Sequence diagrams (Sequenzdiagramme)

(BABOK-v3-Technik 10.42)

Allgemeines

Sequenzdiagramme geben Antworten auf die Frage „**Wie läuft die Kommunikation in meinem System ab?**“. Sie ermöglichen die Modellierung von festen Reihenfolgen, zeitlichen und logischen Ablaufbedingungen oder Schleifen etc.

Das Sequenzdiagramm ist das am meisten verwendete Interaktionsdiagramm der UML. „Wie immer“ in der der UML ist auch das Sequenzdiagramm nur eine mögliche Sicht, die einen bestimmten Aspekt der Systembetrachtung herausstellt.

BABOK v3:

10.42.1 **Purpose**

Sequence diagrams are used to model the logic of usage scenarios by showing the information passed between objects in the system through the execution of the scenario.

Interaktionen

Da die Interaktion ein wichtiges Konzept in der UML ist gehen wir zunächst näher darauf ein.

Interaktionen sind das Zusammenspiel von mehreren Kommunikationspartnern. Dies umfasst den Nachrichten- und Datenaustausch zwischen zwei oder mehreren Beteiligten. Beteiligte können dabei Subsysteme, Komponenten und Klassen etc. sein.

Immer dann, wenn zwei oder mehrere Einheiten, die ein eigenes Verhalten realisieren, miteinander kommunizieren, spricht man von Interaktion. Die Grundelemente einer Interaktion sind

- Die Kommunikationspartner (realisiert durch eine Lebenslinie) und
- Die Nachrichten, die von einem Sender zu einem oder mehreren Empfängern gesendet werden

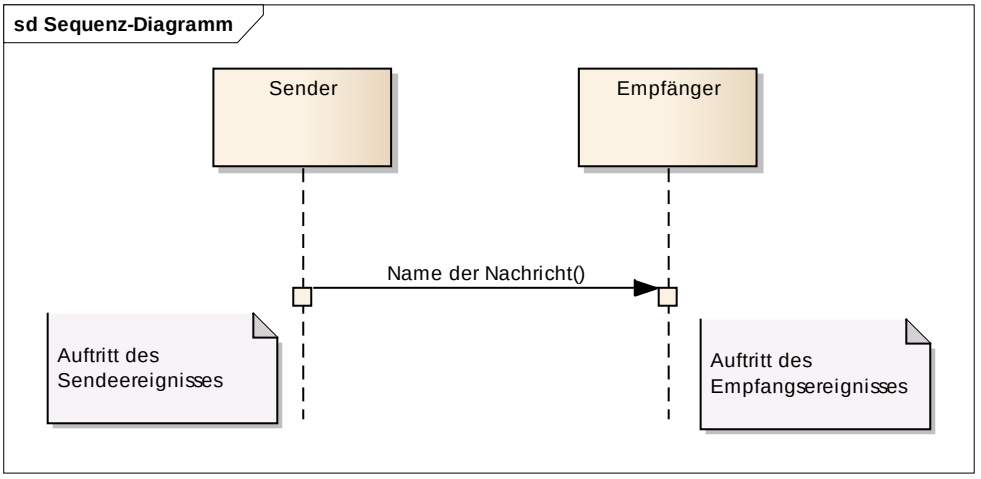
Eine Nachricht kann sein

- Der Aufruf einer Operation (bei einer Klasse)
- Eine Rückantwort als Ergebnis einer Operationsabarbeitung
- Ein Signal (z.B. zur Übertragung eines Zeitereignisses)
- Ein logisches Ereignis (z.B. ein Käufer unterschreibt einen Vertrag)
- Das Setzen einer Variablen mit einem Wert

Der Kommunikation liegt ein spezielles Ereignismodell zugrunde, das den Austausch von Nachrichten in Form von „Auftritts-Spezifikationen“ erklärt (dies ist notwendig, um zwischen einem Ereignis an sich und dem Auftreten eines Ereignisses unterscheiden zu können).

Grundsätzlich werden nicht alle Interaktionen eines Systems dokumentiert, sondern nur typische, wichtige und kritische Sequenzen (es sei denn, die Vollständigkeit wird z.B. zur Code-Generierung benötigt).

Das Grundprinzip einer Interaktion sieht wie folgt aus:



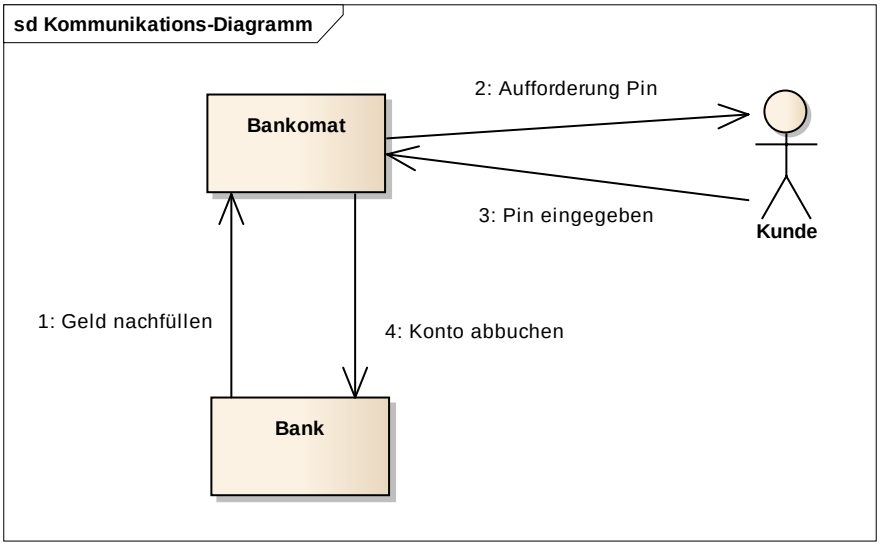
Die Entkopplung zwischen Sende- und Empfangsereignis hat den Vorteil, dass zwischen dem Aussenden und dem Empfang einer Nachricht beliebig viel Zeit liegen kann.

In der Modellierung ist es oft wünschenswert, dass bestimmte Aspekte bewusst ausgearbeitet oder auch bewusst vernachlässigt werden können. Manchmal interessieren nur die beteiligten Kommunikationspartner, ein anderes Mal ist die exakte zeitliche Reihenfolge wichtig, in einer anderen Situation müssen Kommunikationsfehler und deren Behandlung dargestellt werden.

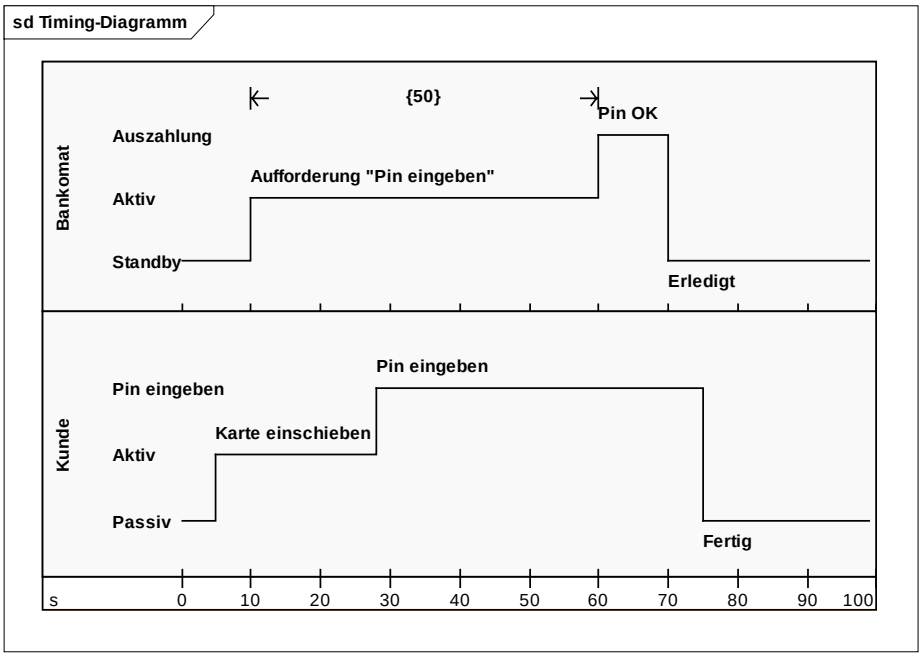
Aus diesem Grund bietet die UML verschiedene Arten von Interaktionsdiagrammen an. Jede Art von Diagrammen hat dabei einen bestimmten Fokus und eine bestimmte Detailtiefe, um den jeweiligen Modellierungsaspekt aufzuzeigen. Zur Darstellung von Interaktionen gibt es folgende Diagramme:

Sequenz-Diagramm	Bietet Antwort auf die Frage „Wer tauscht mit wem welche Informationen in welcher Reihenfolge aus“?	Stellt den zeitlichen Ablauf des Informationsaustausches zwischen Kommunikationspartnern dar
Kommunikations-Diagramm	Bietet Antwort auf die Frage „Wer kommuniziert mit wem“?	Stellt den Informationsaustausch zwischen Kommunikationspartner als Überblick dar
Timing-Diagramm	Bietet Antwort auf die Frage „Wann befindet sich welcher Interaktionspartner in welchem Zustand“?	Stellt das exakte zeitliche Verhalten von Kommunikation dar, wobei es wichtig ist, dass Ereignisse zu einem bestimmten Zeitpunkt auftreten
Interaktions-Übersichtsdiagramm	Bietet Antwort auf die Frage „Wann läuft welche Interaktion ab“?	Verbindet obige Interaktionsdiagramme auf einer Top-Level-Ebene; geeignet zur Strukturierung

Beispiel eines Kommunikations-Diagramms:



Beispiel für ein Timing-Diagramm:



Sequenzdiagramme

Die Verwendung von Sequenzdiagrammen ist dann sinnvoll, wenn eine Interaktion unter folgenden Randbedingungen dargestellt werden soll:

- Die Abfolge der Nachrichten ist wichtig
- Interaktionen sind komplex und müssen gesteuert werden
- Es sollen Ablaufdetails gezeigt werden

BABOK v3:

10.42.2 **Description**

A sequence diagram shows how processes or objects interact during a scenario. The classes required to execute the scenario and the messages they pass to one another (triggered by steps in the use case) are displayed on the diagram. The sequence diagram shows how objects used in the scenario interact, but not how they are related to one another. Sequence diagrams are also often used to show how user interface components or software components interact.

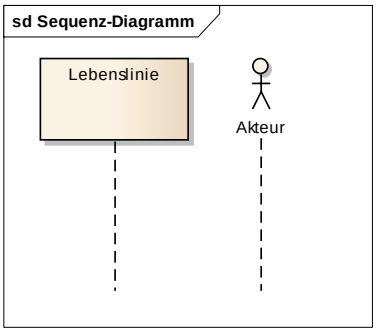
Elemente

Interaktionsrahmen

Ähnlich wie bei anderen Diagrammtypen kann mit einem Interaktionsrahmen eine Abgrenzung vorgenommen werden; insbesondere ist das interessant, wenn eine Interaktion wiederverwendet werden kann - siehe oben „sd Sequenz-Diagramm“.

Lebenslinie

Eine Lebenslinie wird als rechteckiger Kasten mit angeschlossener, gestrichelter Linie dargestellt. Alternativ kann z.B. das Akteurs-Symbol statt des Rechtecks verwendet werden:



Eine Lebenslinie repräsentiert einen Teilnehmer einer Interaktion. Anonyme Lebenslinien sind nicht zulässig, es muss ein Kommunikationspartner angegeben werden.

BABOK v3:

> .1 Lifeline
>
> A lifeline represents the lifespan of an object during the scenario being modelled in a sequence diagram. The example below shows the object order. A lifeline is drawn as a dashed line that vertically descends from each object box to the bottom of the page.

Nachrichten

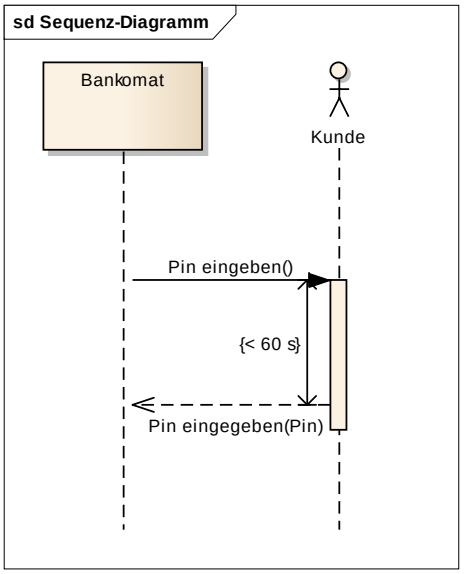
Eine Nachricht repräsentiert den Informationsfluss zwischen den Kommunikationspartnern. Die Nachrichten werden als Pfeile zwischen den Lebenslinien der Interaktionsteilnehmer dargestellt, und zwar immer vom Sender zum Empfänger.

Es wird zwischen synchronen und asynchronen Nachrichten unterschieden.

BABOK v3:

- **Synchronous Call**: transfers control to the receiving object. The sender cannot act until a return message is received.
- **Asynchronous Call**: (also known as a signal) allows the object to continue with its own processing after sending the signal. The object may send many signals simultaneously, but may only accept one signal at a time.

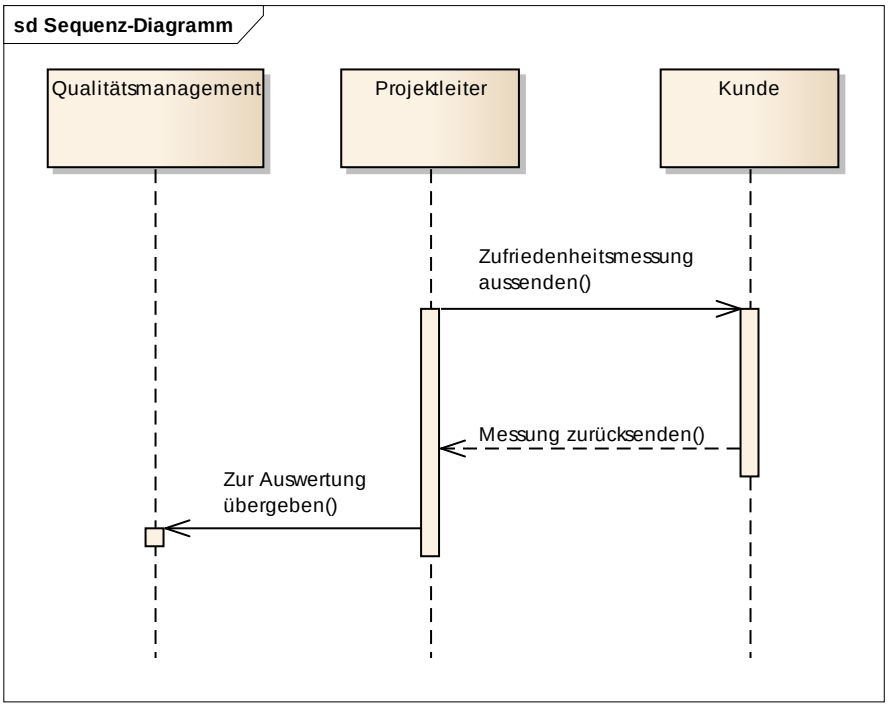
Bei der synchronen Nachricht wartet der Sender nach dem Sendeereignis, bis das initiierte Verhalten beim Empfänger beendet ist und dieser eine Antwortnachricht sendet. Erst dann setzt der Sender seine Abarbeitung fort. In der Antwortnachricht dürfen Antwortdaten mitgeliefert werden:



Synchrone Nachrichten sind durch eine ausgefüllte Pfeilspitze gekennzeichnet.

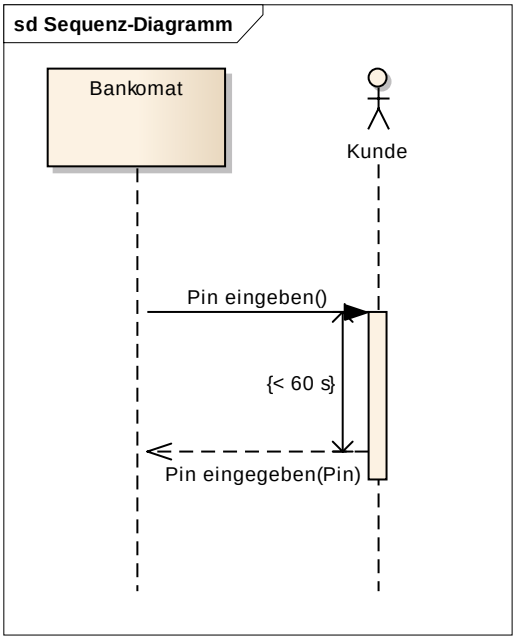
Handelt es sich wie im obigen Bild um eine Antwortnachricht (in diesem Fall um einen Rücksprung auf einen synchronen Operationsaufruf), so wird die Linie gestrichelt dargestellt (Pin eingegeben(Pin)).

Bei der asynchronen Nachricht wartet der Sender nach dem Sendeereignis nicht auf eine Antwort des Emfpängers, sondern setzt unmittelbar nach dem Sendeereignis seine Abarbeitung fort. Ungeachtet dessen kann durchaus (zu einem späteren Zeitpunkt) eine Antwort vom Empfänger kommen, die jedoch wieder als ansynchrone Nachricht interpretiert wird:



Asynchrone Nachrichten sind durch eine offene Pfeilspitze gekennzeichnet.

Nachrichten können Argumente enthalten – siehe Beispiel „Pin eingegeben(Pin)“. Argumente können Konstante, Ein- und Ausgabeparameter etc. sein. Argumente dürfen auch „-„ beinhalten, was keinem konkreten Argumentwert entspricht:



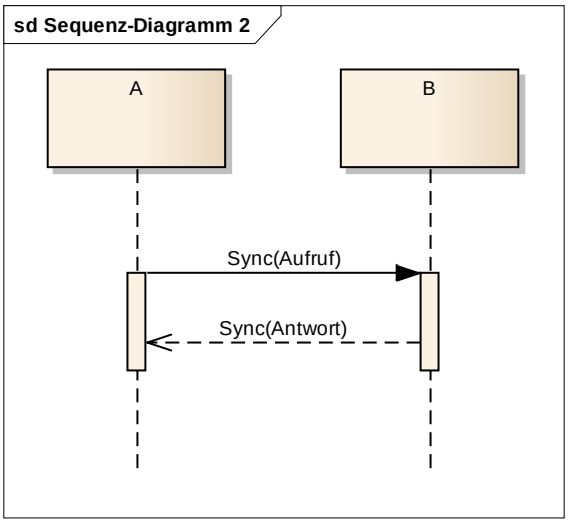
Interessant ist auch die Möglichkeit von Zeitangaben; diese sind jedoch nur dann relevant, wenn sie auch explizit angegeben sind – siehe obiges Beispiel „Pineingabe muss innerhalb von 60 s erfolgen“. Es können Zeitpunkte und Zeitintervalle angegegeben werden.

Hinweise zur Notation

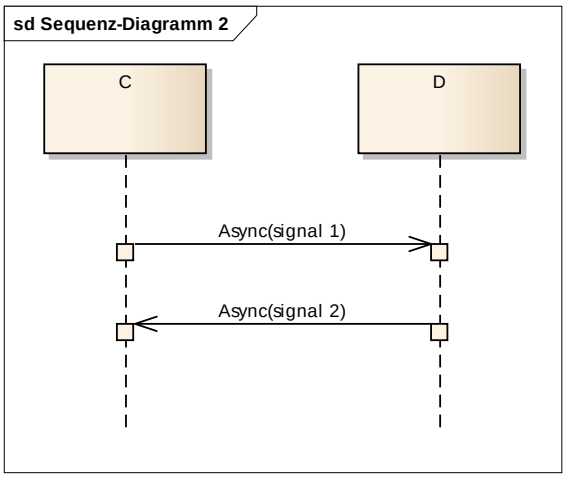
Bei der Notation und beim Lesen von Sequenzdiagrammen ist auf bestimmte Eigenheiten der Darstellung Rücksicht zu nehmen.

Sequenzdiagramme werden „von oben nach unten“ gelesen; insoferne wird durch die Leserichtung eine gewisse implizite Reihenfolge der Ereignisse gebildet: bei synchronen Nachrichten kann das Empfangsereignis erst zwingend nach dem zugehörigen Sendeereignis auftreten. Jedoch hat insbesondere die Darstellung von asynchronen Nachrichten oft mit der tatsächlichen oder vorhergesehenen Reihenfolge nicht unbedingt etwas zu tun.

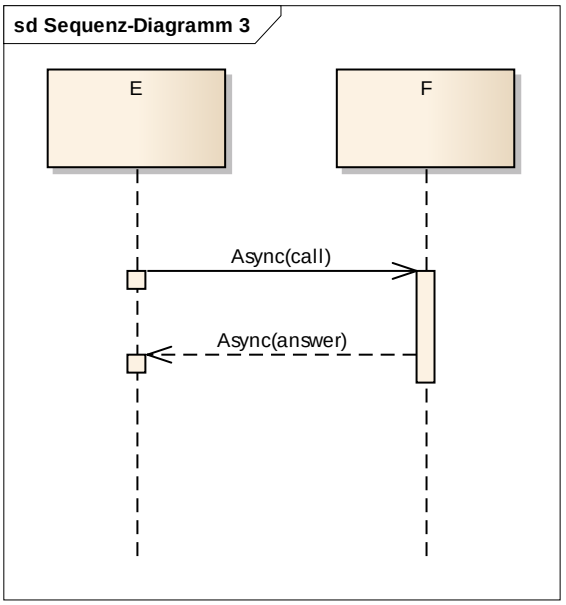
Einige Beispiele sollen dies noch verdeutlichen:



Bei einer synchronen Nachricht wird von A ein Aufruf an B gesendet; das Empfangsereignis bei B kann erst nach dem Sendeereignis von A auftreten. Da der Sender auf eine Antwort wartet, wird dies oben mit einem Rechteck auf der Lebenslinie dargestellt.



Bei einer asynchronen Nachricht wird auf keine Antwort gewartet, Signal 2 kann von D irgendwann nach Erhalt und Verarbeitung von Signal 1 gesendet werden. Da die Nachrichten nicht miteinander in Verbindung stehen, kann Signal 2 auch vor Signal 1 auftreten!



Eine asynchrone Nachricht kann auch vom Typ „call“ sein; mit dem Rechteck auf der Lebenslinie von B wird dies angedeutet, dass darauf hin (wann auch immer) von F->E eine Antwort erwartet wird (zusammengehörige Nachrichten werden sozusagen gruppiert).

Vor- und Nachteile

Vorteile

Mit Sequenzdiagrammen können Abläufe mit Zeitbezug sehr detailliert dargestellt werden. Sie eignen sich daher auch für die Präzisierung von beispielsweise Use-Case-Beschreibungen.

BABOK v3:

10.42.4 Usage Considerations

.1 Strengths

- Shows the interaction between the objects of a system in the chronological order that the interactions occur.
- Shows the interaction between the objects in a visual manner that allows the logic to be validated by stakeholders with relative ease.
- Use cases can be refined into one or more sequence diagrams in order to provide added detail and a more in-depth understanding of a business process.

Nachteile

Sequenzdiagramme können schnell komplex werden und neigen dadurch ohne gute Detailkenntnisse oder entsprechende Vereinbarungen zur Darstellung zu fehlerhaften Interpretationen. Möglicherweise können Abläufe und Interaktionen mit anderen Diagrammtypen (siehe oben) geeigneter dargestellt werden.

BABOK v3:

.2 Limitations

- Time and effort can be wasted creating a complete set of sequence diagrams for each use case of a system, which may not be necessary.
- Have historically been used for modelling system flows and may be considered too technical in other circumstances.

Tasks in BABOK v3

BABOK v3 empfiehlt die Verwendung von Sequenzdiagrammen in dem folgenden Task:

7.1 Specify and Model Requirements

- **Sequence Diagrams**: used to specify and model requirements to show how processes operate and interact with one another, and in what order.

Copyright-Note