

BPMN – Fortgeschrittene Konstrukte & Verifikation mit Petri Netzen

Agenda

1. Aktivitäten (fortgeschritten)

- Buch: 4.1 More on Rework and Repetition

2. Events (fortgeschritten)

- Buch: 4.2 Handling Events & 4.3.1 Process Abortion

3. Qualitätssicherung

- Buch: 5.4 Process Model Quality Assurance

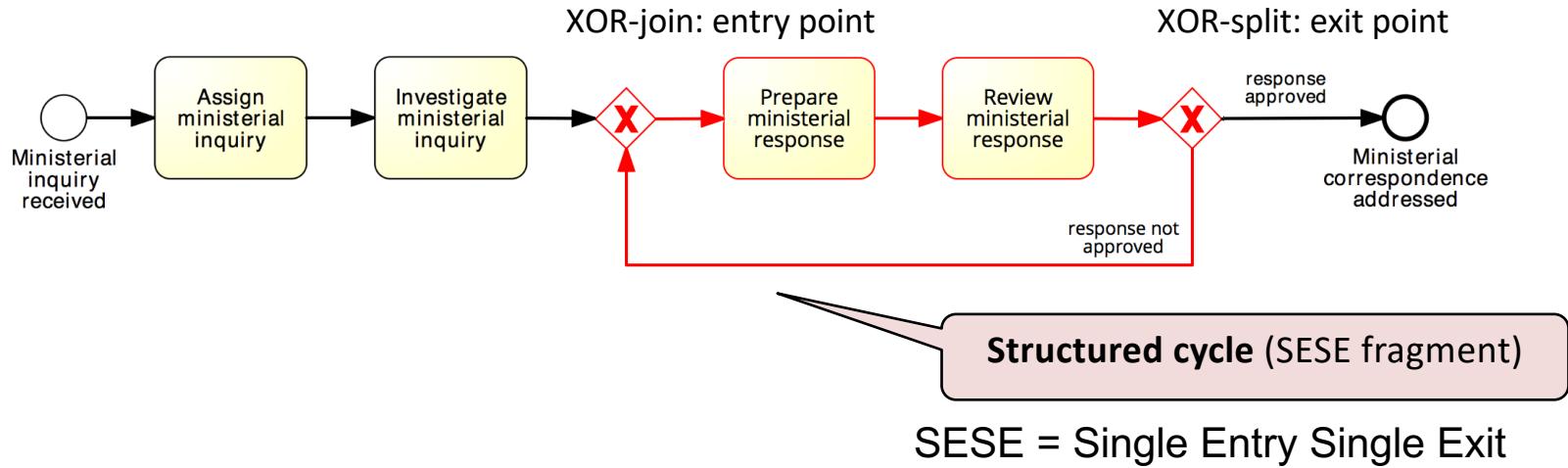
4. BPMN Verifikation mit Hilfe von Petri-Netzen

- Journal Paper: Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). *Semantics and analysis of business process models in BPMN*

Book: 4.1 More on Rework and Repetition

AKTIVITÄTEN (FORTGESCHRITTEN)

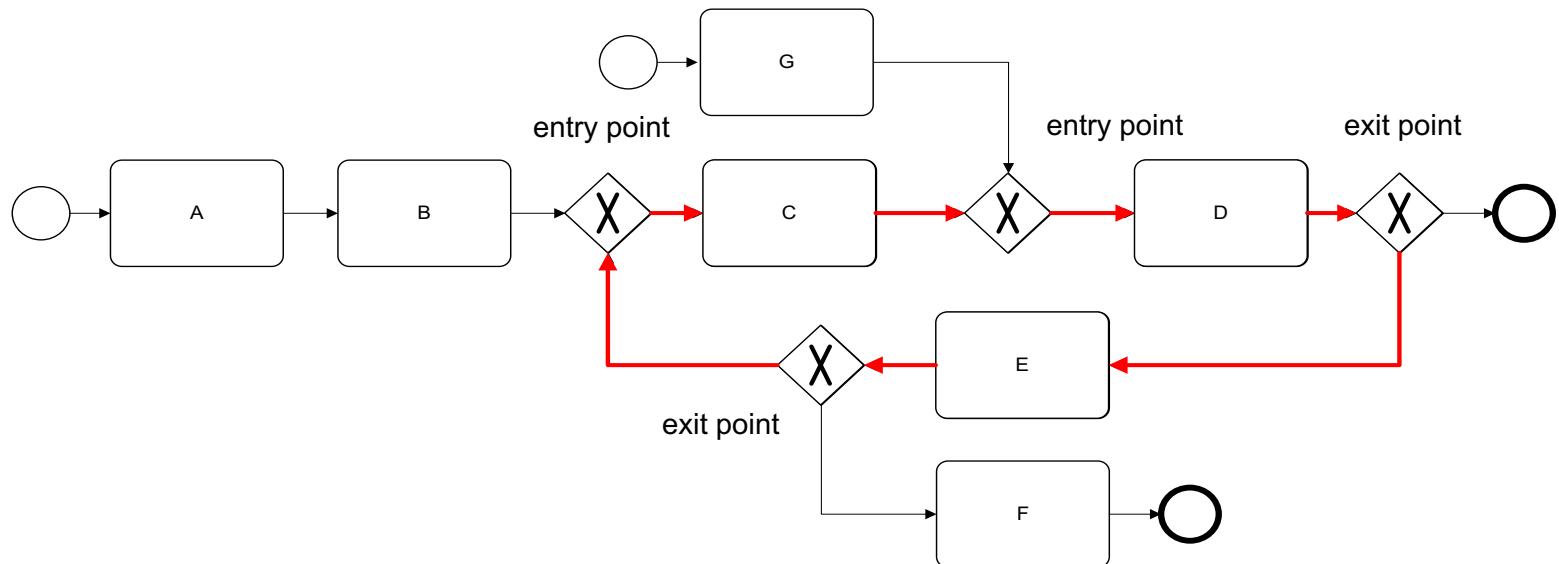
Zyklus (cycle)



Unstrukturierter Zyklus

Unstrukturierter Zyklus (arbitrary/unstructured cycle):

- mehrere Ein- und/oder Ausgangsknoten



Activity Loop Marker

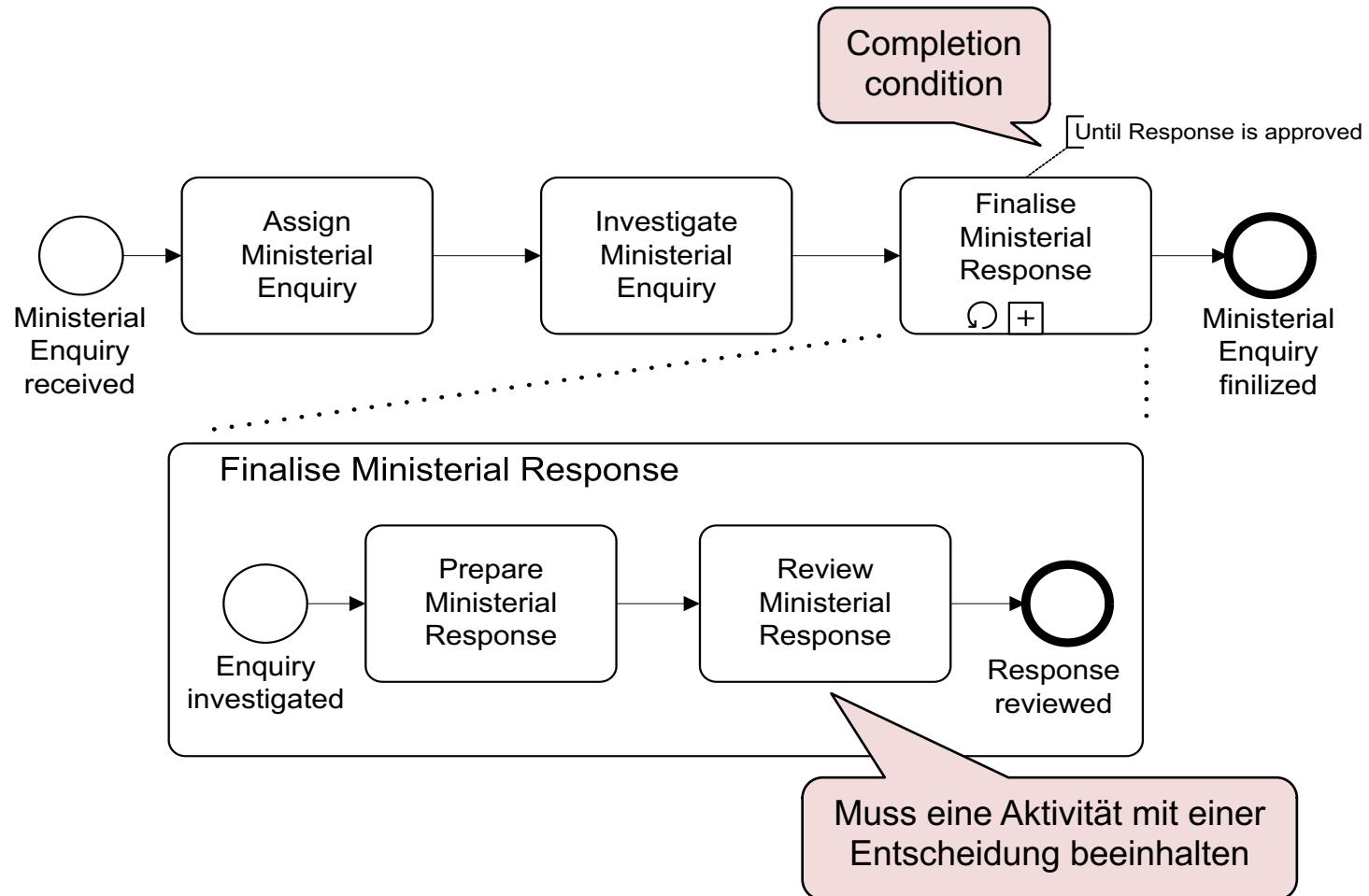
- Mit dem *Activity Loop* Marker können wiederholende Tasks oder Unterprozesse modelliert werden
- Eignet sich lediglich für strukturierte Zyklen

Task
Loop
⌚

Sub-process
Loop
⌚ +

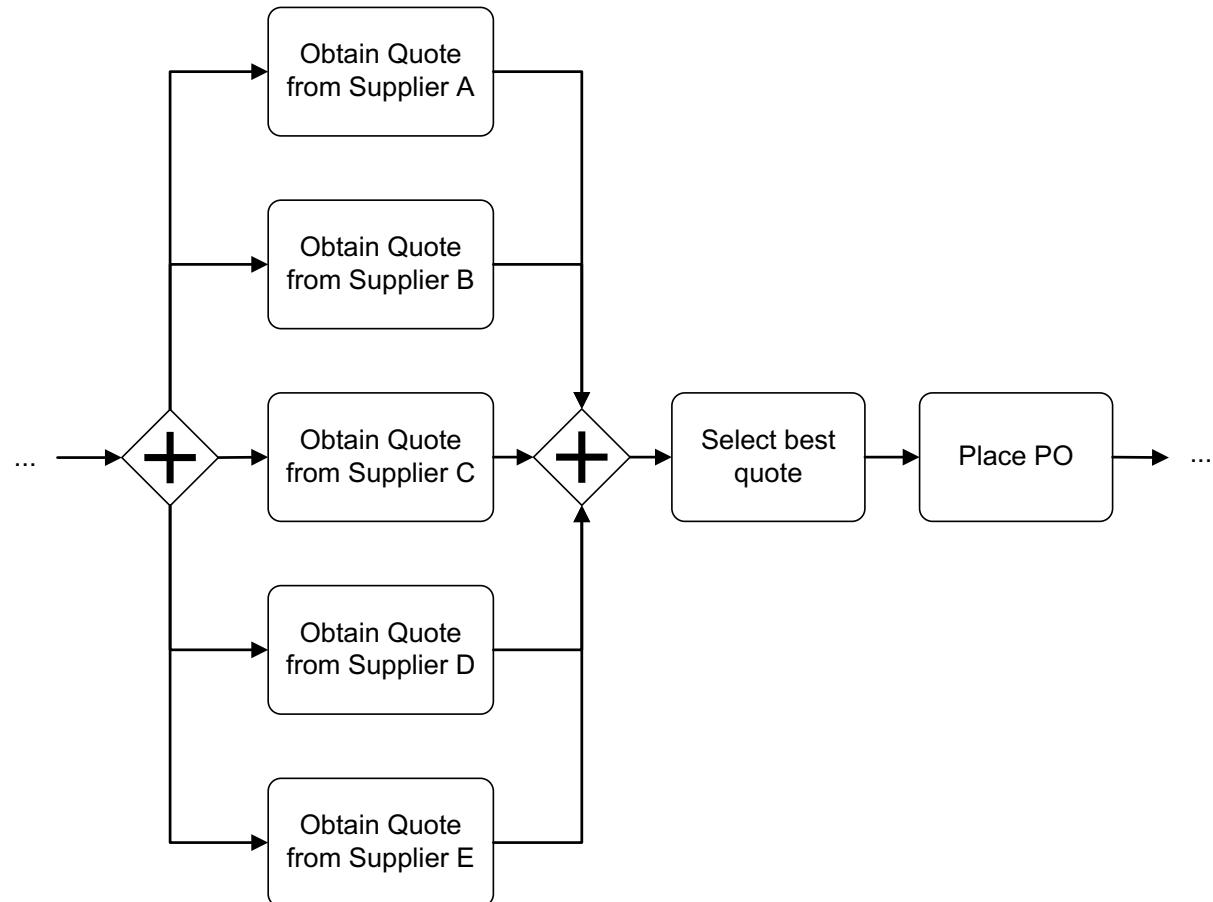


Beispiel: Activity Loop



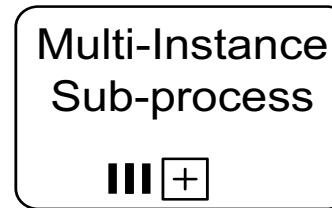
Parallele Mehrfachausführung – Motivation

Einkauf



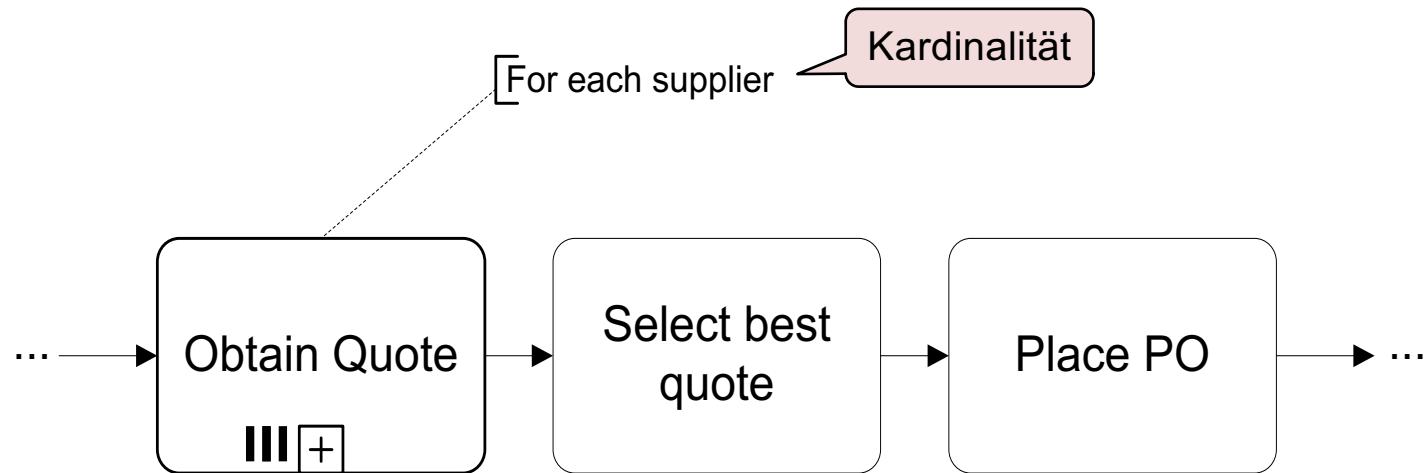
Parallele Mehrfachausführung – Multi-instance Activity

- Über eine Markierung (drei vertikale Linien) wird die *parallele Mehrfachausführung* kenntlich gemacht



- Hilfreich wenn die gleiche Aktivität für Entitäten oder Datenelemente mehrfach ausgeführt werden muss, z.B.:
 - Angebote von mehreren Lieferanten einholen
 - Verfügbarkeitsprüfung für jede Position eines Auftrags durchführen
 - Fragebögen von mehreren Zeugen in einem Versicherungsfall aufnehmen

Parallele Mehrfachausführung – Beispiel

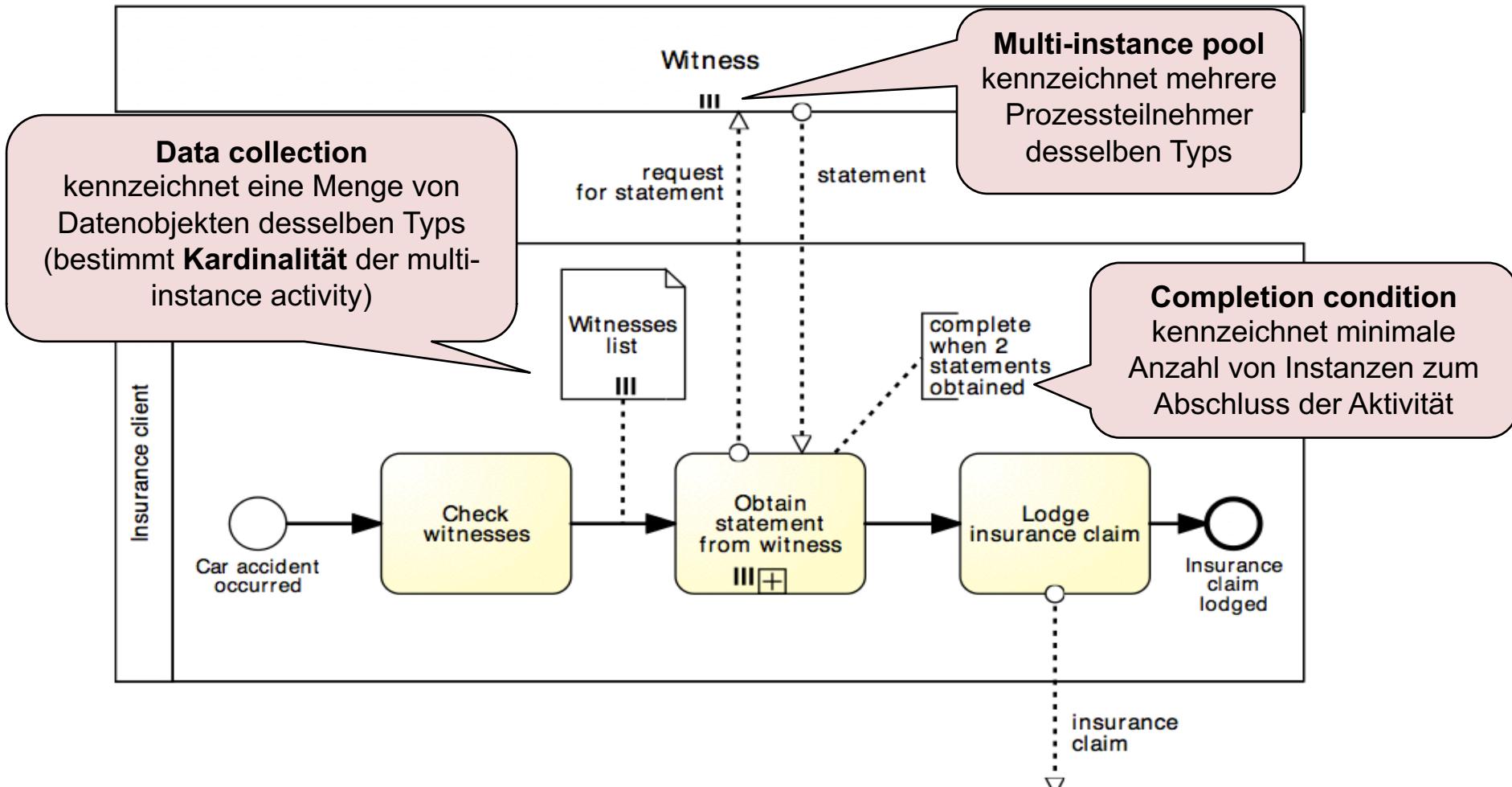


Parallele Mehrfachausführung – Beispiel 2

Abwicklung von Kfz-Versicherungsansprüchen

Nach einem Autounfall wird von den anwesenden Zeugen (witnesses) eine Aussage (statement) eingeholt, um den Versicherungsanspruch (insurance claim) geltend zu machen. Sobald die ersten beiden Aussagen vorliegen, kann der Anspruch bei dem Versicherungsunternehmen (insurance company) eingereicht werden, ohne die anderen Aussagen abzuwarten.

Parallele Mehrfachausführung – Data Collection & Multi-instance pool



Book: 4.2 Handling Events & 4.3.1 Process Abortion

EVENTS (FORTGESCHRITTEN)

Events in BPMN

Events: Modellieren Ereignisse die ohne Verzögerung während der Ausführung eines Prozesses auftreten

Je nach Platzierung im Prozessmodell beeinflussen Events den Prozessablauf unterschiedlich:

- *Start Event*: Token wird erstellt
- *End Event*: Token wird "zerstört"
- *Intermediate Event*: Token wartet im eingehenden sequence flow und überläuft Event sobald es eintritt



BPMN Eventtypen

Start Intermediate End



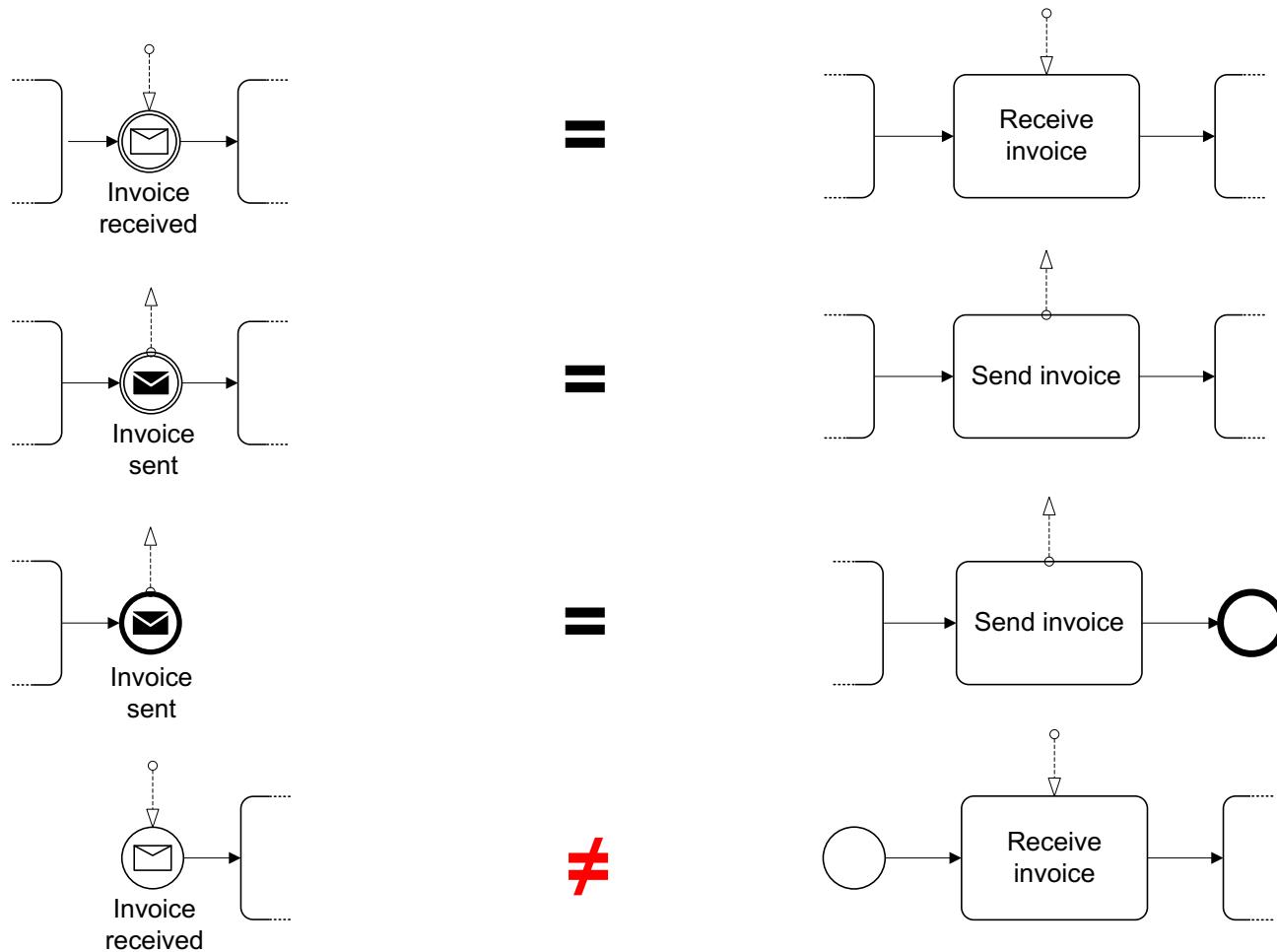
Untypisiertes Event – Definiert den Start (start) oder Abschluss (end) einer Prozessinstanz, ohne auf den Grund genauer einzugehen

Start Message Event – Die Prozessinstanz wird mit dem Eintreffen der Nachricht erstellt

End Message Event – Die Prozessinstanz wird mit dem Senden einer Nachricht beendet

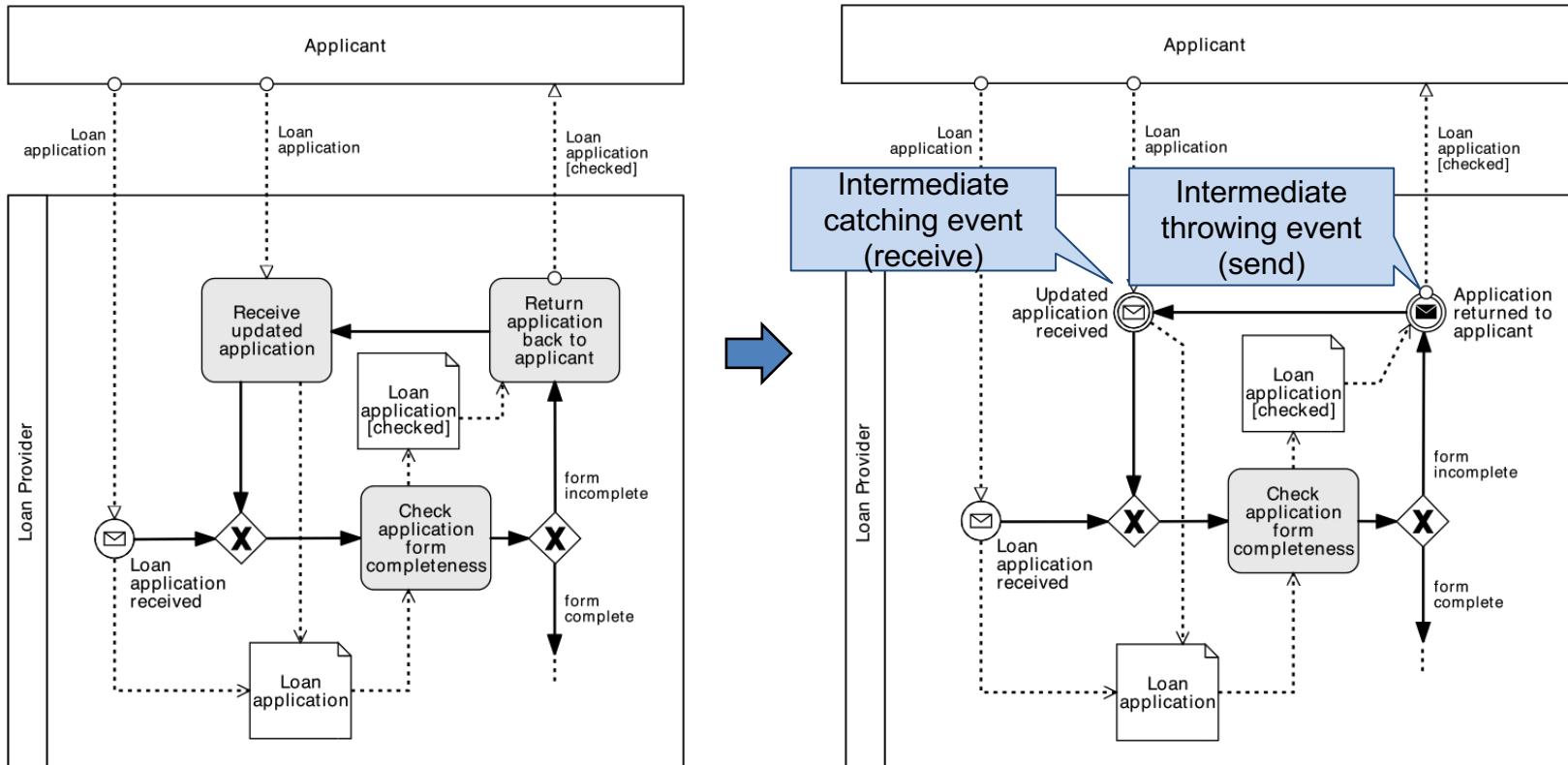
Intermediate Message Event – Während der Ausführung des Prozesses wird eine Nachricht empfangen (weißer Brief) oder gesendet (schwarzer Brief)

Message Event vs. Task - Vergleich



Message Event vs. Task – Best Practice

Message Event sollte genau dann genutzt werden, wenn die korrespondierende Aktivität lediglich eine Nachricht empfängt oder sendet (und sonst nichts tut)



Timer Events

Start Intermediate End



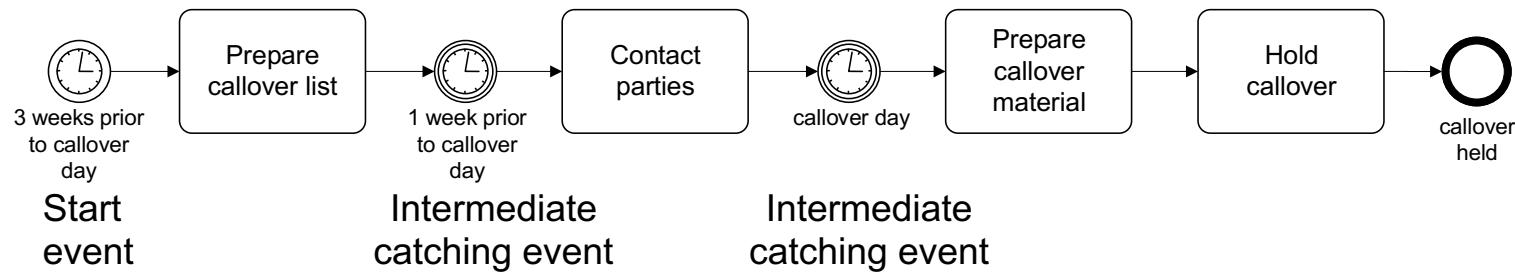
Start Timer Event – Definiert den Start einer Prozessinstanz zu festen Daten/Uhrzeiten, z.B. jeden Freitag um 18 Uhr.



Intermediate Timer Event – Wird zu einer bestimmten Zeit ausgelöst, oder nachdem ein Zeitintervall seit der Aktivierung des Events überschritten wurde

Timer Events – Beispiel

In a small claims tribunal, callovers occur once a month to set down the matter for the upcoming trials. The process for setting up a callover starts **three weeks prior** to the callover day, with the preparation of the callover list containing information such as contact details of the involved parties and estimated hearing date. **One week prior** to the callover, the involved parties are notified of the callover date. Finally, **on the callover day**, the callover material is prepared and the callover is held.



Event-based Gateway

Bei dem XOR-split wird die Verzweigung gewählt, bei der die Bedingung basierend auf den verfügbaren Daten zutrifft (z.B. Artikel verfügbar)

→ Die Entscheidung kann direkt mit Eintreffen des Tokens getroffen werden

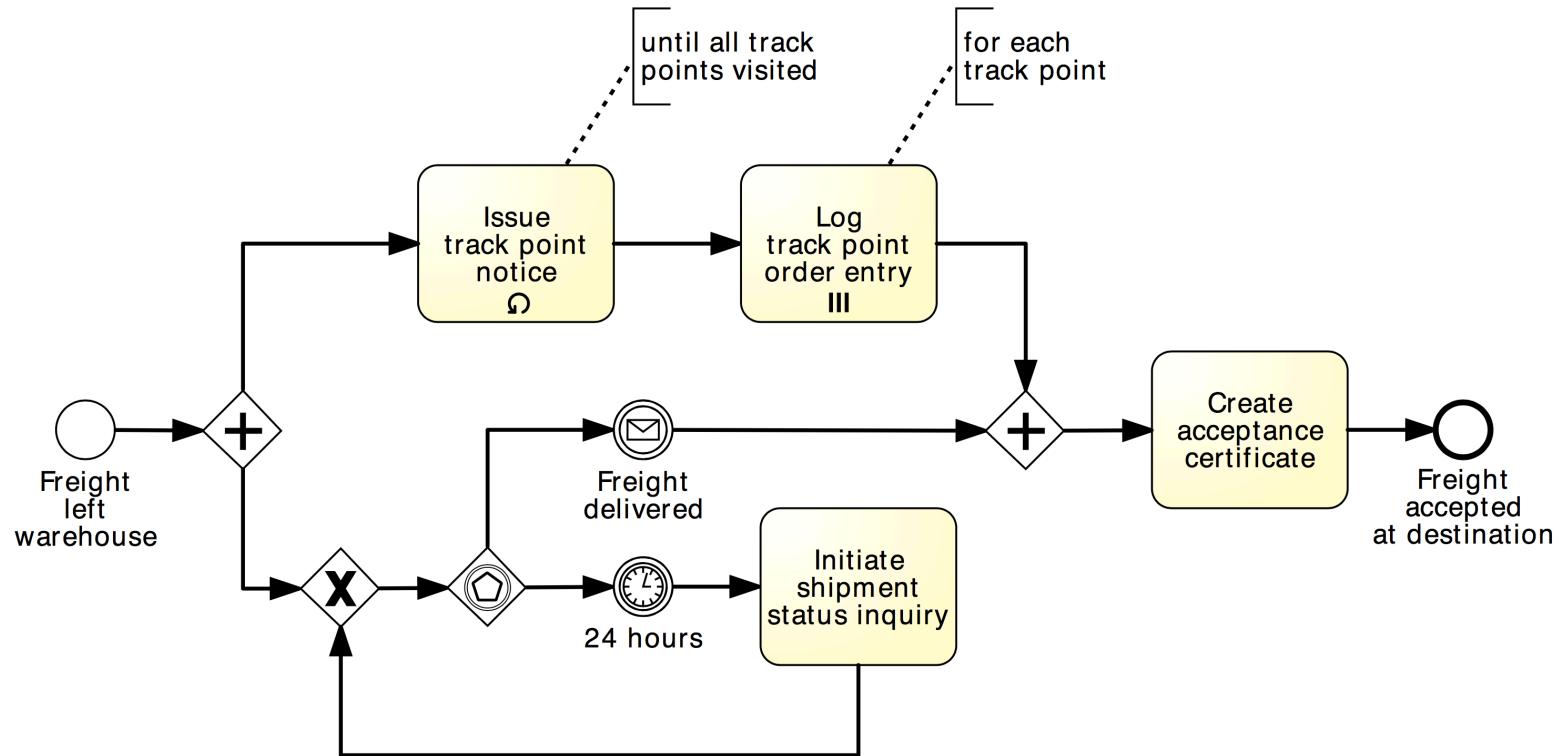
Manchmal muss die Entscheidung verzögert werden bis ein Event eintritt

→ Die Entscheidung basiert auf einem Wettrennen zwischen versch. Events

Zwei Arten von XOR-split:

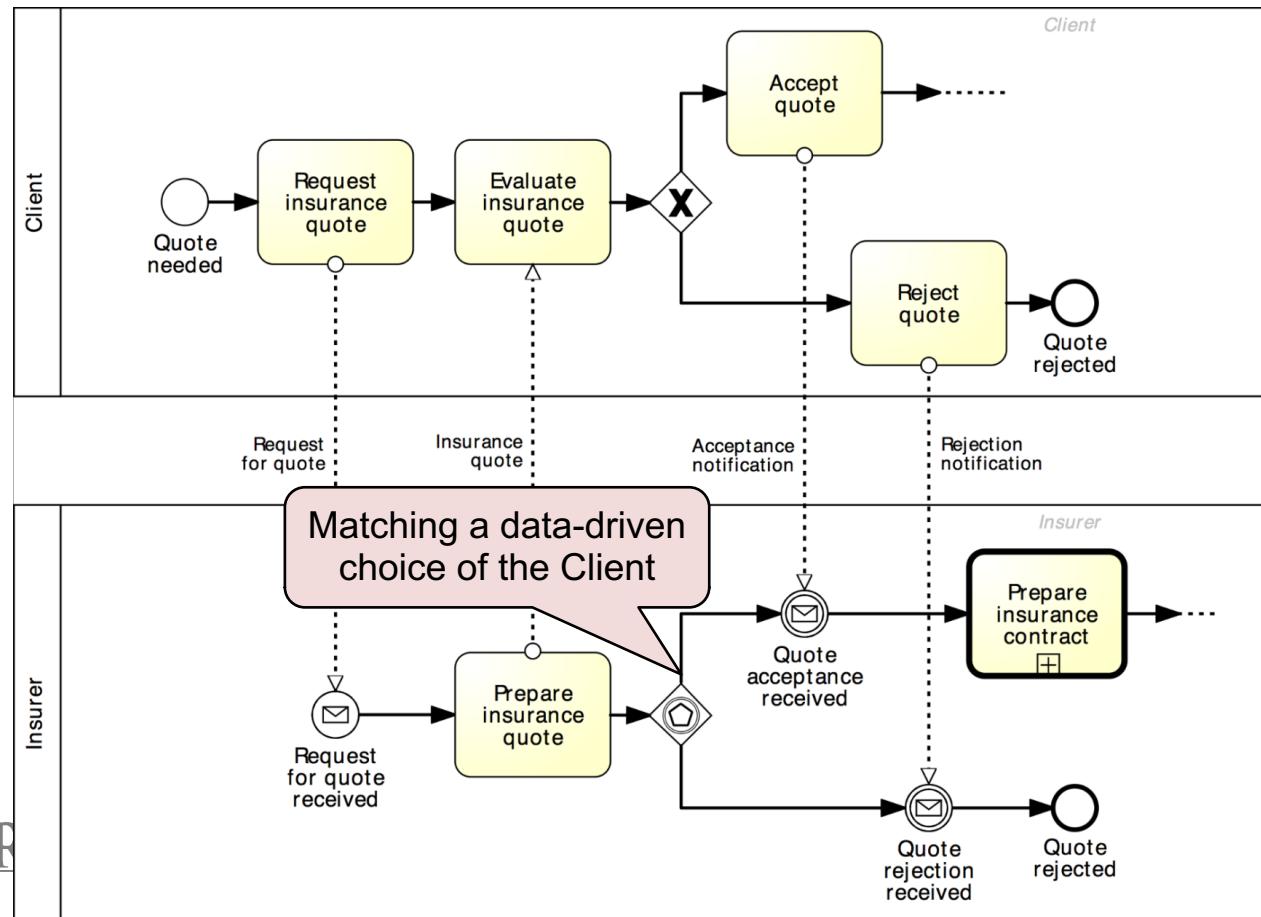


Event-based Gateway - Beispiel



Event-based Gateway- Beispiel (2)

Der event-based XOR-split kann auch als Gegenstück zu einer internen Entscheidung einer anderen Geschäftspartei genutzt werden



Prozessabbruch

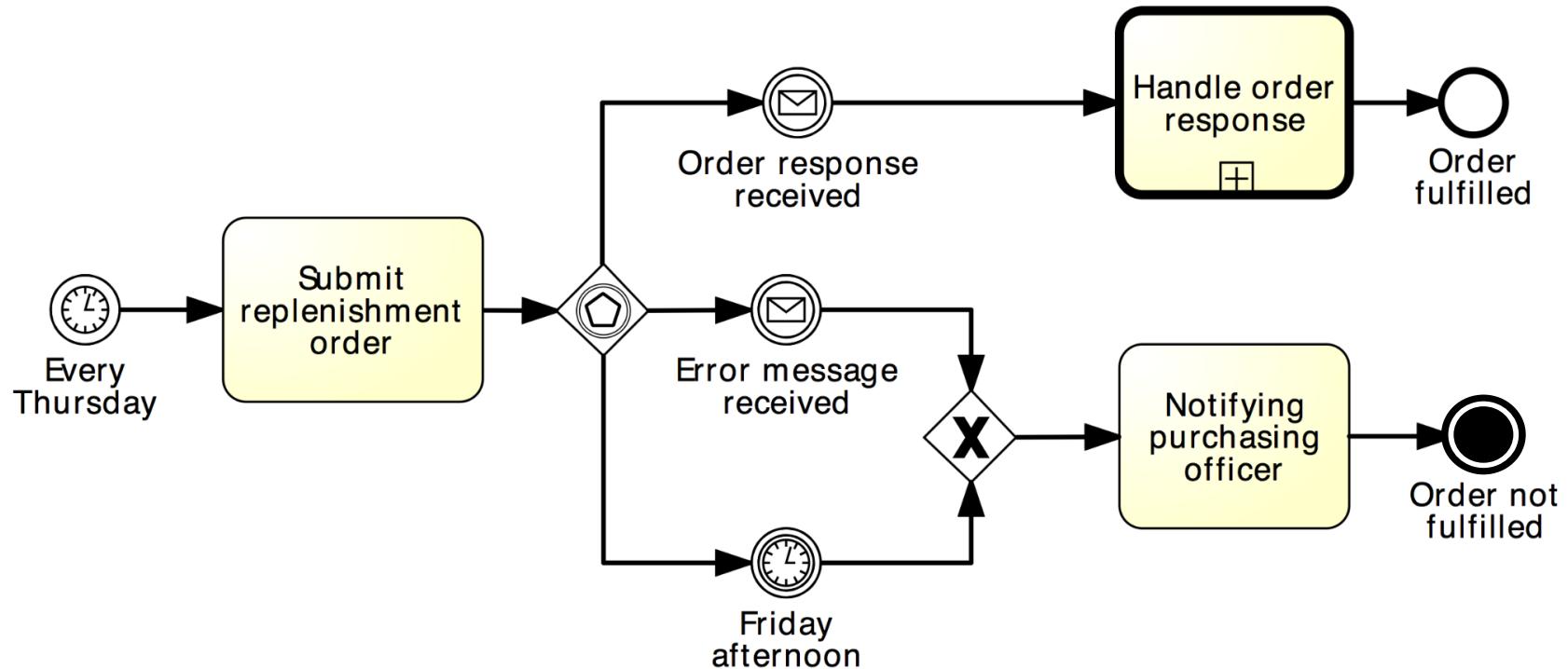
Exceptions sind Events die aufzeigen, dass ein Prozess von seinem “normalen” Verlauf abweicht.

Die einfachste Form einer Exception ist das *Terminate End Event*: es zwingt den gesamten Prozess zum Abbruch und zerstört alle bestehenden Tokens einer Prozessinstanz



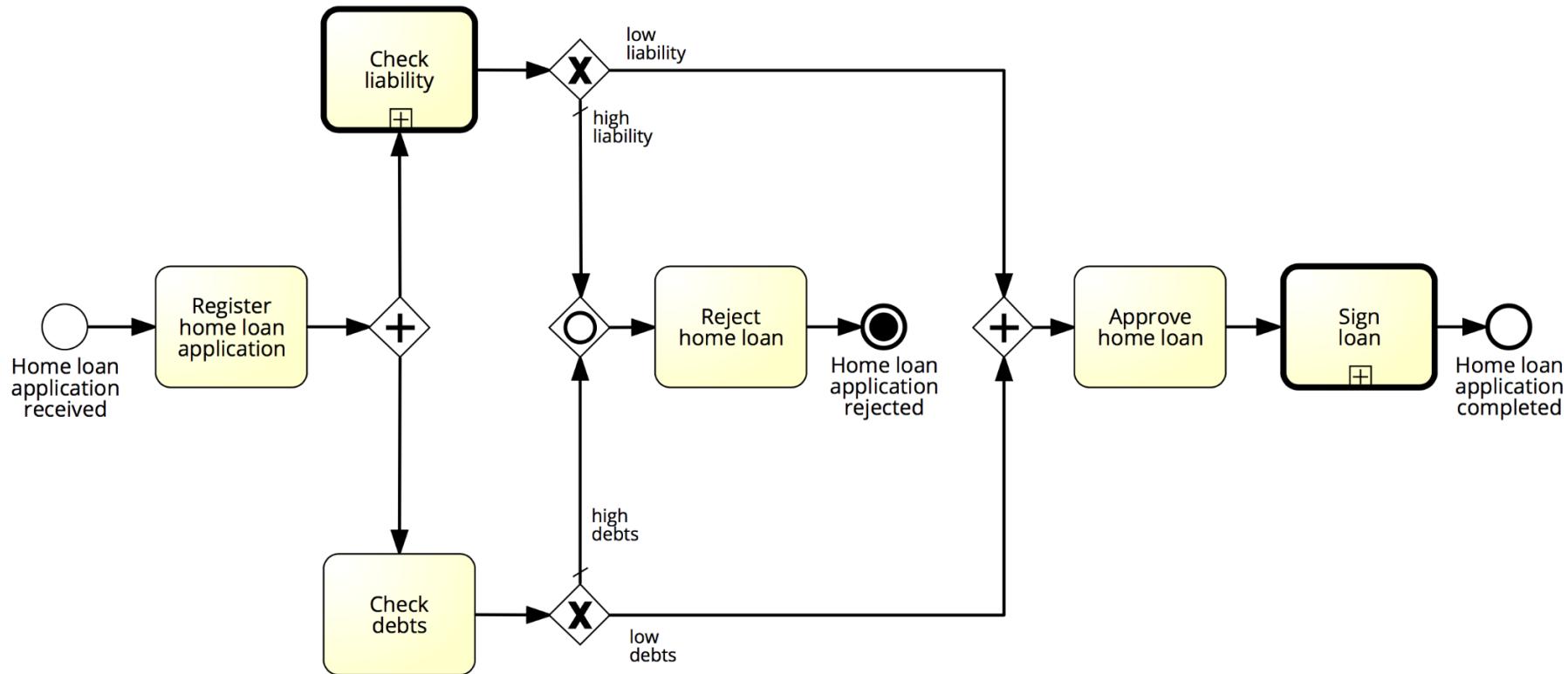
Prozessabbruch – Beispiel 1

- Hier wird das *Terminate End Event* lediglich genutzt um einen negativen Ausgang zu kennzeichnen



Prozessabbruch – Beispiel 2

Der Prozess wird abgebrochen und alle bestehenden Tokens zerstört...



Recap – Events

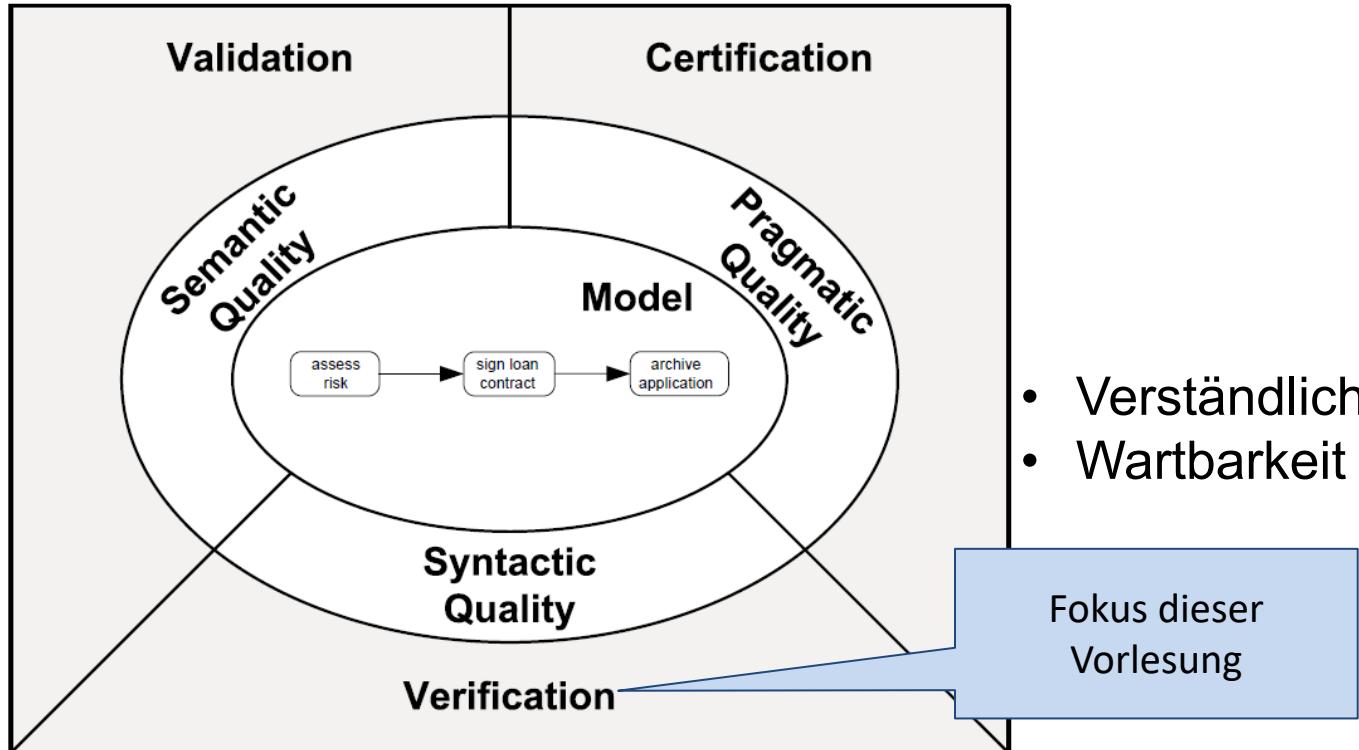
	Start	Intermediate	End
	Catching	Throwing	
Untyped: indicate start point, state changes or final states.			
Message: Receiving and sending messages.			
Timer: Cyclic timer events, points in time, time spans or timeouts.			
Terminate: Triggering the immediate termination of a process.			

Book: 5.4 Process Model Quality Assurance

QUALITÄTSSICHERUNG

Qualitätsdimensionen

- Validität
- Vollständigkeit



- Strukturelle Korrektheit (structural correctness)
- Soundness (behavioral correctness)

Syntaktische Qualität: Verifikation

- Syntaktische Qualität: Übereinstimmung eines Prozessmodells mit den syntaktischen Regeln der Modellierungssprache.
- Zwei Arten von syntaktischen Regeln: *Strukturregeln* (*structural rules*) und *Verhaltensregeln* (*behavioral rules*).
- Ein Modell hat eine hohe syntaktische Qualität, wenn es syntaktisch korrekt ist:
 - Strukturell korrekt +
 - Verhaltenskorrekt



Strukturelle Korrektheit

Ein Modell ist strukturell korrekt wenn es folgende syntaktischen Regeln erfüllt:

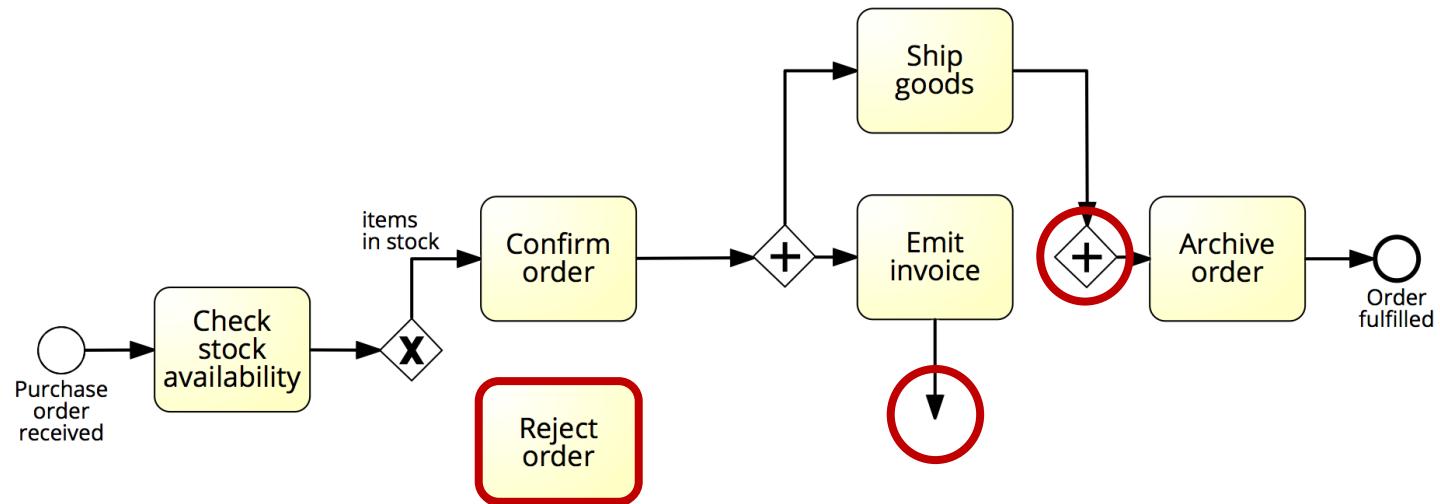
1. Regeln auf Elementebene:

- *Aktivitäten*: mindestens ein eingehender und ein ausgehender sequence flow
- *start/end events*: keine eingehenden/ausgehenden Verbindungen
- *Gateways*: genau ein eingehender und mindestens zwei ausgehende Verbindungen (Splits) bzw. mindestens zwei eingehende und genau eine ausgehende Verbindungen (Joins)
- *sequence flows*: muss zwei Knoten im gleichen Pool verbinden
- ...

2. Regel auf Modellebene: alle Knoten müssen sich auf einem Pfad von einem *start*- zu einem *end event* befinden

- impliziert, dass ein Modell mindestens ein *start*- und ein *end event* haben sollte

Beispiel: Strukturelle Korrektheit

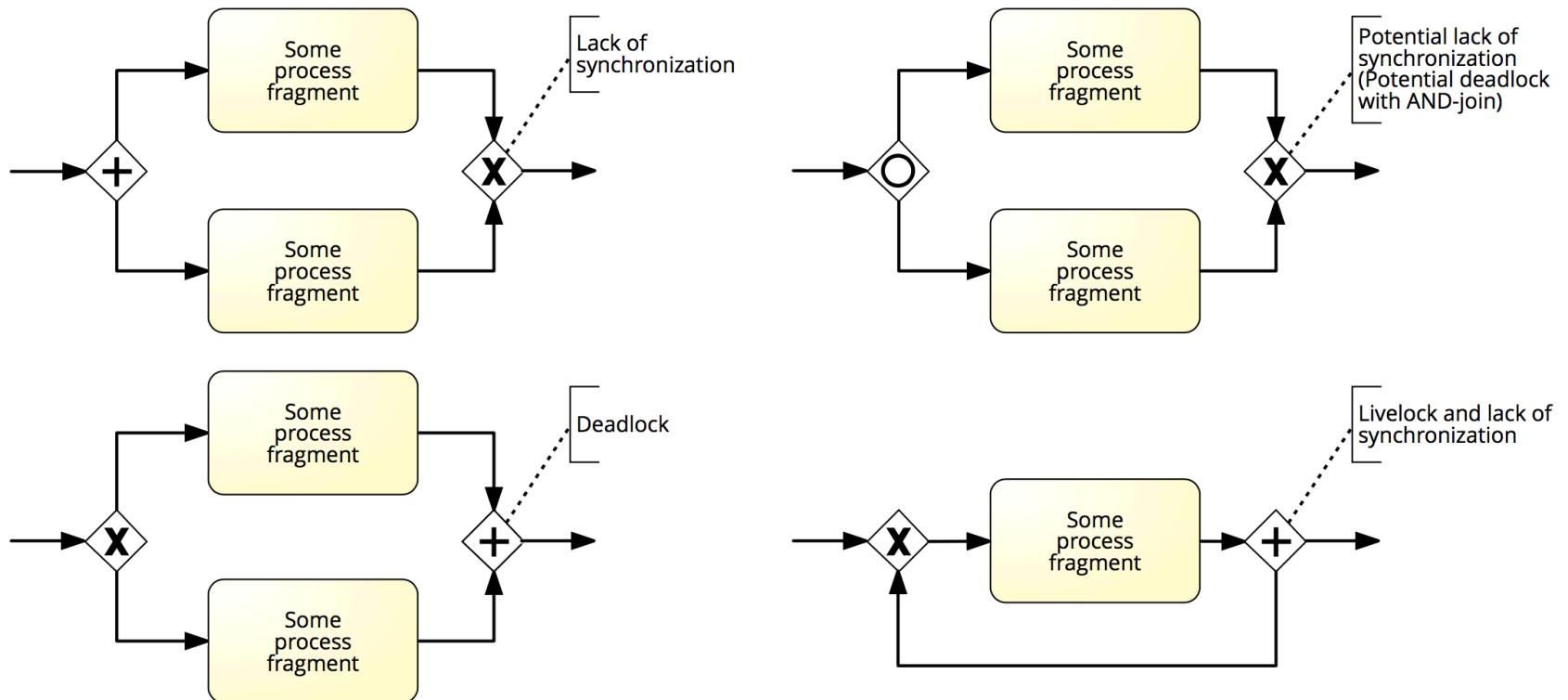


Soundness (behavioral correctness)

Ein Model ist “sound” wenn es folgende Regeln erfüllt:

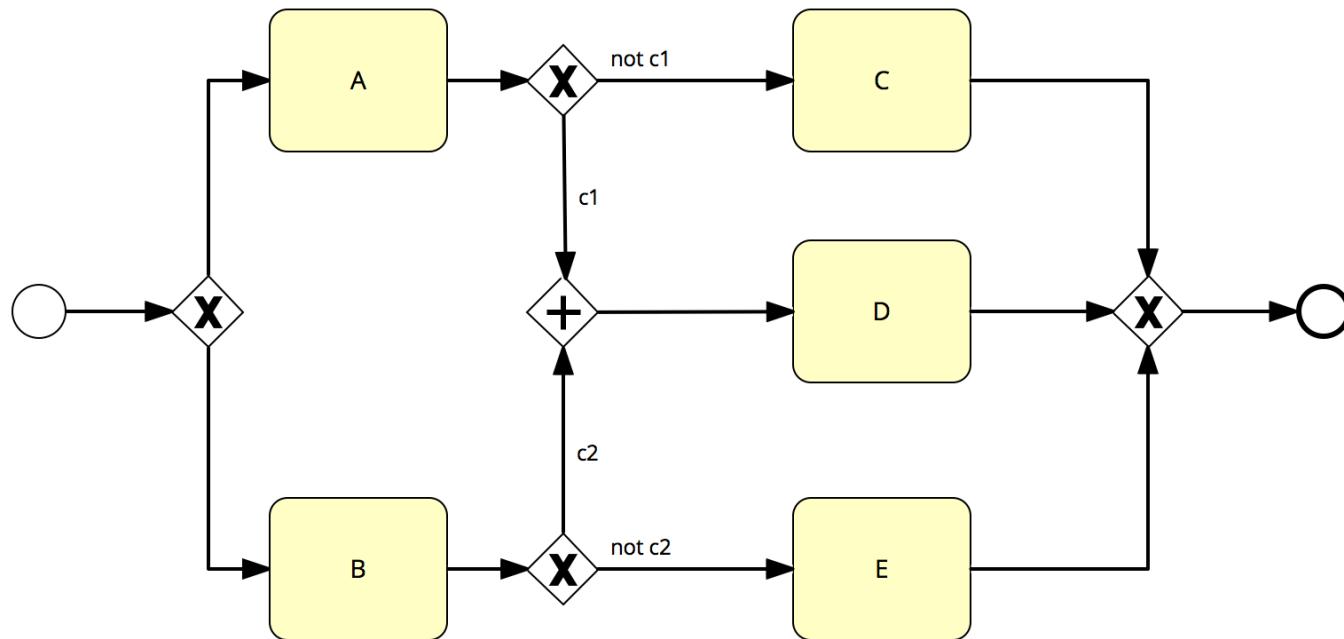
1. *option to complete*: jeder laufende Prozess muss irgendwann abschließen,
2. *proper completion*: bei Abschluss des Prozesses muss sich jeder Token der Prozessinstanz in einem anderen end event befinden
3. *no dead activities*: jede Aktivität kann potentiell in einer Prozessinstanz ausgeführt werden.

Beispiel: Gateway Mismatch



Beispiel: no option to complete (deadlock)

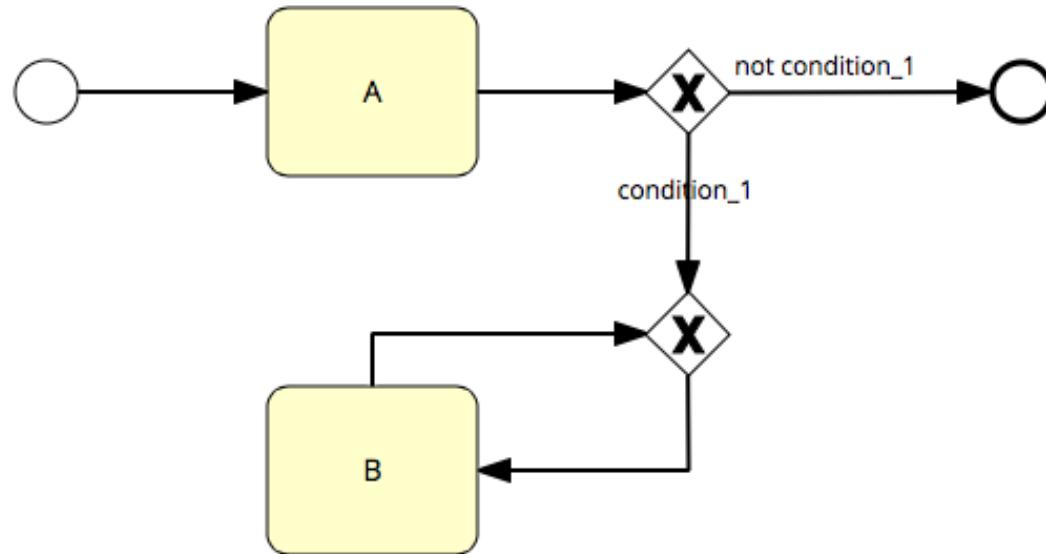
Wenn $c1$ nach der Ausführung von A wahr ist, oder $c2$ nach der Ausführung von B, kann die Instanz nie abgeschlossen werden (Deadlock)



Hinweis: Das Modell verletzt außerdem die “no dead activity” Regel (D)

Beispiel: no option to complete (livelock)

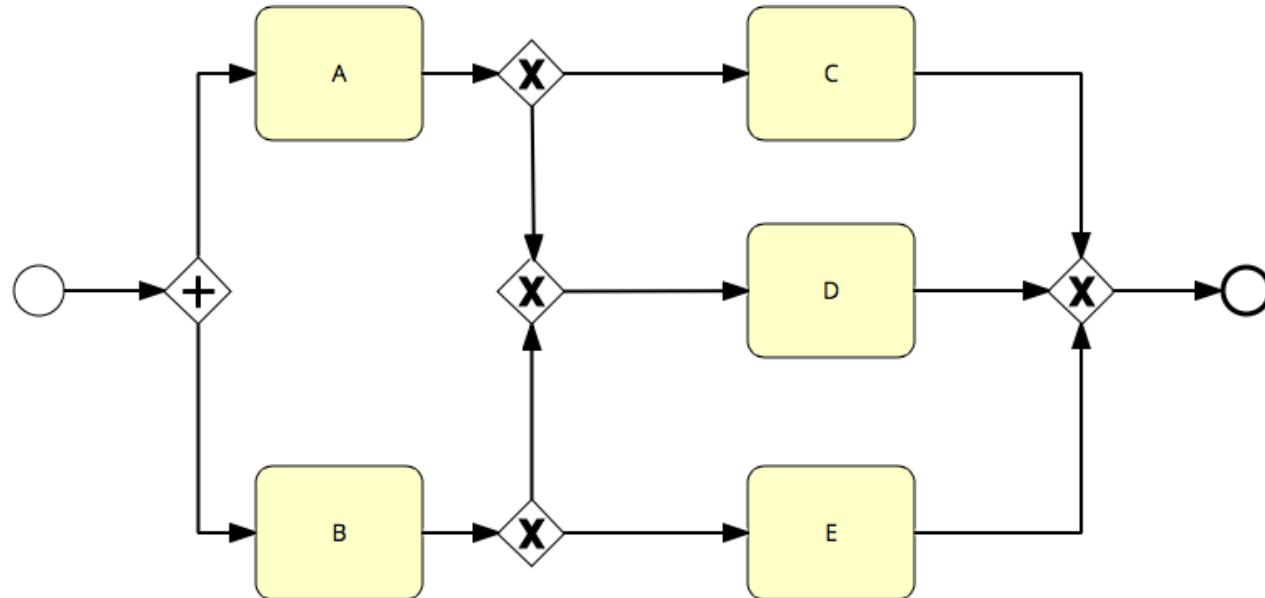
Wenn condition_1 wahr ist, kann die Instanz nie abschließen, da Aktivität B unendlich oft wiederholt wird (livelock)



Hinweis: Dieses Modell ist auch **strukturell inkorrekt**, da sich B nicht auf einem Pfad zum end event befindet.

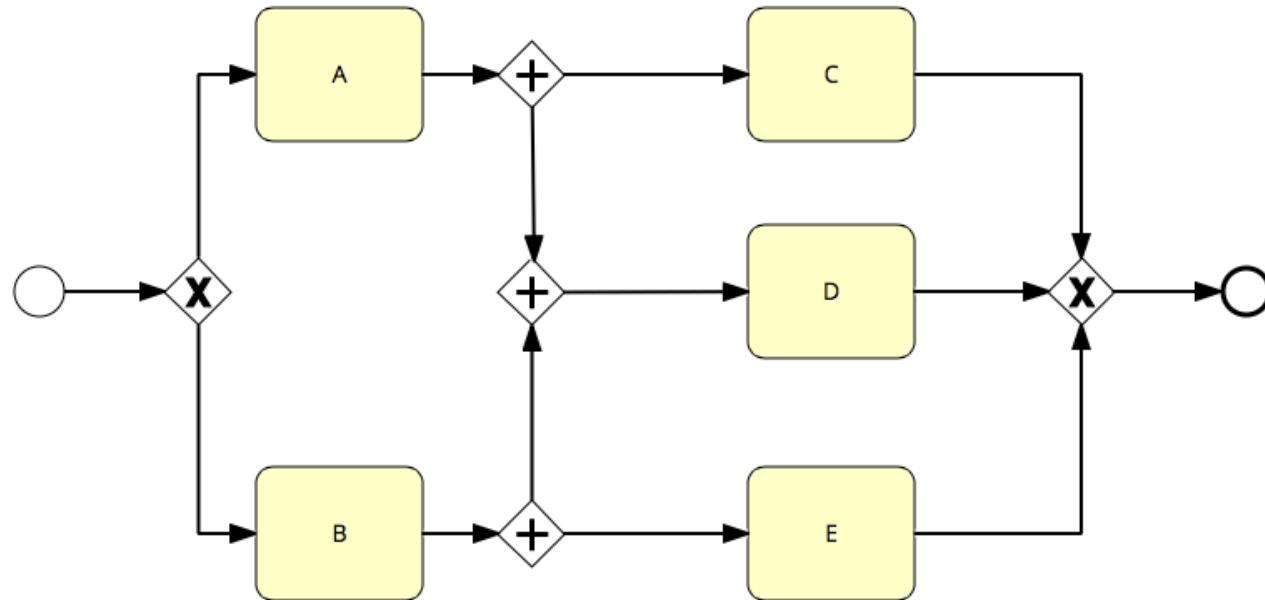
Beispiel: no proper completion

Beim Abschluss des Prozesses befinden sich zwei tokens im end event (fehlende Synchronisierung)



Beispiel: dead activity

Der Prozess kann immer abschließen, die Aktivität D wird jedoch nie ausgeführt werden.

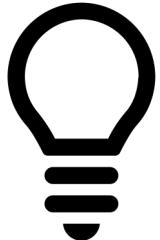


Hinweis: Das Modell enthält außerdem einen Deadlock, da ein Token vor dem AND-JOIN hängen bleibt. Die Instanz schließt jedoch immer mit dem Eintreffen des ersten Tokens im end event ab.

BPMN Soundness – Formalisierung?

Ausgangslage

- Die Soundness eines Petri Nets kann formal mittels Erreichbarkeitsgraph geprüft werden
- Geschäftsprozesse werden i.d.R. mit BPMN modelliert



Idee:

1. Konvertieren BPMN Modell in Petri Net
2. Prüfung Soundness über Petri Net Erreichbarkeitsgraph

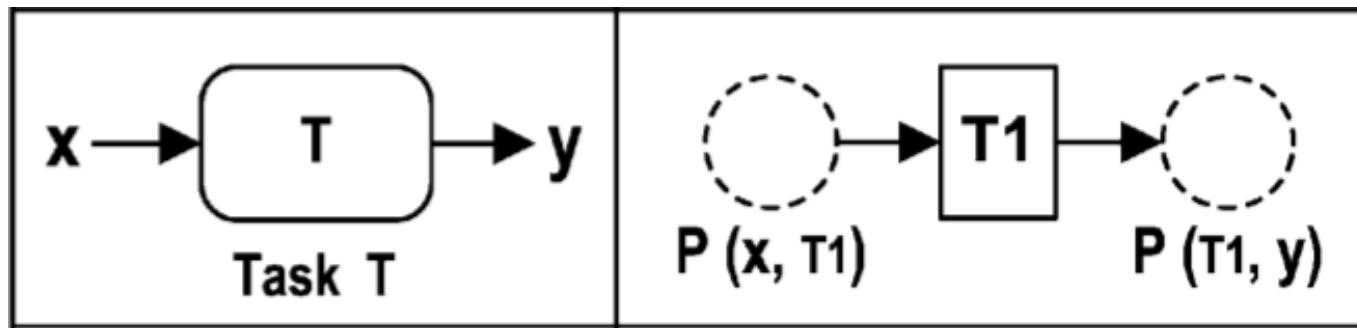


Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). *Semantics and analysis of business process models in BPMN*

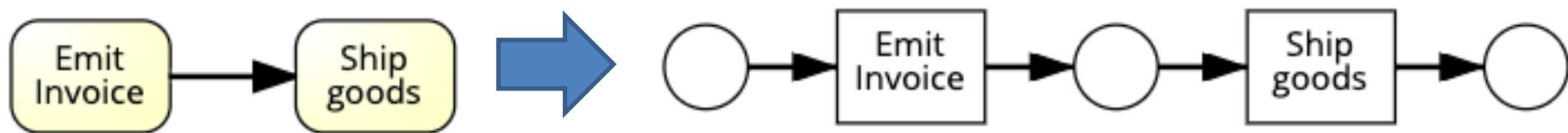
BPMN VERIFIKATION MIT HILFE VON PETRI-NETZEN

Übersetzung von Aktivitäten

- Die Ausführung einer Aktivität in BPMN entspricht dem Feuern einer Petri Net Transaktion



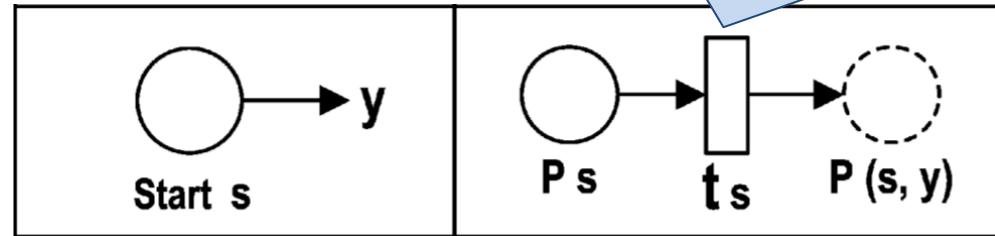
Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). *Semantics and analysis of business process models in BPMN*



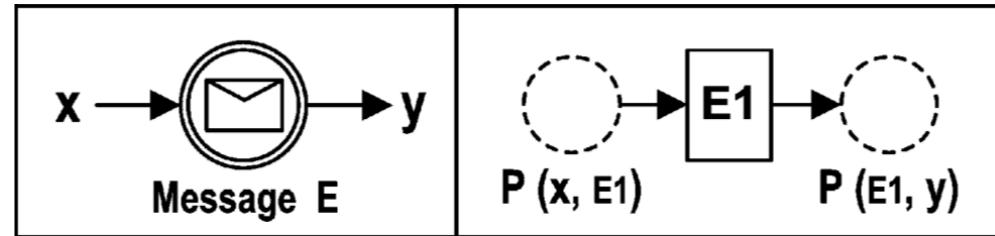
Events

Silent Transition: Transition ohne Label

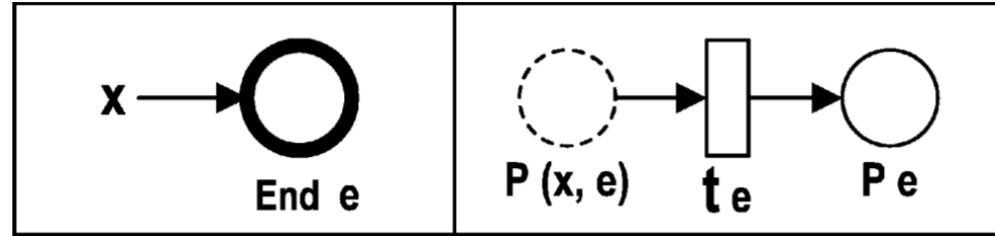
- Start event



- Intermediate event

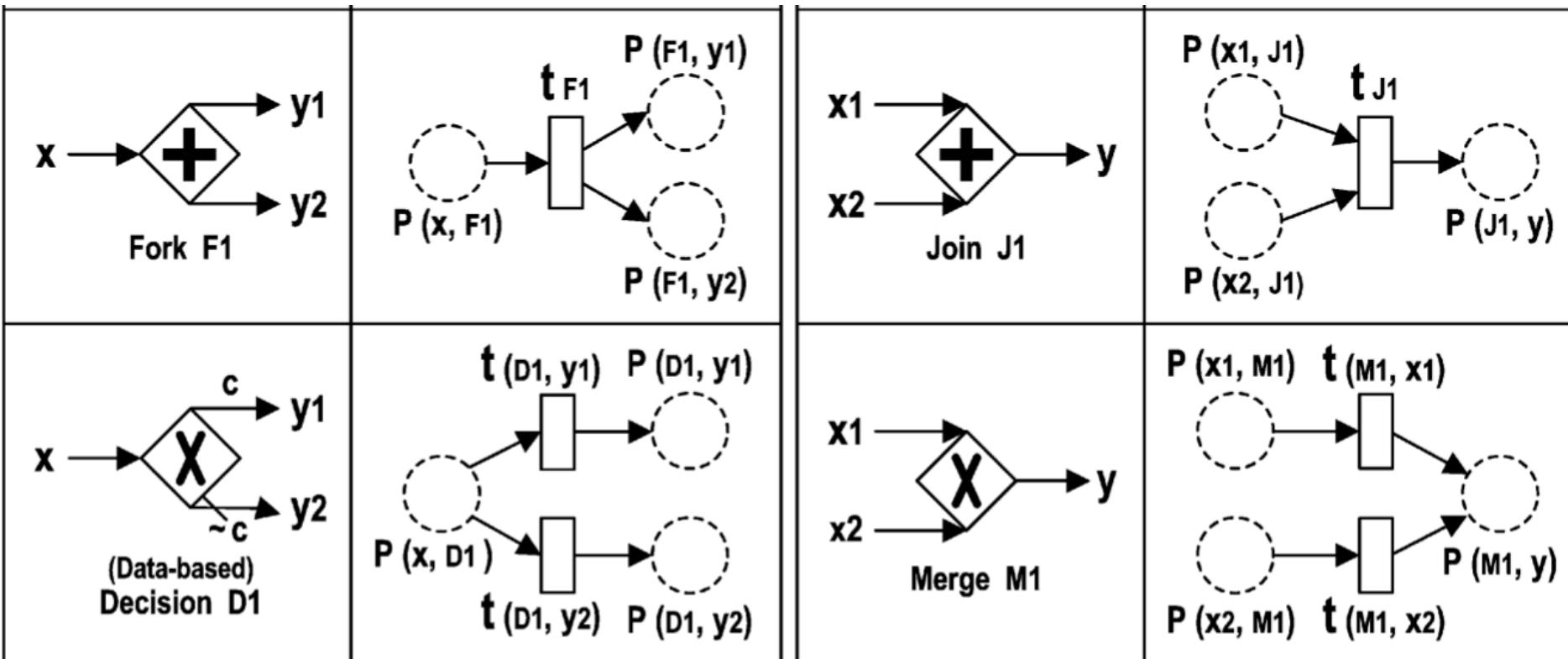


- End event



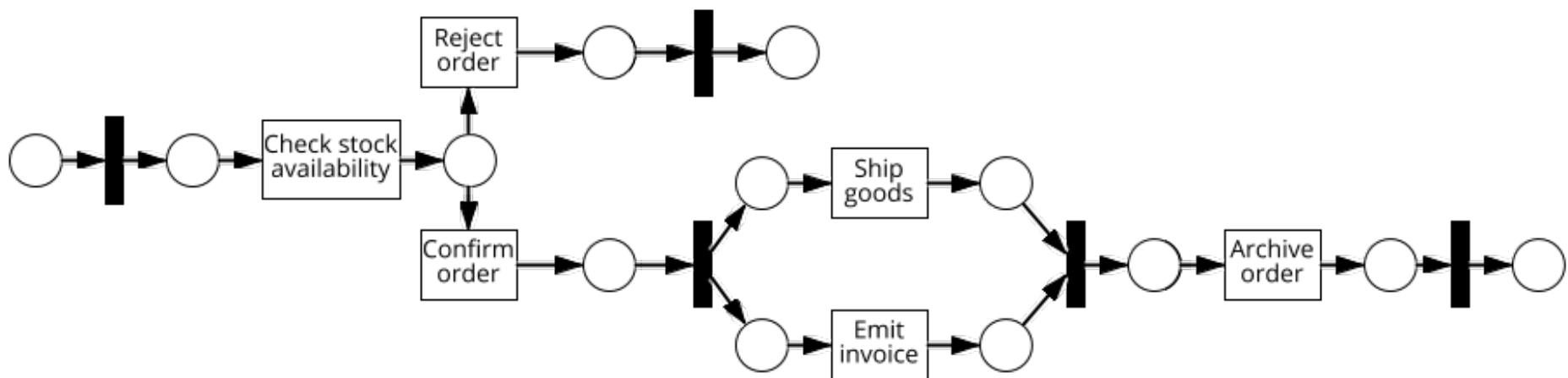
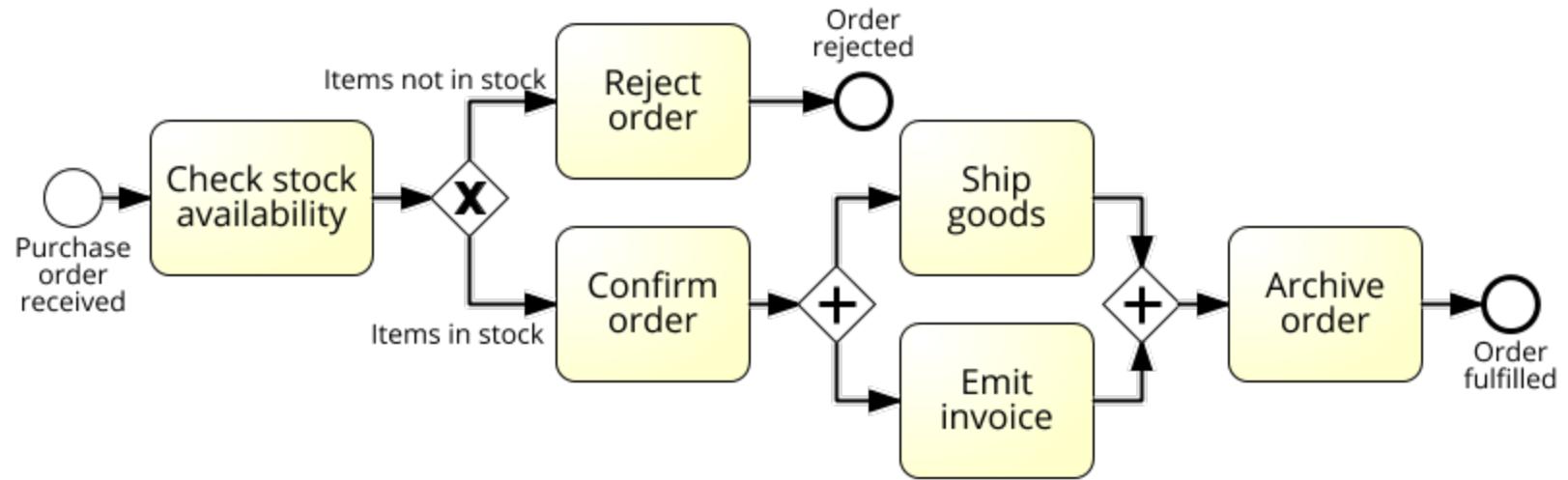
Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). *Semantics and analysis of business process models in BPMN*

Gateways

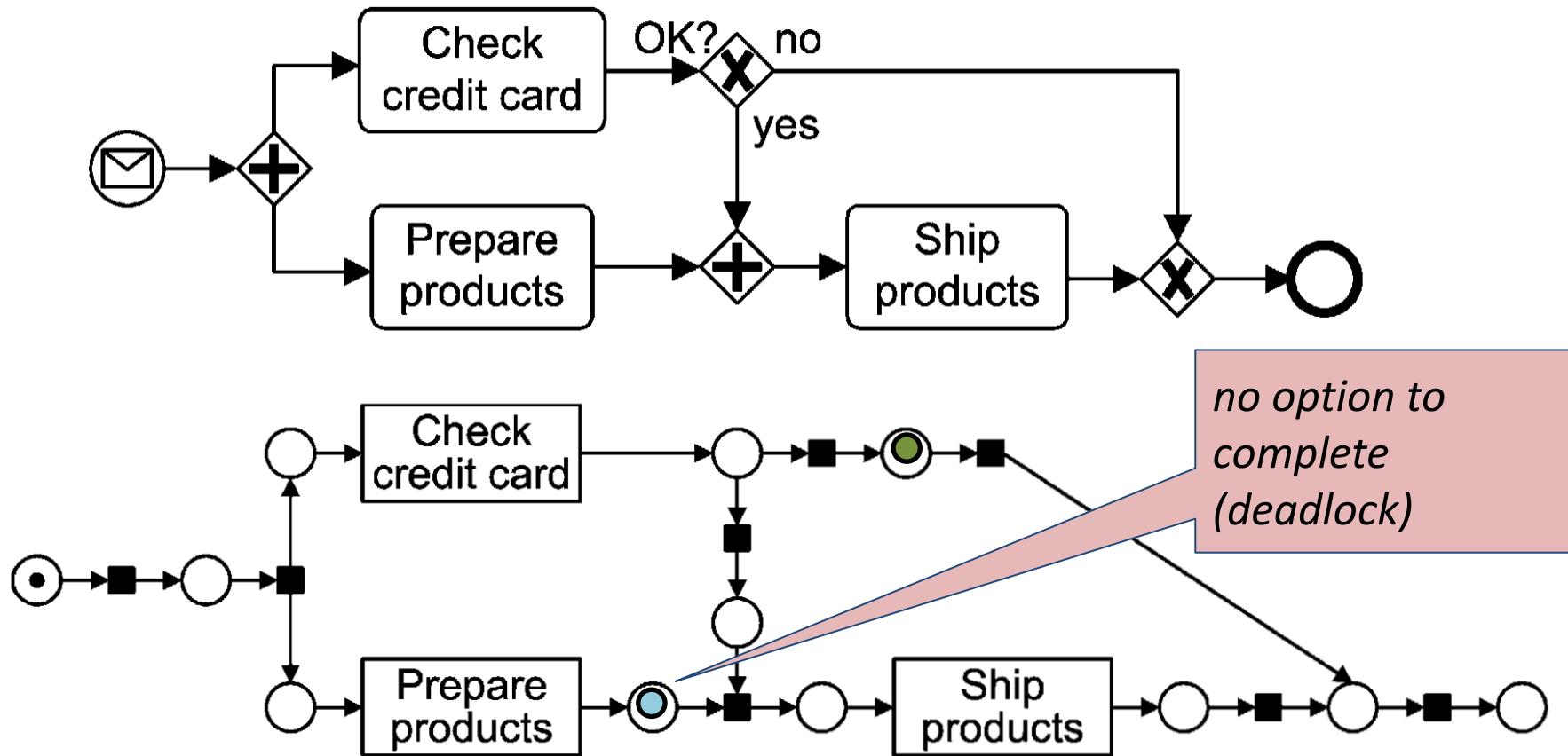


Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). *Semantics and analysis of business process models in BPMN*

Beispiel – Order-to-Cash



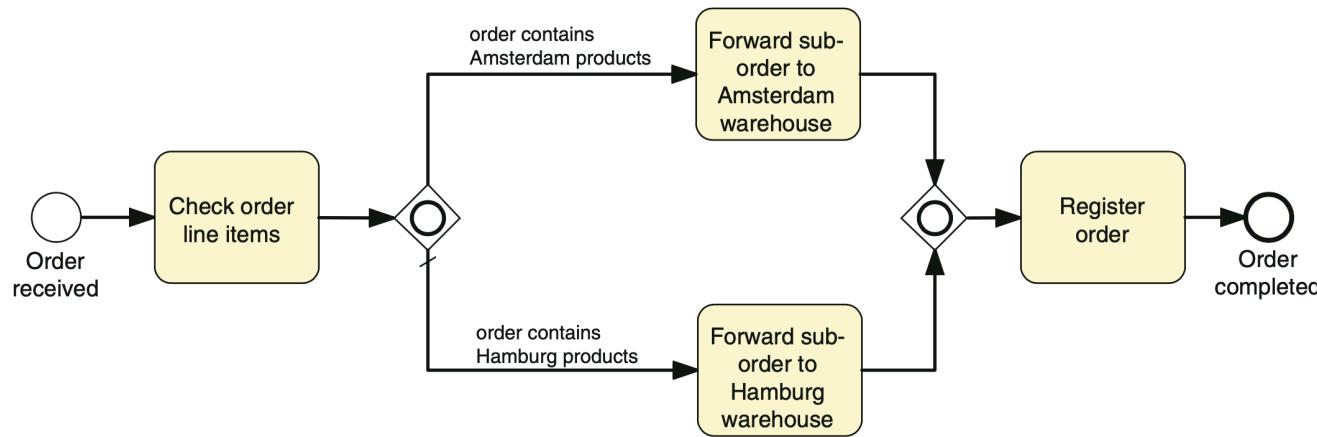
Beispiel – Order Process



Dijkman, R. M., Dumas, M., & Ouyang, C. (2008). *Semantics and analysis of business process models in BPMN*

Einschränkungen BPMN → Petri Netz

- Nicht jeder BPMN Prozess lässt sich eindeutig in ein Petri Netz übersetzen
- Beispiel: OR-Join hat keine eindeutige Semantik



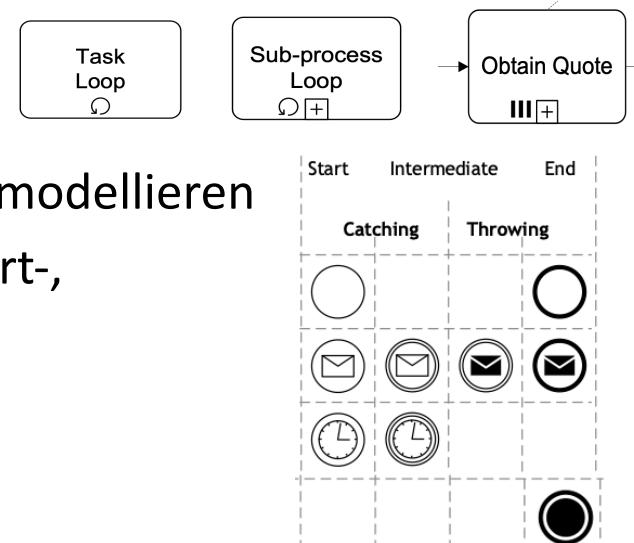
Hinweis: OR-Gateways sollten generell vermieden werden^[1]

[1] Mendling, J., Reijers, H. A., & van der Aalst, W. M. P. (2010). *Seven process modeling guidelines (7PMG)*

Recap

BPMN Syntax:

- BPMN bietet “Marker” auf Aktivitätsebene, um wiederholende/parallele Tasks/Unterprozesse zu modellieren
- BPMN bietet verschiedene Eventtypen, die als start-, intermediate-, und end event auftreten können



BPMN syntaktische Qualität:

- Drei Qualitätsaspekte zur Bewertung eines Modells: syntaktische, semantische, und pragmatische Qualität
- Syntaktische Qualität ist abhängig von der strukturellen Korrektheit und der Soundness des Modells
- Übersetzung BPMN → Petri Netz erlaubt formalisierte Überprüfung der Soundness eines Modells