Use-Cases

(BABOK-v3-Technik 10.47)

Allgemeines

Use-Cases geben Antworten auf die Frage „**Was soll das geplante System leisten**“? Diese Frage sollte generell zu Beginn jeder Systementwicklung stehen.

BABOK v3:

10.47.1 Purpose

Use cases and scenarios describe how a person or system interacts with the solution being modelled to achieve a goal.

Use-Cases – genauer Use-Case-Diagramme – zeigen das Verhalten eines Systems **aus Sicht der externen Nutzer**, in dem es die Nutzer (genannt „Akteure“), die Use-Cases und deren Beziehungen zueinander darstellt.

Ein Nutzer in diesem Sinne kann dabei eine Person, aber auch ein Fremd- oder Nachbarsystem sein. Use-Cases bilden somit die Reaktion eines Systems auf (externe) Ereignisse seiner Umwelt ab.

BABOK v3:

10.47.2 Description

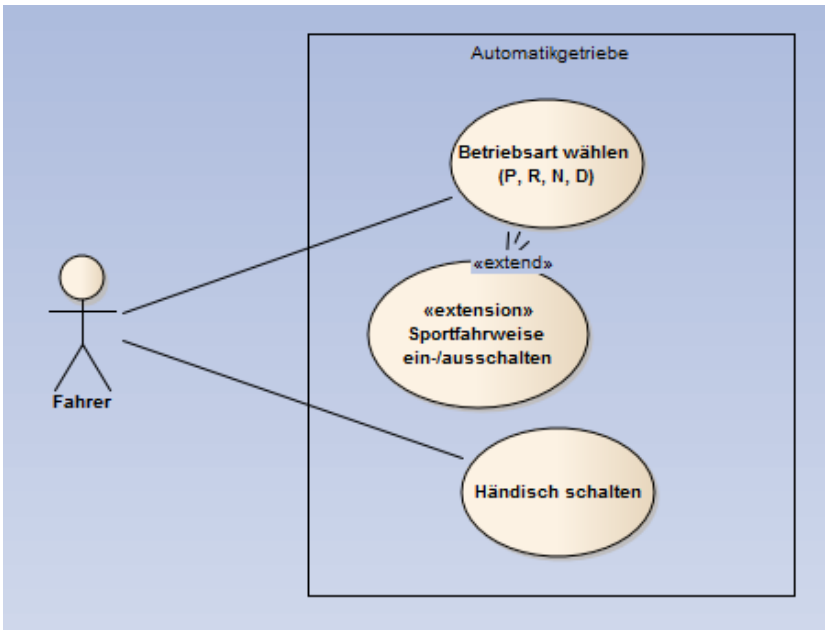
Use cases describe the interactions between the primary actor, the solution, and any secondary actors needed to achieve the primary actor's goal. Use cases are usually triggered by the primary actor, but in some methods may also be triggered by another system or by an external event or timer.

Gelegentlich wird zwischen Business-Use-Cases (welche das Systemverhalten bezüglich Prozessen zeigen) und System-Use-Cases (welche das Verhalten eines realen Systems, z.B. eine Software-Applikation, zeigen) unterschieden.

BABOK v3:

Some use case approaches distinguish between business use cases and system use cases, with business use cases describing how actors interact with a particular process or business function, and system use cases describing the interaction between an actor and a software application.

Ein einfaches Beispiel für ein Automatik-Getriebe eines Personenkraftwagens soll die Darstellung und Verwendung von Use-Cases verdeutlichen:



Der Akteur (in diesem Fall der Fahrer) hat zunächst die Möglichkeit, die Betriebsart zu wählen (P für Parken, R für Rückwärtsgang, N für Leerlauf/Neutral und D für Drive/Fahren). Zusätzlich kann eine Sportabstimmung des Getriebes eingestellt werden. Ferner kann das Getriebe auch händisch bedient werden.

Wie aus obiger Darstellung ersichtlich ist, geben Use-Cases nur ein grobes Bild des Verhaltens wieder. Das Bild sagt beispielsweise nichts direkt darüber aus, dass die Sportabstimmung nur in der „D“-Stellung des Schalthebels aktiviert werden kann oder wie genau das händische Schalten vor sich geht.

Use-Cases brauchen daher zur näheren Definition eine textuelle Beschreibung und/oder ergänzende Diagramme und können zusätzliche Angaben, Voraussetzungen und Ergebnisse zum Ablauf enthalten.

BABOK v3:

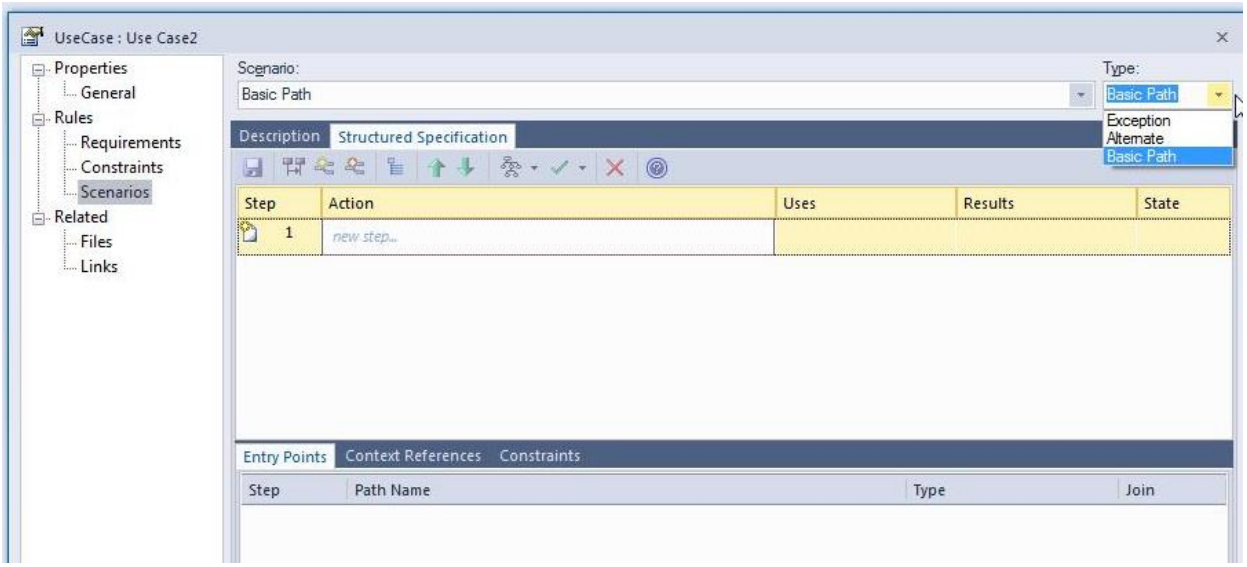
> A use case describes the possible outcomes of an attempt to accomplish a particular goal that the solution will support. It details different paths that can be followed by defining primary and alternative flows. The primary or basic flow represents the most direct way to accomplish the goal of the use case. Special circumstances and exceptions that result in a failure to complete the goal of the use case are documented in alternative or exception flows. Use cases are written from the point of view of the actor and avoid describing the internal workings of the solution.

Mittels sogenannter Szenarien werden bestimmte Abläufe schrittweise beschrieben. Für eine vollständige Ablaufbeschreibung sind meist mehrere Szenarien erforderlich. Ebenso können dabei Systemfunktionen mit alternativem Verhalten oder für Ausnahmesituationen definiert werden.

BABOK v3:

> A scenario describes just one way that an actor can accomplish a particular goal. Scenarios are written as a series of steps performed by actors or by the solution that enable an actor to achieve a goal. A use case describes several scenarios.

In der Modellierungs-Software Enterprise Architect bildet sich dies beispielsweise wie folgt ab:



Darstellung

Use-Case-Diagramme enthalten die grafische Darstellung

- des Systems
- der Use-Cases (Anwendungsfälle)
- der Akteure außerhalb des Systems
- der Beziehungen zwischen Akteuren und Use-Cases oder der Beziehungen zwischen Use-Cases untereinander

BABOK v3:

> Use case diagrams are a graphical representation of the relationships between actors and one or more use cases supported by the solution.

Ein Use-Case ist somit ein in sich abgeschlossener Vorgang, der von außen ausgelöst wird und ein Ergebnis liefert. Akteure initiieren Use-Cases und die Ergebnisse werden an den auslösenden Akteur zurückgeliefert oder an einen anderen Akteur geliefert. Zur Laufzeit agiert eine Instanz eines Akteurs mit einer Instanz eines Use-Cases.

Use-Case-Diagramme gehören zur Gruppe der UML-Verhaltensdiagramme. Use-Cases und Szenarien können in Folge z.B. mit Aktivitätsdiagrammen verfeinert werden, womit die (internen) Abläufe eines Use-Cases abgebildet werden (im Gegensatz dazu ist ein Use-Case eine abgeschlossene Einheit, deren interne Strukturen auf dieser Darstellungsebene irrelevant sind).

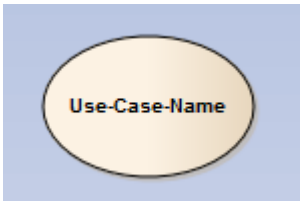
BABOK v3:

> .1 Use Case Diagram
>
> A use case diagram visually depicts the scope of the solution, by showing the actors who interact with the solution, which use cases they interact with, and any relationships between the use cases. Unified Modelling Language™ (UML®) describes the standard notation for a use case diagram.

Notation

Use-Cases

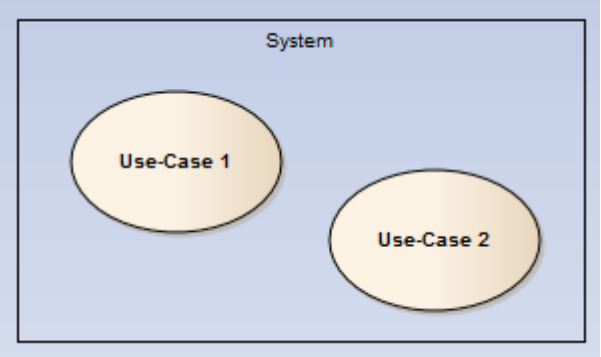
Use-Cases werden üblicherweise als Ellipse dargestellt. Der Use-Case muss einen Namen haben, welcher **die Sicht des auslösenden Akteurs** wiedergibt (und nicht die Sicht des Systems).



BABOK v3:

> Name
>
> The use case has a unique name. The name generally includes a verb that describes the action taken by the actor and a noun that describes either what is being done or the target of the action.

Da Use-Cases das Verhalten eines Systems „nach außen hin“ abbilden, können (sollen) die Use-Cases auch innerhalb einer Systemgrenze liegend dargestellt werden:



Auch wenn eine Kurzbeschreibung alleine zur vollständigen Definition eines Use-Case nicht ausreichend ist, soll zumindest eine kurze Zusammenfassung der Use-Case-Funktion vorhanden sein.

BABOK v3:

> Goal
>
> The goal is a brief description of a successful outcome of the use case from the perspective of the primary actor. This acts as a summary of the use case.

Akteure

Die Akteure werden typisch als Strichmännchen dargestellt; es ist aber auch die Verwendung anderer Symbole möglich (beispielsweise ist ein Drucker besser als Symbol abzubilden):

BABOK v3:

> **Actors**
>
> An actor is any person or system external to the solution that interacts with that solution. Each actor is given a unique name that represents the role they play in interactions with the solution. Some use case authoring approaches recommend against the use of systems or events as actors.
>
> A use case is started by an actor, referred to as the primary actor for that use case. Other actors who participate in the use case in a supporting role are called secondary actors.

Ein Akteur kann auch ein periodisch auftretendes Ereignis sein (z.B. tägliche Sicherung, Monatsabschluss der Buchhaltung). Solche Zeitereignisse werden meist mit einem Timer-/Uhren-Symbol dargestellt.

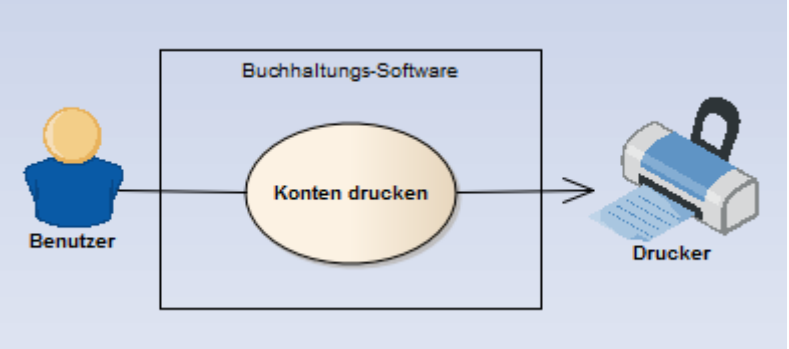
Falls es hilfreich ist, können auch die Ereignisse, welche den Akteur zur Interaktion mit dem Use-Case veranlassen, als sogenanne Trigger angegeben werden.

BABOK v3:

> **Trigger**
>
> A trigger is an event that initiates the flow of events for a use case. The most common trigger is an action taken by the primary actor.
>
> A temporal event (for example, time) can initiate a use case. This is commonly used to trigger a use case that must be executed based on the time of day or a specific calendar date, such as an end-of-day routine or an end-of-month reconciliation of a system.

Assoziationen

Die Akteure werden mit den Use-Cases über Assoziationen verbunden. Diese können bidirektional oder gerichtet sein:



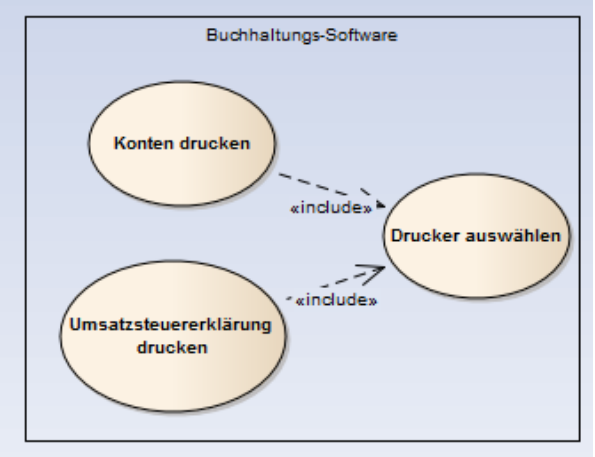
Obiges Beispiel deutet an, dass der Benutzer die Systemfunktion „Konten drucken“ ausführt und durch die Bidirektionalität auch eine Rückmeldung bekommt (z.B. wieviele Seiten gedruckt werden). Hingegen kommen vom Drucker keine Rückmeldungen zu diesem Vorgang.

BABOK v3:

> **Relationships**
>
> Relationships between actors and use cases are called associations. An association line indicates that an actor has access to the functionality represented by the use case. Associations do not represent input, output, time, or dependency.

Include-Beziehung

Wenn mehrere Ablaufschritte großteils identisch sind, können Use-Cases miteinander verknüpft werden. Als Regel gilt, dass Include-Beziehungen dann sinnvoll sind, wenn etwa mehr als 2/3 der Funktionalität identisch oder nahezu identisch ist.



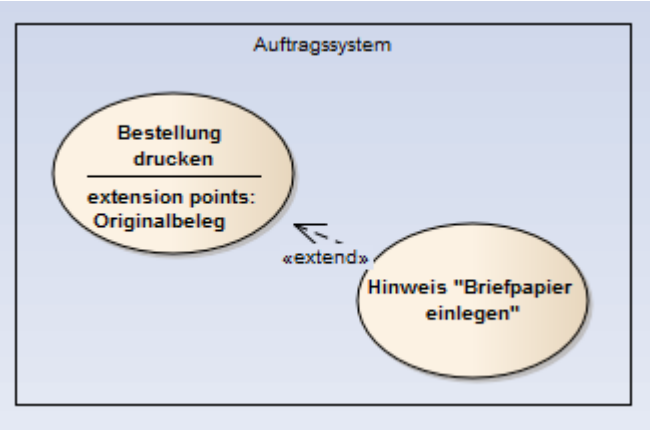
Eine `<<include>>`-Beziehung gibt dabei an, dass ein Use-Case das Verhalten eines anderen Use-Cases importiert, also die Ausführung des aufgerufenen (inkludierten) Use-Cases für den Ablauf des aufrufenden Use-Cases nicht optional, sprich **zwingend notwendig**, ist. Include-Beziehungen werden immer dann dargestellt, wenn ein Ablauf von mehreren Use-Cases verwendet werden kann:

BABOK v3:

> - **Include**: allows for the use case to make use of functionality present in another use case. The included use case does not need to be a complete use case in its own right if it is not directly triggered by an actor. This relationship is most often used either when some shared functionality is required by several use cases or to abstract out a complex piece of logic.

Extend-Beziehung

Eine <<extend>>-Beziehung gibt hingegen an, dass ein Use-Case das Verhalten eines anderen Use-Cases erweitern kann, aber nicht muß. Unter anderem kann dies über eine Bedingung entschieden werden, also die Ausführung des aufgerufenen (erweiterten) Use-Cases nur dann erfolgt, wenn die Bedingung wahr oder falsch ist. Eine Extend-Beziehung zeigt somit an, dass es neben dem Normalverhalten auch auslagerbare Sonderfälle gibt:



Wenn Bestellungen als Originalbelege gedruckt werden sollen, soll zusätzlich der Hinweis „Briefpapier einlegen“ erscheinen. Werden hingegen nur Kopien gedruckt, so kann normales Druckerpapier verwendet werden. Der `extension point` gibt die Bedingung dazu an.

BABOK v3:

> • **Extend**: allows for the insertion of additional behavior into a use case. The use case that is being extended must be completely functional in its own right and must not depend on the extending use case for its successful execution. This relationship may be used to show that an alternate flow has been added to an existing use case (representing new requirements).

Constraints

Eine Randbedingung (Constraint) ist ein boolescher Ausdruck, der allgemein zur Präzisierung eines Modellelements dient. Bei Use-Cases werden constraints typisch als

- pre-conditions und als
- post-conditions

verwendet und dienen zur näheren Beschreibung von Use-Cases, wenn dies erforderlich ist.

Pre-Conditions sind Bedingungen, die **vor** Ausführung des Use-Case erfüllt sein müssen (= Bedingung = true):

BABOK v3:

> **Preconditions**
>
> A precondition is any fact that must be true before the use case can begin. The precondition is not tested in the use case but acts as a constraint on its execution.

Post-Conditions sind Bedingungen, die **nach** Ausführung des Use-Case erfüllt sein müssen (= Bedingung = true):

BABOK v3:

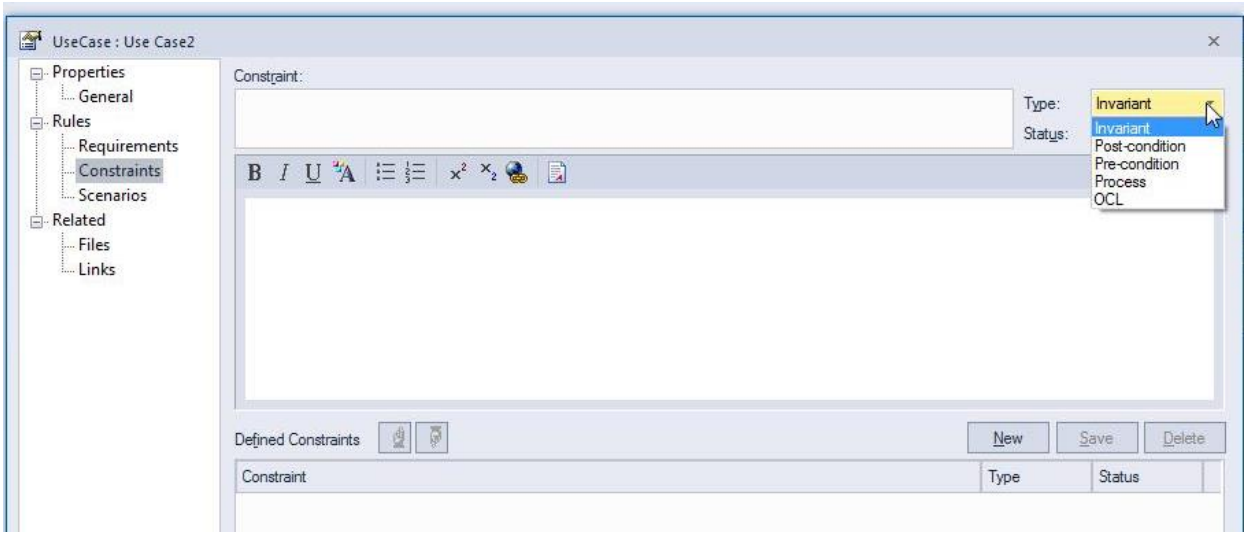
> **Post-conditions or Guarantees**
>
> A post-condition is any fact that must be true when the use case is complete. The post-conditions must be true for all possible flows through the use case, including both the primary and alternative flows. The use case may describe separate post-conditions that are true for successful and unsuccessful executions of the use case. These can be called guarantees; the success guarantee describes the post-conditions for success. Minimal guarantees describe the conditions that are required to be true, even if the actor's goal is not achieved, and may address concerns such as security requirements or data integrity.

Einige Tools erlauben auch die Angabe zusätzlicher Constraints, z.B. für „Process", d.h. eine ergänzende Beschreibung innerhalb des Prozesses (beispielsweise ob es speziell Ausnahme- oder Verzweigungsbedingungen gibt):

BABOK v3:

> **Flow of Events**
>
> The flow of events is the set of steps performed by the actor and the solution during the execution of the use case. Most use case descriptions separate out a basic, primary, or main success flow that represents the shortest or simplest successful path that accomplishes the goal of the actor.
>
> Use cases may also include alternative and exception flows. Alternative flows describe other paths that may be followed to allow the actor to successfully achieve the goal of the use case. Exception flows describe the desired response by the solution when the goal is unachievable and the use case cannot be successfully completed.

In der Modellierungs-Software Enterprise Architect bildet sich dies beispielsweise wie folgt ab:



Beschreibung

Auf Grund der nur sehr groben Darstellung einer Aktion innnerhalb eines Systems benötigen Use-Cases eine nähere Beschreibung. Dies kann erfolgen durch

- eine textuelle Beschreibung, ein oder mehrere Szenarios u.dgl. (nützlich für kurze und klar Abläufe mit wenigen Sonderfällen)
- ein oder mehrere Aktivitätsdiagramme (sinnvoll für ablauf- oder schrittorientierte Use-Cases)
- Kommunikationsdiagramme (einfache, datenorientierte Abläufe)
- Sequenzdiagramme (komplexe, datenorientierte Abläufe)
- Zustandsautomaten (kein „typischer“ Ablauf, sondern Zustände und deren Übergänge)
- Interaktionsübersichtsdiagramme (zur Bündelung mehrerer Use-Cases)

Beispiel BABOK v3 (Technik 10.42 Sequence Diagrams, Usage Considerations - Auszug):

- 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.

Generell gilt: es darf bei der UML niemals grundsätzlich von einer vollständigen Beschreibung aller möglichen Fälle ausgegangen werden. Wird beispielsweise ein Sequenzdiagramm eingesetzt, so stellt dies im allgemeinen nur einen bestimmten Ablauf und nicht alle möglichen Abläufe dar. Die Regel lautet „es wird nur das modelliert, was dargestellt werden soll“.

Vor– und Nachteile

Vorteile

Use-Cases eignen sich auf Grund deren Einfachheit und der hohen Abstraktionsebene hervorragend

- Zur Diskussion und Festlegung von Abläufen mit Stakeholdern
- Zur (vollständigen) Übersicht über Abläufe innerhalb eines Systems
- Zur Übersicht über die beteiligten Akteure (User, Fremdsysteme, ...)

BABOK v3:

10.47.4 Usage Considerations

.1 Strengths

- Use case diagrams can clarify scope and provide a high-level understanding of requirements.
- Use case descriptions are easily understood by stakeholders due to their narrative flow.
- The inclusion of a desired goal or outcome ensures that the business value of the use case is articulated.
- Use case descriptions articulate the functional behaviour of a system.

Nachteile

Wenn Use-Cases „richtig“ eingesetzt werden und die zugehörigen Beschreibungen ein grobes Bild eines Systems geben, so gibt es keine unmittelbaren Nachteile.

BABOK v3:

.2 Limitations

- The flexibility of the use case description format may lead to information being embedded that would be better captured using other techniques such as user interface interactions, non-functional requirements, and business rules.
- Decisions and the business rules that define them should not be recorded directly in use cases, but managed separately and linked from the appropriate step.
- The flexible format of use cases may result in capturing inappropriate or unnecessary detail in the attempt to show every step or interaction.
- Use cases intentionally do not relate to the design of the solution and as a result, significant effort may be required in development to map use case steps to software architecture.

Sonstiges

Kontext

Use-Cases sollen und können in enger Abstimmung mit Kontextdiagrammen erstellt und gesehen werden. Gerade ein Kontextdiagramm liefert meist viele Ideen, welche Use-Cases in einem System ablaufen werden, da hierbei die Fremdsysteme ersichtlich sind.

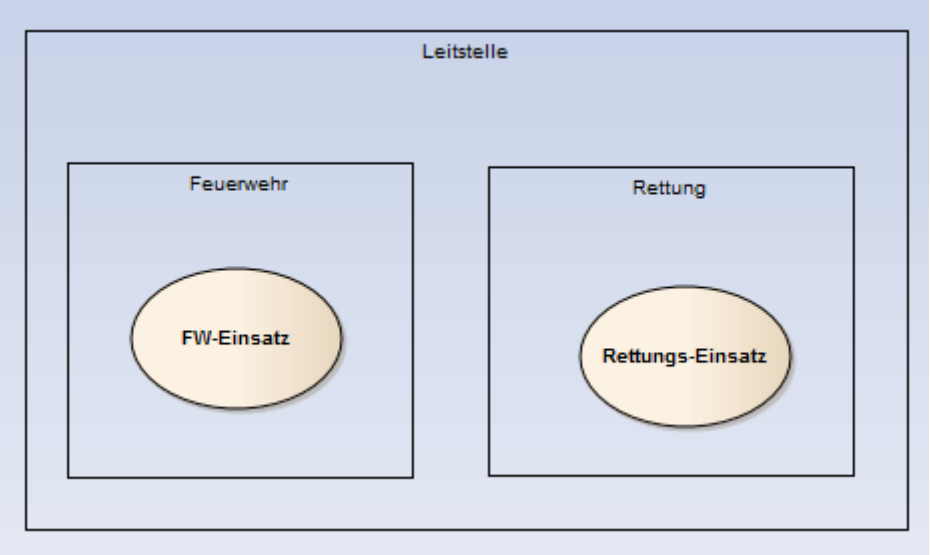
BABOK v3 (Technik 10.41 Scope Modeling, Description - Auszug):

- **Both**: the model identifies a boundary as seen from both sides, as well as elements on both sides of the boundary (for example, venn diagram or use case model).

BABOK v3 (Technik 10.41 Scope Modeling, Elements - Auszug):

- **Function-Responsibility**: relates a function with the agent (stakeholder, organizational unit, or solution component) that is responsible for its execution. Relationships of this type appear on business process models and on collaboration, sequence, and use case diagrams.

Sind sehr viele Use-Cases in einem System vorhanden, so können diese auch auf mehrere (Sub-)Systeme aufgeteilt werden:

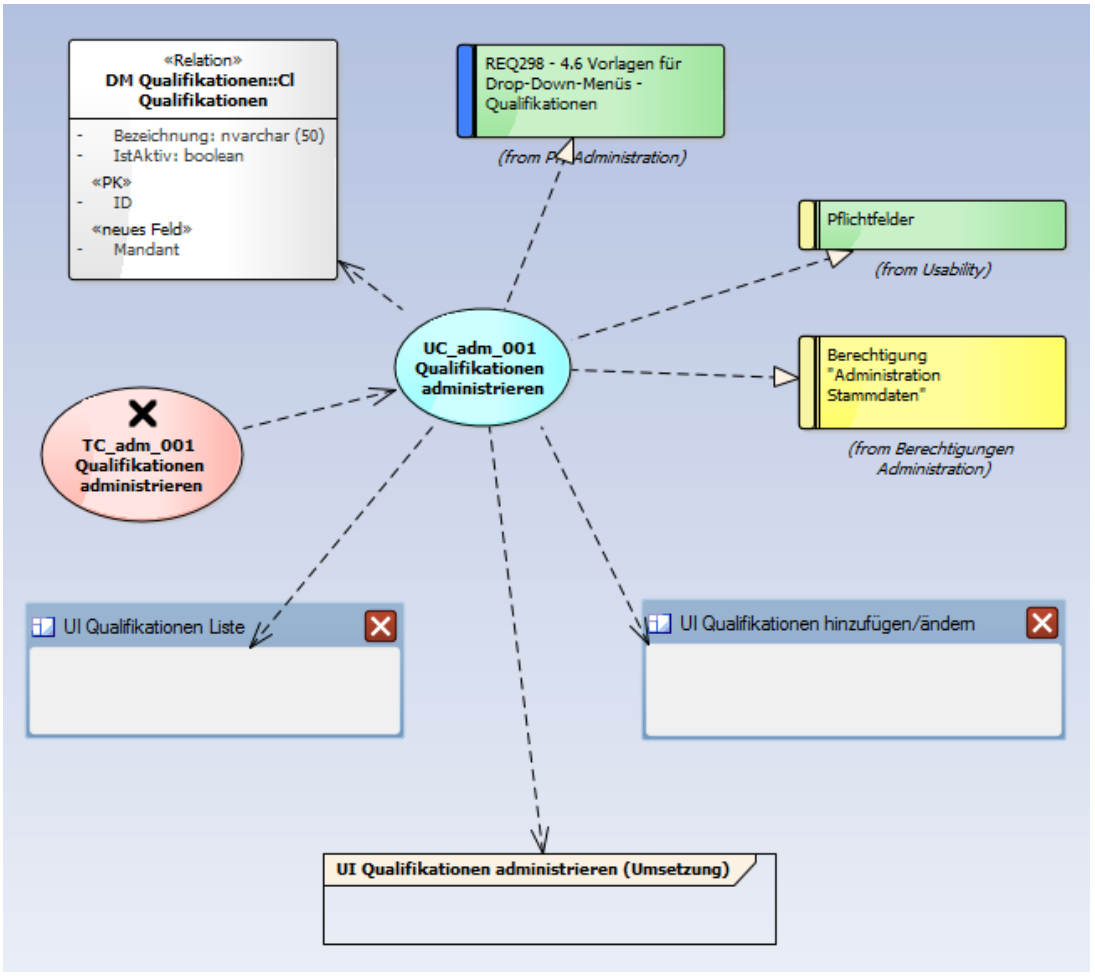


Implementation-Diagramm

Der Use-Case stellt zunächst selbst nur dar, welche (Teil-)Funktionalität im System damit abgebildet und realisiert wird.

Mit einem Implementation-Diagramm können die dazugehörigen Komponenten (typisch sind das z.B. Klassen für Datenmodelle, Anforderungen, Testfälle sowie GUI-Entwürfe) dargestellt werden. Dies erlaubt eine tiefere Sicht auf die Zusammenhänge für den beschriebenen Anwendungsfall.

Anmerkung: ein Implementation-Diagramm ist kein UML-Standard-Diagramm-Typ!



Use-Cases können auch bei der Funktionszerlegung (Functional Decomposition, BABOK v3-Technik 10.22), in der Prozess-Modellierung (Process Modelling, BABOK v3-Technik 10.35) und in der Rollen- und Berechtigungsmatrix (Roles and Permissions Matrix, BABOK v3-Technik 10.39) eine Rolle spielen.

Tasks in BABOK v3

BABOK v3 empfiehlt die Verwendung von Use-Cases in den folgenden Tasks:

5.2 Maintain Requirements

> • **Use Cases and Scenarios**: used to identify a solution component that may be utilized by more than one solution.

7.1 Specify and Model Requirements

> • **Activity Flow**: models represent a sequence of actions, events, or a course that may be taken. Techniques used to represent activity flows include Process Modelling, Use Cases and Scenarios, and User Stories.

sowie

> • **Use Cases and Scenarios**: used to model the desired behaviour of a solution, by showing user interactions with the solution, to achieve a specific goal or accomplish a particular task.

8.1 Measure Solution Performance

> • **Use Cases and Scenarios**: used to define the expected outcomes of a solution.

Copyright-Note