

# LFCX:Diagramme

## für Fachinformatikerinnen<sup>1</sup>

Karsten Reincke

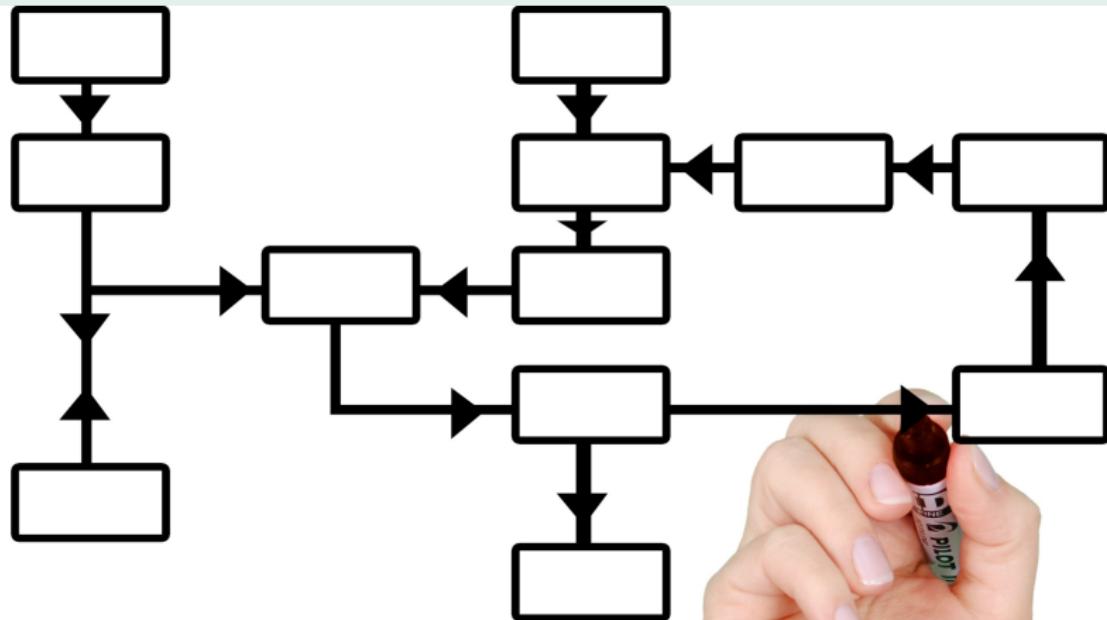
GS-LDK

8. Januar 2026

---

<sup>1</sup> Diese Präsentation stammt aus dem Open-Source-Projekt [proScientia.ltx](#), ist [CC-BY-4.0](#) lizenziert und wurde auf Basis von [proScientia.ltx](#) entwickelt.

# LFCX:Diagramme:UML



Crashkurs UML

# LFCX:Diagramme:UML:Systematik<sup>2</sup>

## Strukturdiagramme:

- Klassendiagramm
- Objektdiagramm
- Komponentendiagramm
- Kompositionsstrukturdiagramm
- Verteilungsdiagramm
- Paketdiagramm
- Profildiagramm
- Anwendungsfalldiagramm

## Verhaltensdiagramme:

- Aktivitätsdiagramm
- Interaktionsdiagramm
  - Interaktionsübersicht
  - Kommunikationsdiagramm
  - Sequenzdiagramm
  - Zeitdiagramm
- Zustandsdiagramm
- Protokollautomat

Hinweis: Das Anwendungsfalldiagramm wird oft auch als Verhaltensdiagramm klassifiziert

<sup>2</sup>nach Bern Oestereich und Axel Scheithauer: Die UML-Kurzreferenz 2.5 für die Praxis. kurz, bündig, ballastfrei, 6. Aufl., Print, München: De Gruyter / Oldenbourg, 2014, ISBN: 978-3-486-74909-0, S. 8.

# LFCX:Diagramme:UML:Kontext

## Structural Diagrams:

- **Class Diagram** (*beschreibt Klassen von Objekten, ihre Eigenschaften, Operationen und Beziehungen untereinanderS*)
- ...
- **Object Diagram** (*ist eine zur Laufzeit vorhandene Einheit und ein Exemplar einer Klasse*)
- **Use Case Diagram** (*zeigt Akteure, Anwendungsfälle und deren Beziehungen untereinander*)

## Behavioral Diagrams:

- **Activity Diagram** (*beschreibt einen Ablauf mittels Knoten und Kontrollflüssen*)
- Interaktionsdiagramm
- ...
- **Sequence Diagram** (*zeigt eine Reihe von Nachrichten, die von den Beteiligten in einer bestimmten Reihenfolge ausgetauscht werden.*)
- **State Diagram** *beschreibt den Zustand (Stand der jeweiligen Daten) der Komponenten eines Systems*
- ...

# LFCX::UML:Aktivitätsdiagramm:Symbole



**Startknoten:** markiert den Anfang des Kontrollflusses



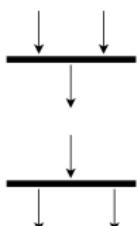
**Endknoten:** markiert das Ende aller Kontrollflüsse



**Aktivität:** "Arbeitsschritt" in einem Prozess



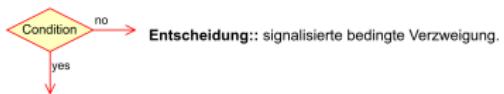
**Connector:** verbindet andere Elemente im Diagramm



**Joint:** synchronisiert nebenläufige Aktionen



**Fork:** Startet nebenläufige Aktionen



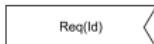
**Entscheidung:** signalisierte bedingte Verzweigung.



**Kommentar:** erlaubt den Diagrammautoren textuelle Ergänzungen, die nicht direkt ins Diagramm gehören



TurnOn



Req(Id)



Diagram Heading



**Loop:** Clustert Activities als Schleifen Body.



**Shallow History:** Verweis im Dokument

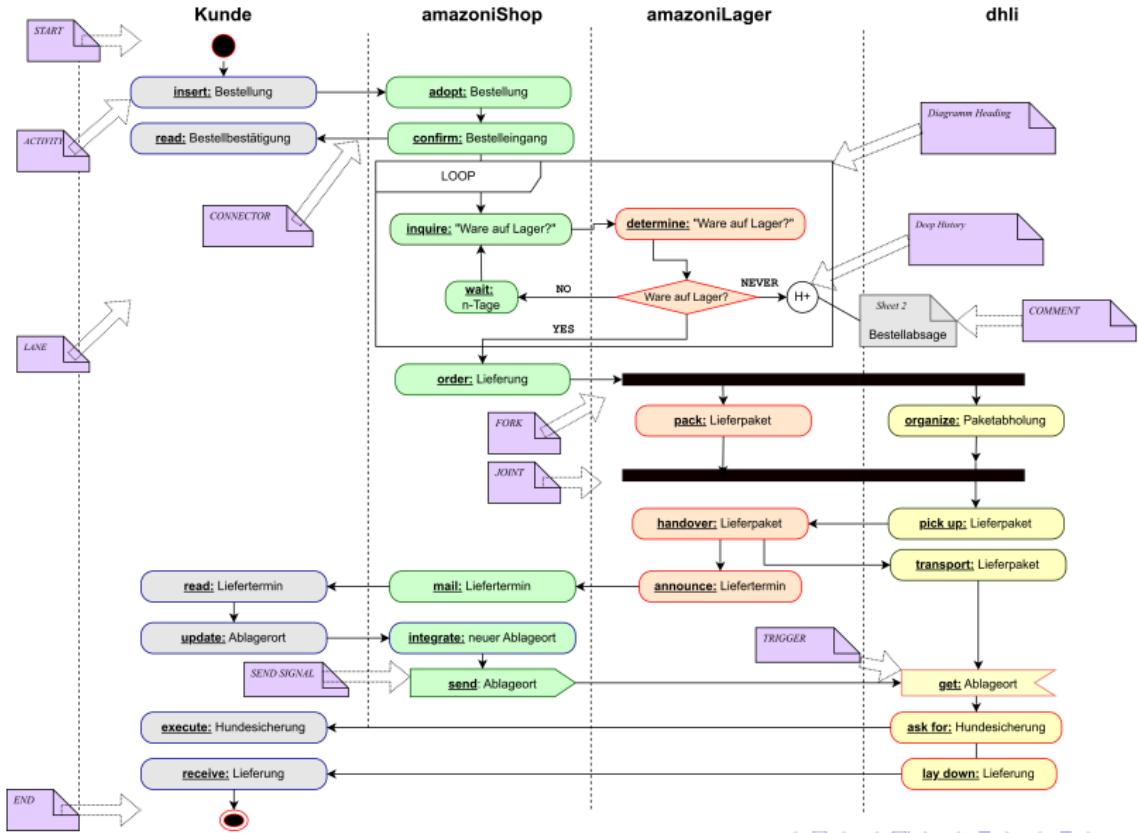


**Deep History:** Verweis auf anderes Dokument

## #UML Activity Notationselemente nach

- <https://de.wikipedia.org/Aktivit%C3%A4tsdiagramm>
- <https://www.lucidchart.com/pages/uml-activity-diagram>
- <https://www.oose.de/wp-content/uploads/2012/05/UML-Notations%C3%BCbersicht-2.5.pdf>

# LFCX::UML:Aktivitätsdiagramm:Beispiel



# LFCX::UML:Sequenzdiagramm:Symbole



Darstellung einer Klasse oder eines Objekts. Es zeigt, wie sich ein Objekt im Kontext des Systems verhalten wird.



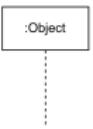
Repräsentiert die Zeit, die ein Objekt zum Abschließen einer Aufgabe benötigt. Je länger die Ausführung der Aufgabe dauert, desto länger der Aktivitätsbalken.



Stellt Entitäten dar, die mit dem System interagieren bzw. systemintern sind.



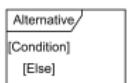
Dient der Gruppierung und Kapselung von Elementen, um die Übersichtlichkeit des Diagramms zu verbessern.



**Lebenslinie:** Zeigt, wann ein Objekt oder Akteur aktiv ist und wann Nachrichten gesendet oder empfangen werden.



**Optionenschleife:** Gruppiert bedingte Abläufe, die nur ausgeführt werden, wenn die angegebene Bedingung wahr ist.



**Alternative:** Gruppiert verschiedene Pfade, von denen nur einer basierend auf den angegebenen Bedingungen ausgeführt wird.



**Synchrone Nachricht:** Zeigt an, dass der Sender eine Anfrage sendet und auf die Antwort des Empfängers wartet.



**Asynchrone Nachricht:** Zeigt an, dass der Sender eine Nachricht sofort weiterarbeitet, ohne auf eine Antwort zu warten.



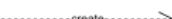
**Synchrone Antwort:** Zeigt an, dass die Antwort auf eine Nachricht synchron und blockierend erfolgt.\*



**Asynchrone Antwort:** Zeigt an, dass die Antwort auf eine Nachricht asynchron und nicht blockierend erfolgt.\*



**Synchrone Nachricht / asynchrone Antwort:** Zeigt an, dass der Sender auf die Antwort des Empfängers wartet, der antwortende Empfänger aber nicht mehr.



**Asynchrone Nachrichtenerstellung:** Zeigt an, dass der Sender ein neues lauschendes Objekt erstellt und selbst nicht blockiert wird. (TCP basierter Server)

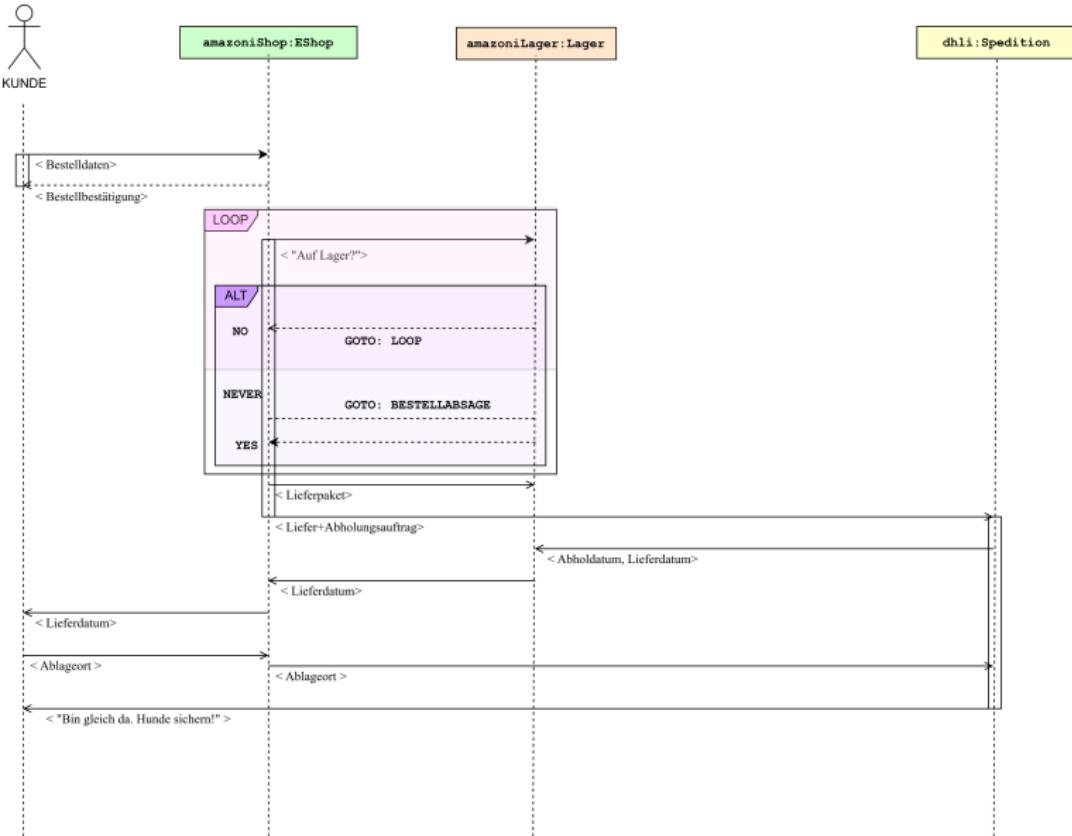


**Löschnachricht:** Zeigt an, dass das Objekt nach dem Empfang der Nachricht gelöscht wird. (TCP basierter Server)

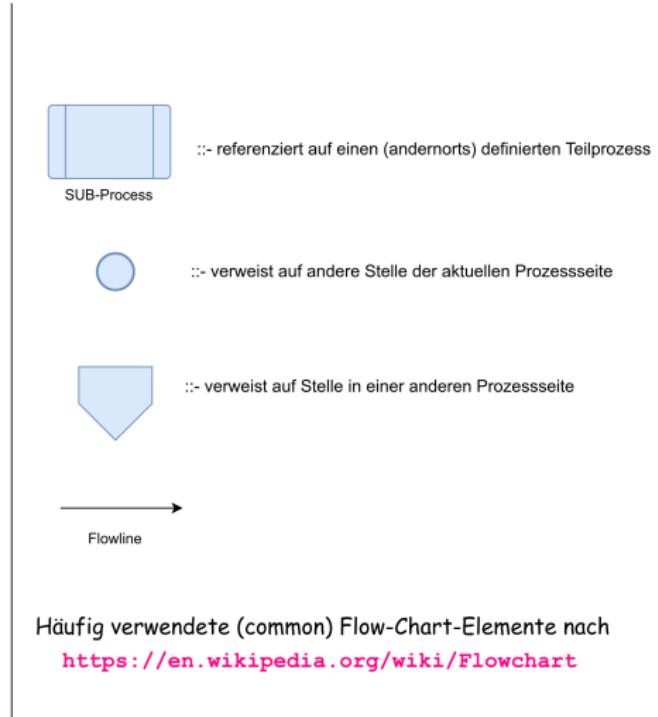
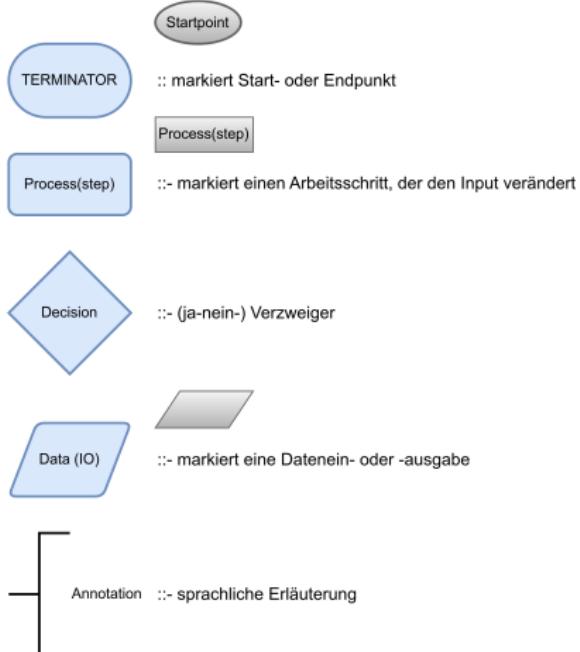
\* Hinweis: Welche Linie-Pfeilspitzenkombination richtig ist, ergibt sich aus dem Protokoll! Das traditionelle Client-Server-Modell hat eine synchrone Nachricht (Client wartet auf Antwort) und eine asynchrone Antwort (Server liefert und bedient den nächsten).

# LFCX::UML:Sequenzdiagramm:Beispiel

#Demo UML Sequence Diagram



# LFCX::Flussdiagramm:Symbole



# LFCX::Flussdiagramm:Erweiterte Symbole



:: markiert Wartezeiten in der Abfolge



:: markiert einen Vorbeitungsschritt



:: markiert manuellen Schritt in semi-automatischem Prozess



:: markiert Datenzusammenführung nebenläufiger Prozesse



:: markiert die Aufsplitzung in nebenläufige Prozesse



:: markiert die logische Konjunktion von Prozessschritten

*entspricht eigentlich einem Split*



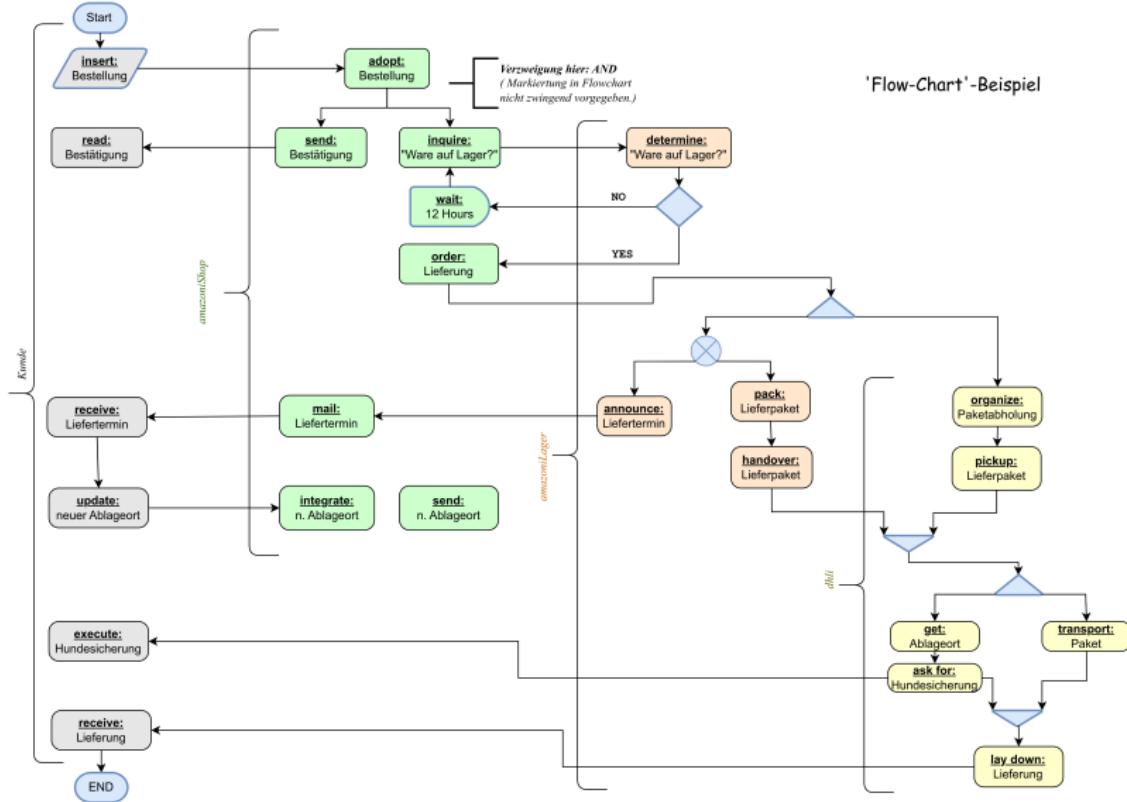
:: markiert die logische Disjunktion von Prozessschritten

*entspricht eigentlich einer Entscheidung*

Häufig Ergänzungen der Flow-Chart-Elemente nach

<https://www.breeztree.com/articles/excel-flowchart-shapes>

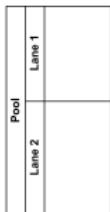
# LFCX::Flussdiagramm: Beispiel



# LFCX::Business Process Model + Notation: Symbole



:- markiert den Arbeitsbereich einer Gruppe  
→ Kersken S. 431 + BPMN-2.0.LPDE, S.27



:- fasst Aufgaben einer ORG-Einheit zusammen  
→ Kersken S. 431 + BPMN-2.0.LPDE, S.27

Ergebnisse (Events), Aufgaben (Tasks) und Verzweigungen (Gateways) werden in Lanes je nach Zuständigkeit angeordnet



:- Startereignis [Beginn des Prozesses] → Kersken S. 431 + BPMN-2.0.LPDE, S.29



:- Messagestart [Prozess durch Nachricht gestartet]



:- Timerstart [Prozess durch Zeitevent gestartet]



:- Zweischenergebnis



→ Kersken S. 431 + BPMN-2.0.LPDE, S.29



:- Gateway\* (exklusiv) [Ein Nachfolger wird ausgeführt]



:- Gateway (parallel) [Alle Nachfolger werden ausgeführt]



:- Gateway\* (inklusive) [einer oder mehrere Nachfolger werden (nebenläufig) ausgeführt]

→ Kersken S. 432 + BPMN-2.0.LPDE, S.32



:- Task [unspezifizierte Standardaufgabe]

→ Kersken S. 432 + BPMN-2.0.LPDE, S.30



:- User-Task [von Person ausgeführte Aufgabe]



:- Script-Task [durch Code spezifizierte Aufgabe]



:- Sub-Task [Aufgabe mit eigenständiger Definition]



:- normal / uncontrolled Flow

→ Kersken S. 433 + BPMN-2.0.LPDE, S.32



:- Conditional Flow: (condition evaluated at runtime)

to determine whether or not the Sequence Flow will be used

→ BPMN-2.0.LPDE, S.31



:- Default Flow: (if no conditional flow is used)

→ BPMN-2.0.LPDE, S.33



:- FORK :-



→ BPMN-2.0.LPDE, S.34



:- MERGE :-



→ BPMN-2.0.LPDE, S.38

#Business Process Model and Notation'-Elemente nach

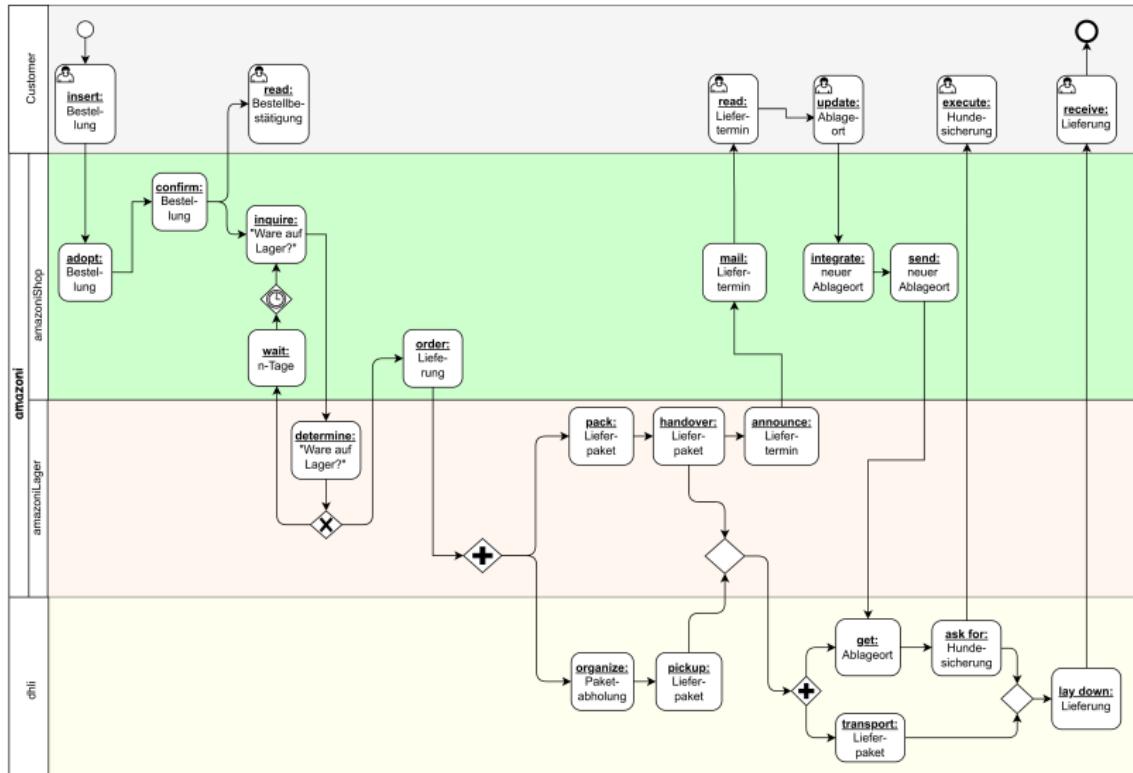
**Kersken: Daten- und Prozessanalyse, S.429ff**

+ Seitenverweis auf BPMN- 2.0.1 Standardisierungsdokument

<https://www.omg.org/spec/BPMN/2.0.1/PDF>

\* Bindungen werden an Pfeile geschrieben

# LFCX::Business Process Model + Notation: Beispiel



'Business Process Model and Notation'-Beispiel

# LFCX::Ereignisgesteuerte Prozess-Kette: Symbole

## EPC/EPK-Elemente:



::- Prozessschritt / Aktion / Aufgabe, die auf Ereignis folgt



::- Zustand, vor oder nach einer Aktion / Funktion



::- Verweis auf andernorts definierten Prozessschritt



::- AND-Konnektor (Nachfolger werden parallel ausgeführt)



::- OR-Konnektor (1-n Nachfolger werden ausgeführt)

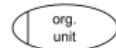


::- xOR-Konnektor (1 aus n Nachfolger werden ausgeführt)

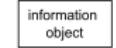


::- Vorgänger/Nachfolger Markant

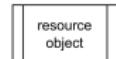
## eEPC/eEPK-Ergänzungen



::- Verantwortlicher Step-Eigner



::- erzeugtes Meldungsobjekt, Datenspeicher



::- verwendete Ressource

----- ::- verlinkt Eigner, Output und Ressourcen mit Funktionen

Regel 1: Jeder Aktion, Aufgabe, Prozesswegweiser geht ein Ereignis / Zustand voraus.

Regel 2: Jeder Aktion, Aufgabe, Prozesswegweiser folgt mindestens ein Ereignis Zustand nach.

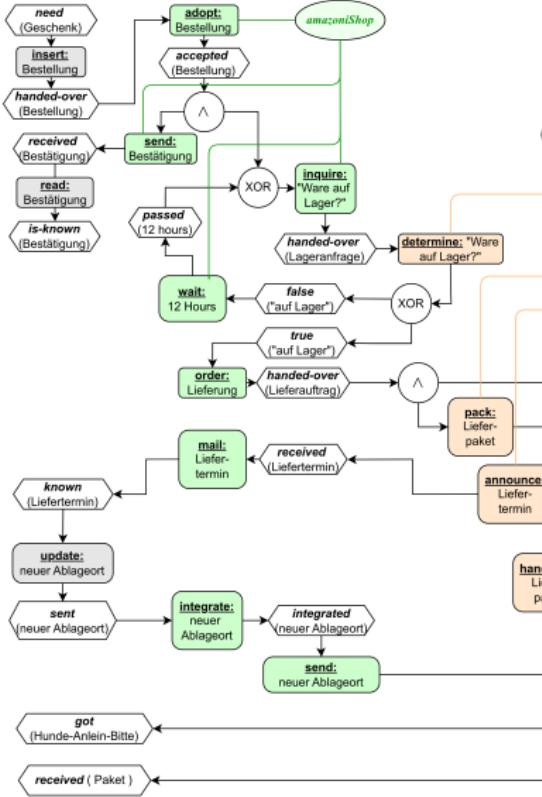
Regel 3: Folgen Aktionen, etc. mehrere Zustände, werden sie per Konnektoren 'aufgespalten'.

Regel 4: Ereignisse und Aktionen haben grundsätzlich nur einen Eingang und einen Ausgang (1 Pfeil rein und 1 Pfeil raus).

## EPC/EPK-Elemente nach

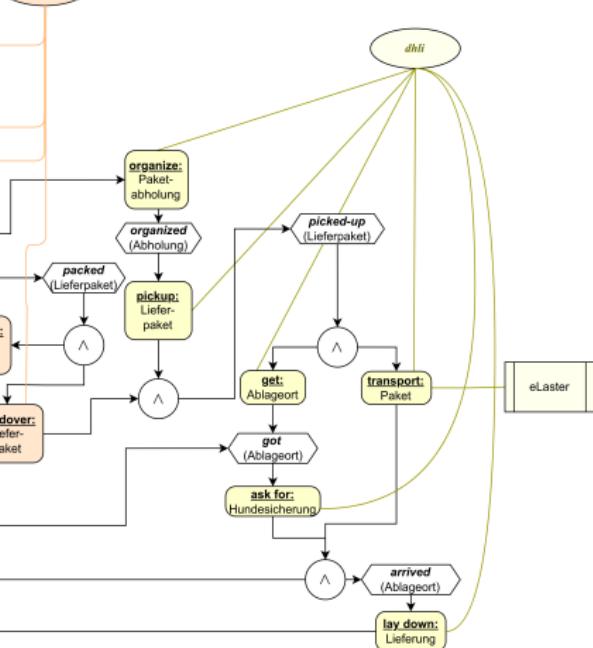
[https://de.wikipedia.org/wiki/Ereignisgesteuerte\\_Prozesskette](https://de.wikipedia.org/wiki/Ereignisgesteuerte_Prozesskette)

# LFCX::Ereignisgesteuerte Prozess-Kette: Beispiel



1. Jeder Aktion, Aufgabe, Prozesswegweiser gibt ein Ereignis / Zustand vor.
2. Jeder Aktion, Aufgabe, Prozesswegweiser folgt mindestens ein Ereignis / Zustand nach.
3. Folgen Aktionen, etc., mehrere Zustände, werden sie per Konsektoval "infolgedessen".
4. Ereignisse und Aktionen haben grundsätzlich nur einen Eingang und einen Ausgang (1 Pfeil rein und 1 Pfeil raus).

'Ereignisgesteuerte-Prozesskette'-Beispiel



# LFCX::Synopse der Prozessdiagrammsymbole

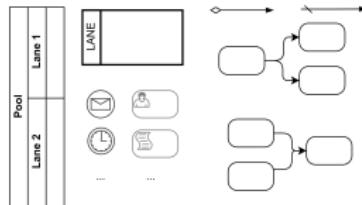
## Symbolsynopse

	<i>Flochart</i>	<i>UML-Activity</i>	<i>BPMN</i>	<i>EPK</i>
<b>Prozess-</b> -START				
-STOP				
-SCHRITT → Aktivität				
-ZUSTAND → Ereignis				
<b>NACHFOLGER</b> → Connector				
<b>XOR-VERZWEIGUNG</b> → Entweder/Oder → Entscheidung				
<b>OR-VERZWEIGUNG</b>				
<b>AND-VERZWEIGUNG</b> → Fork → Aufteilung in scheitende Prozesse → Split				
<b>MERGE</b> → Zusammenführung scheitender Prozesse → Joint				
<b>LANES</b>				
<b>Kommentar</b> → Annotation				

## Abgrenzung der Prozessdiagrammsymbole

*Flochart*      *Data (IO)*

*BPMN*



*EPK*



Herzlichen Dank für Ihre Aufmerksamkeit!



Diese Präsentation gehört zum Open-Source-Projekt *proTironeComputatri*<sup>3</sup>, initiiert v. Karsten Reincke, Hohenahr<sup>4</sup>. Die Unterrichtseinheiten stehen unter den Bedingungen der CC-BY-4.0-Lizenz zur freien Verfügung. Die Quellen dazu finden Sie unter *protico.ltx*<sup>5</sup>.

<sup>3</sup> → <https://github.com/pro-tirone-computatri/>

<sup>4</sup> → <https://github.com/pro-tirone-computatri/protico.ltx/CONTRIBUTORS.md>

<sup>5</sup> → <https://github.com/pro-tirone-computatri/protico.ltx>

# Bildnachweise

-  von Karsten Reincke. Lizenziert unter proTirone-Logo-License. Bereitgestellt auf github. (may only be used as logo for proTirone)
-  von anonymous. Lizenziert unter CC0. Bereitgestellt auf Pxhere:774947.