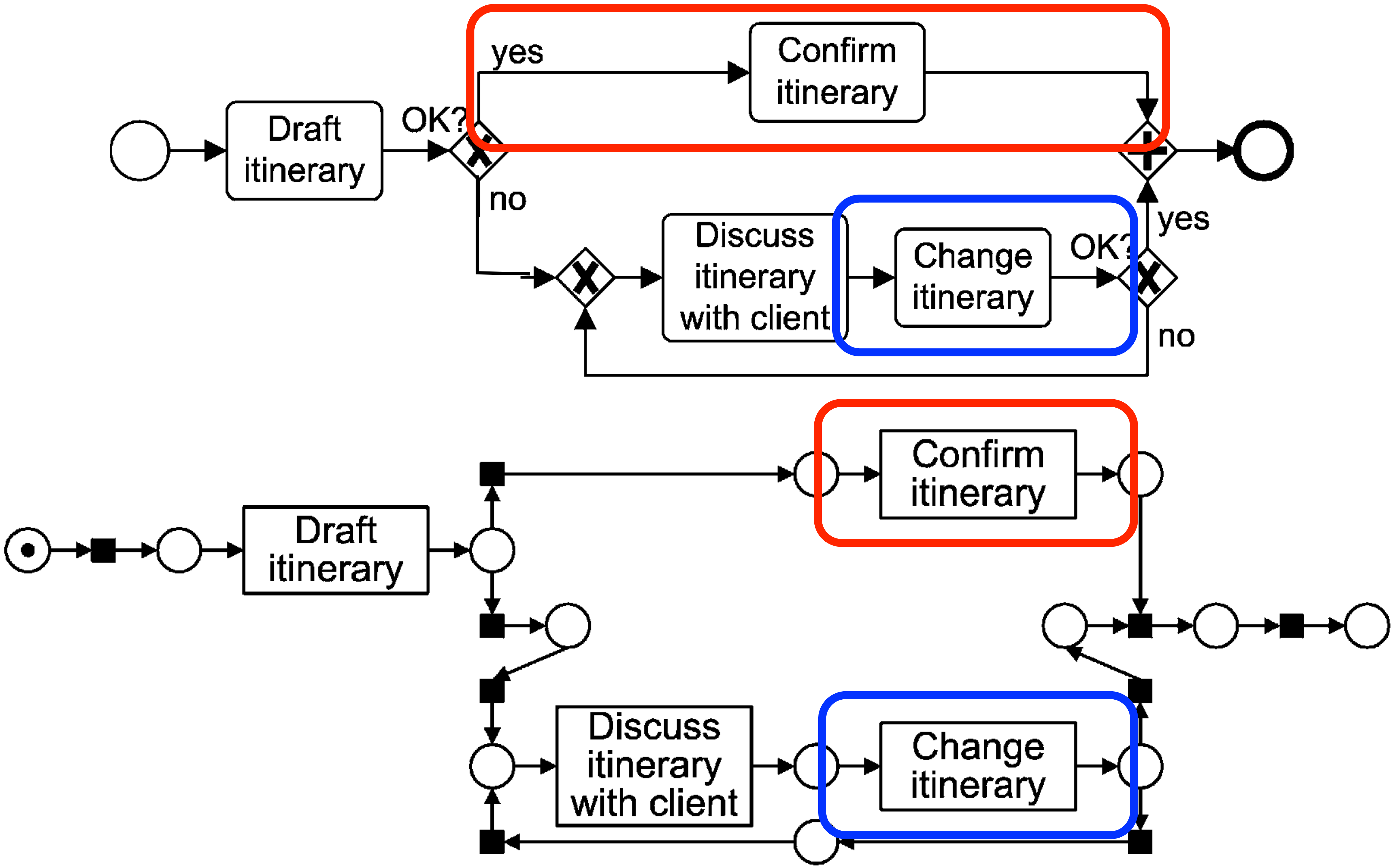


Example: Travel itinerary

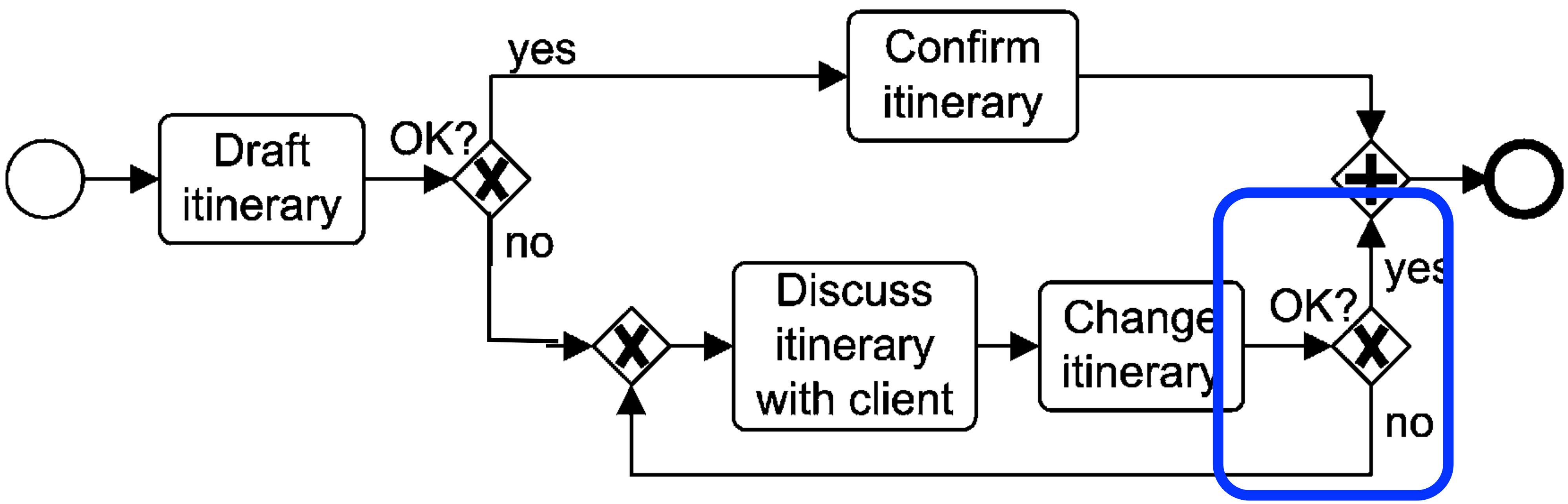


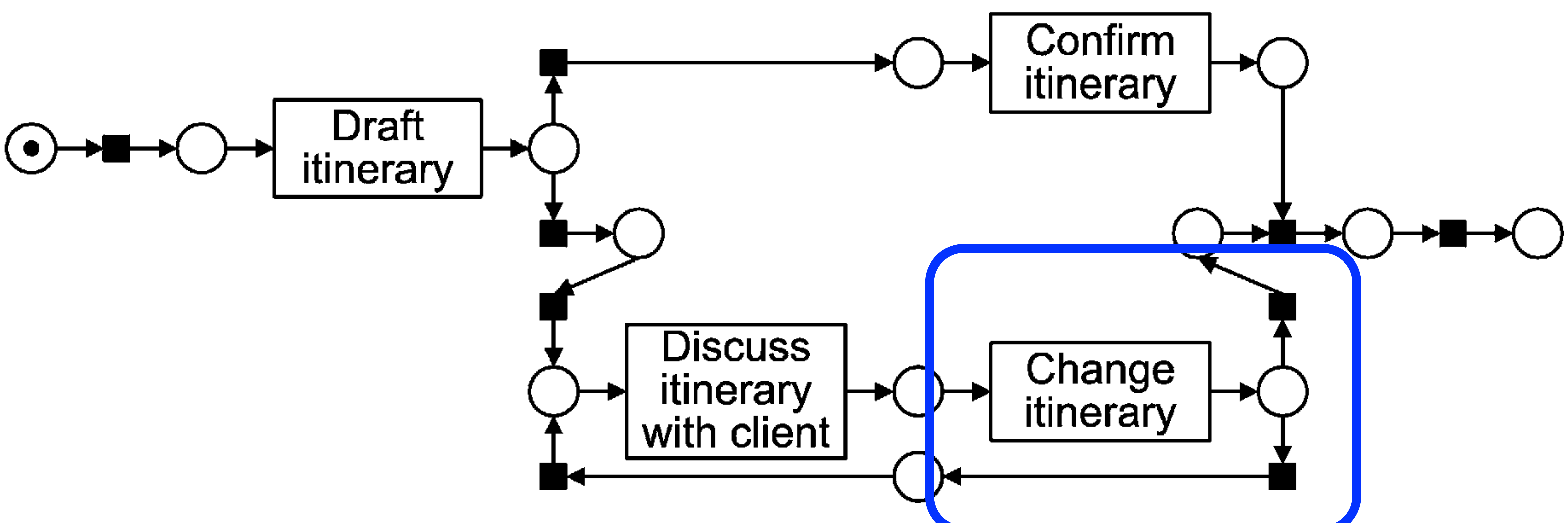


Example: Travel itinerary

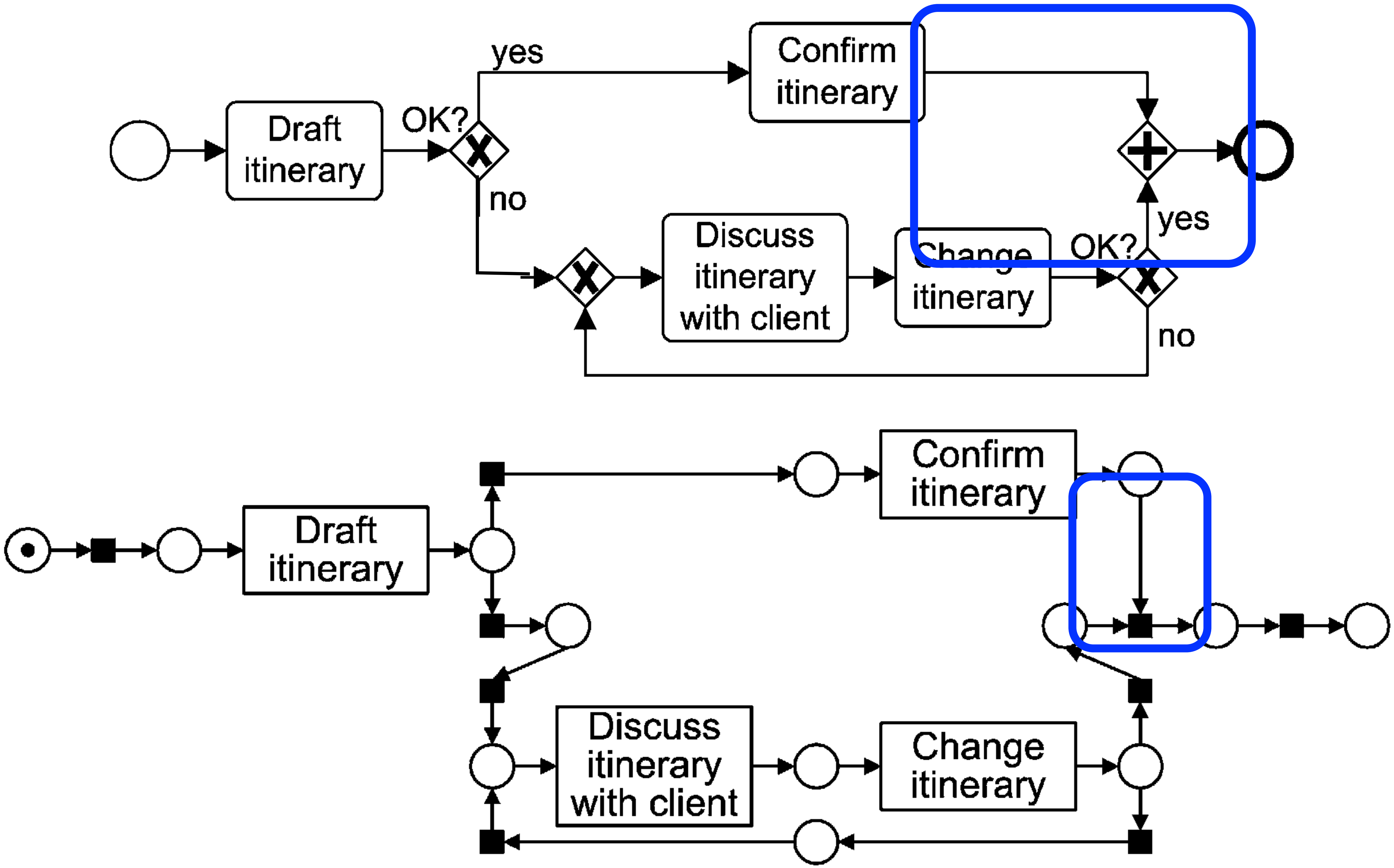


Example: Travel itinerary

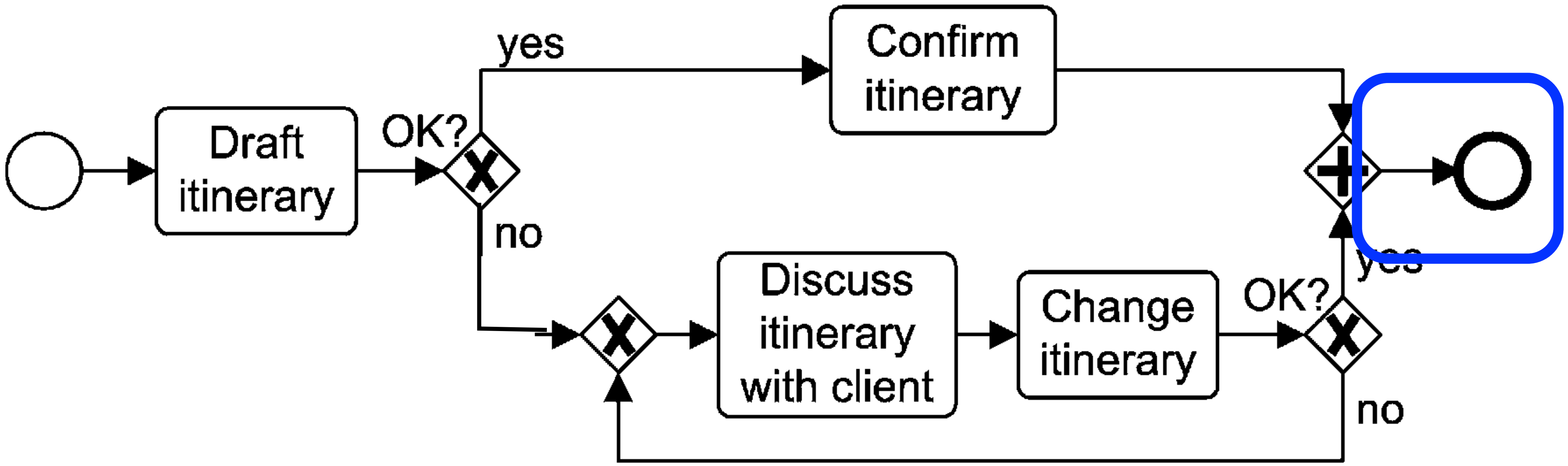


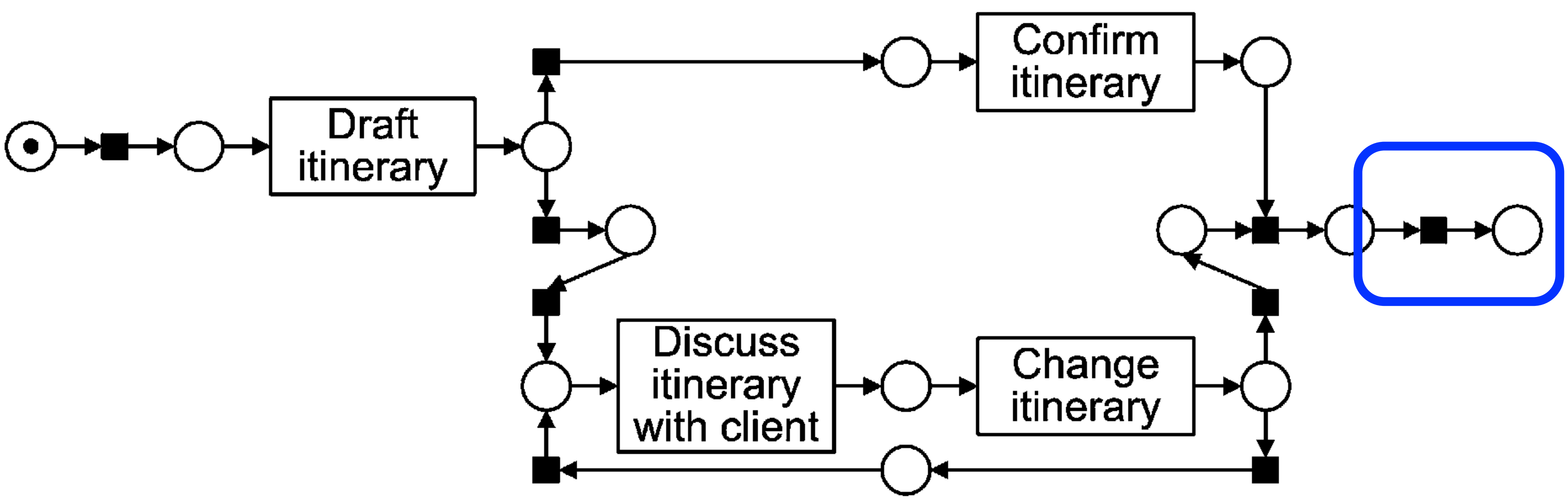


Example: Travel itinerary

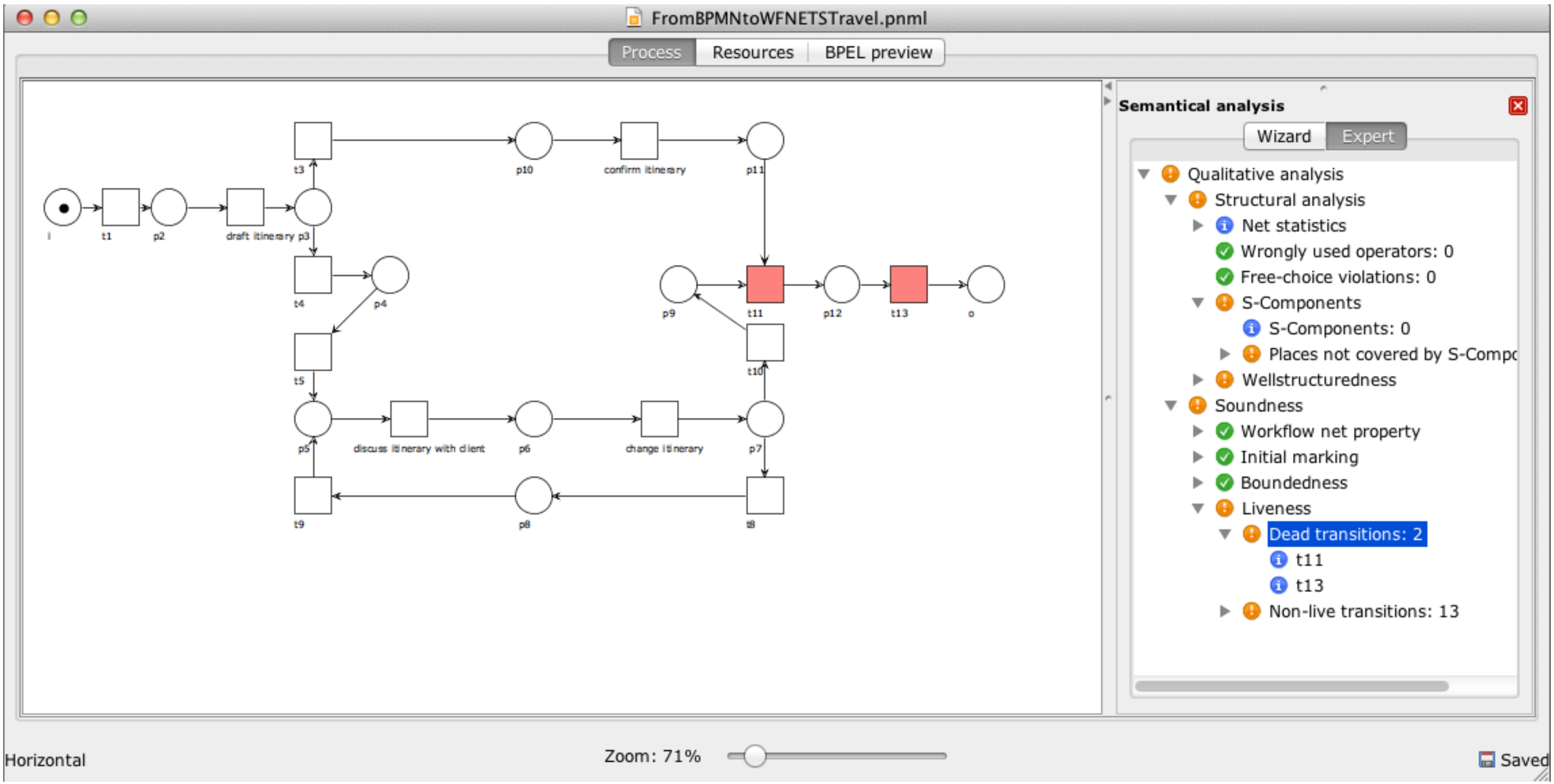


Example: Travel itinerary

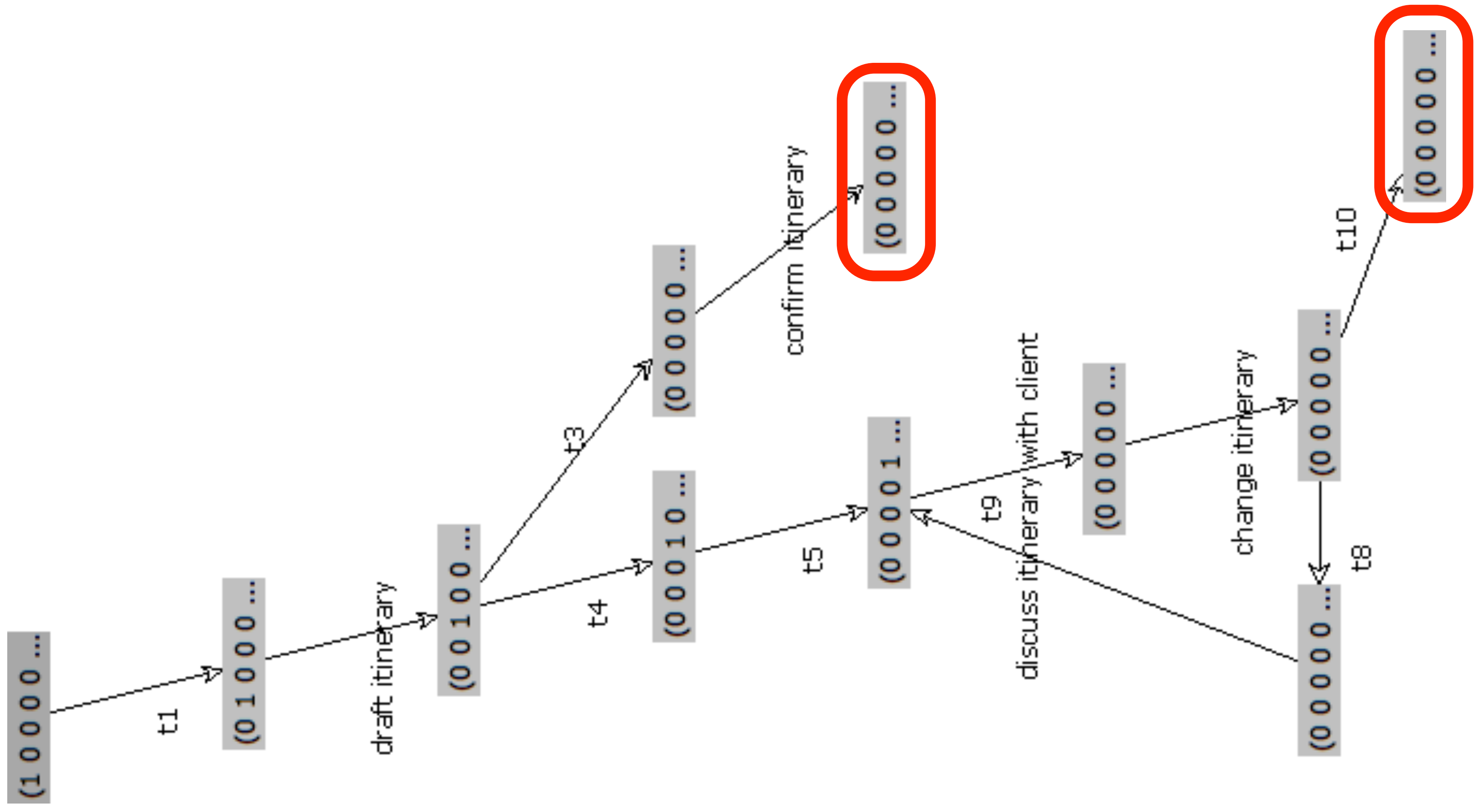




Example: Travel itinerary



Example: Travel itinerary



Exercise

Translate the BPMN collaboration diagram to nets and discuss problematic issues