# Scope Modelling

(BABOK-v3-Technik 10.41)

## Allgemeines

Scope-Modelling gibt Antworten auf die Fragen „Was gehört zum System und was nicht“? sowie „Woher kommen die Anforderungen“? Diese Fragen sollten generell zu Beginn jeder Systementwicklung stehen und stehen in engem Zusammenhang mit Use-Cases.

*BABOK v3:*

**10.41.1 Purpose**

Scope models define the nature of one or more limits or boundaries and place elements inside or outside those boundaries.

Scope-Models – hier hauptsächlich im Sinne von **Kontextdiagrammen** behandelt - zeigen, was Liefer- und Funktionsumfang eines Systems ist und welche Nutzer (Personen, Fremdsysteme) mit dem System interagieren, dienen also zur klaren Abgrenzung = Festlegung von Systemgrenzen und auch zur Definition von Schnittstellen.

*BABOK v3:*

**10.41.2 Description**

Scope models are commonly used to describe the boundaries of control, change, a solution, or a need. They may also be used to delimit any simple boundary (as distinct from horizons, emergent properties, and recursive systems).

Die Abgrenzung eines Systems ist auch im Requirements Engineering nach IREB-Standard wichtig, weil

*IREB Foundation Level („Basiswissen Requirements Engineering K. Pohl, C. Rupp“):*

*[…] „Der Systemkontext ist der Teil der Umgebung eines Systems, der für die Definition und das Verständnis der Anforderungen des betrachteten Systems relevant ist“ sowie*

*„Die Systemgrenze separiert das geplante System von seiner Umgebung. Sie grenzt den im Rahmen des Entwicklungsprozesses gestaltbaren und veränderbaren Teil der Realität von Aspekten in der Umgebung ab, die durch den Entwicklungsprozess nicht verändert werden können“. […]*

Es geht somit um die Erfassung und Darstellung von Grenzen. Der BABOK v3 unterscheidet Scope models nach In-scope, Out-of-scope und In- und Out-of-scope-Darstellungen:

*BABOK v3:*

> These models may show elements that include:
>
> - **In-scope**: the model identifies a boundary as seen from inside, as well as the elements contained by that boundary (for example, functional decomposition).
> - **Out-of-scope**: the model identifies a boundary as seen from outside, as well as the elements that are not contained by that boundary (for example, context diagram).
> - **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).

## Modellierungstechniken

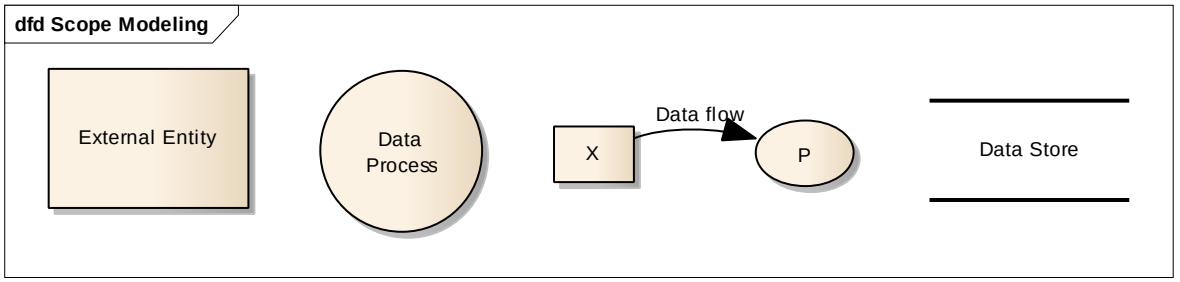
Als Modellierungstechniken für Kontextdiagramme bieten sich unter anderem (eingeschränkte) Use-Case-Diagramme oder Datenflußdiagramme, eventuell auch Komponentendiagramme, an.

*BABOK v3:*

> Scope models are typically represented as a combination of diagrams, matrices, and textual explanations. If the scope is implemented in phases or iterations, the scope model should be described per each phase or iteration.

### Datenflußdiagramme

Der Ursprung der Datenflußdiagramme kommt von Tom De Marco (1978). Datenflußdiagramme kennen lediglich vier Elemente zur Modellierung:




Datenflußdiagramme zeigen wie Daten/Informationen in das System gespeist werden, wie diese verarbeitet und gespeichert werden sowie welche Daten nach außen gegeben werden.

Sie dienen also der Darstellung

- wie Informationen und Daten in das System gebracht und dort verarbeitet werden
- welche externe Einheiten Daten an das System empfangen oder von diesem empfangen
- der datenverarbeitenden Prozesse
- welche Datenspeicher verwendet werden und welche Datenflüsse stattfinden

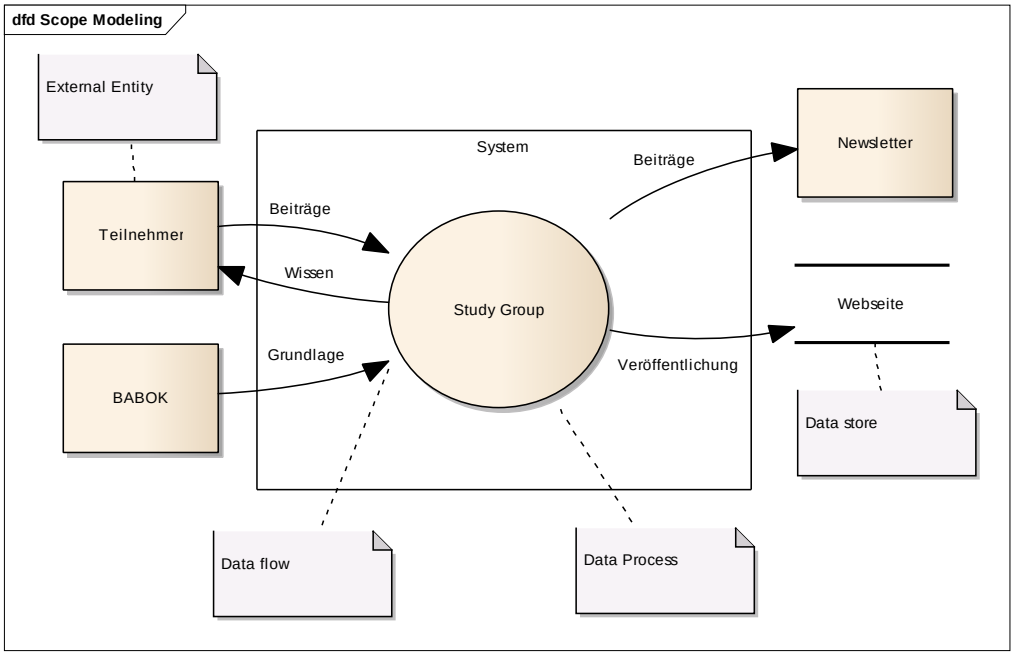
*IREB Foundation Level („Basiswissen Requirements Engineering K. Pohl, C. Rupp“):*

*[…] „Zur Identifikation der Schnittstellen des Systems mit der Umgebung können u.a. die Quellen und Senken des geplanten Systems betrachtet werden. Quellen liefern Eingaben für das System, Senken erhalten Ausgaben vom System“. […]*

Datenflußdiagramme sind als Top-Level-Diagramme geeignet, weil sie nur einen Datenprozess sowie die externen Einheiten und Datenspeicher beschreiben, also die Datenflüsse in das und aus dem System darstellen.

Vorteile sind eine übersichtliche Prozess- und Datendarstellung und die Möglichkeit der Verifikation der Funktionalitäten sowie, dass es sich um ein einfaches Diagramm handelt. Nachteilig dabei ist, dass Verantwortungen nicht ersichtlich und alternative Wege nicht darstellbar sind.

Ein Beispiel einer BABOK-Study Group soll die Verwendung von Datenflußdiagrammen verdeutlichen:




Die Teilnehmer (externe Einheit) liefern Diskussionsbeiträge (Input Data) und erhalten in der Diskussion durch den Data Process „Study Group“ Wissen retour (Output Data). Als Input dient zusätzlich der BABOK als Diskussionsgrundlage.

Der Data Process der Study Group stellt die Diskussionsbeiträge auf die Webseite (Data store) und veröffentlicht (ausgewählte) Themen im Newsletter (somit ebenso ein Output).

*BABOK v3:*

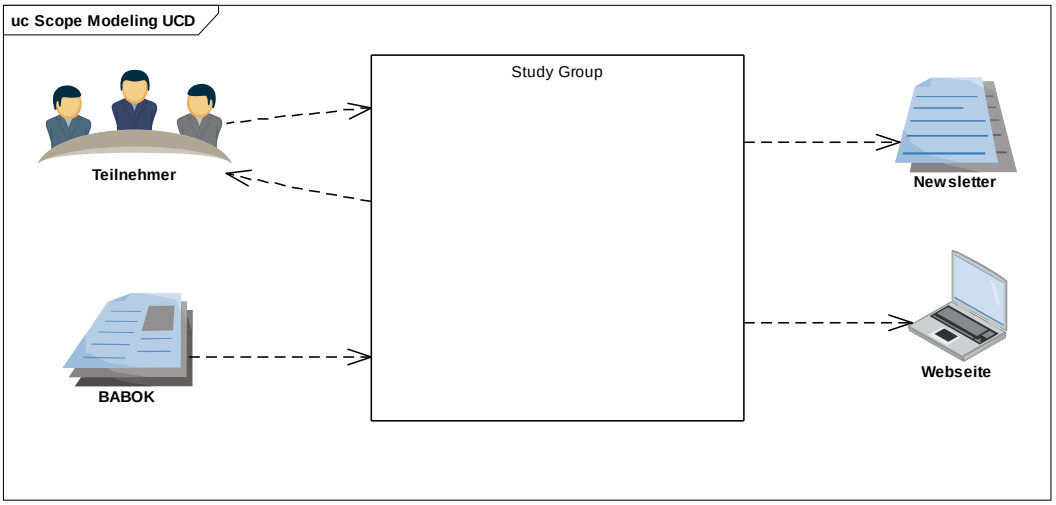
- **Supplier-Consumer**: relates elements by way of the transmission of information or materials between them. Elements can be processes, systems, solution components, and organizational units, for both internal and external entities. Relationships of this type occur in data flow diagrams, business process models, and in collaboration, sequence, and robustness diagrams.

## Use-Case-Diagramme

Die UML bietet kein gesondertes Kontext-Diagramm an, die Freiheitsgrade der UML erlauben es jedoch, mittels UML-Notationsmitteln (Use-Case-Diagramme ohne Use-Cases) ein Kontextdiagramm nachzubilden.

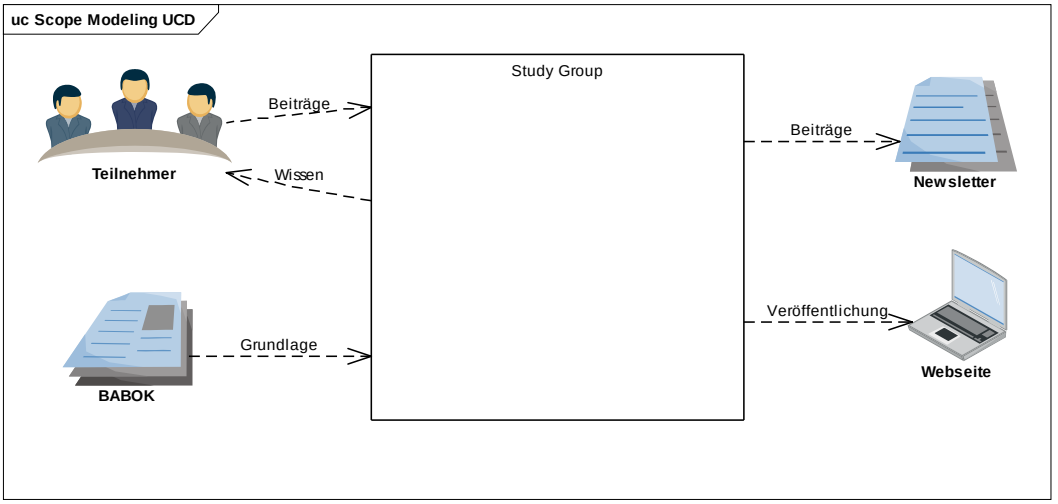
*BABOK v3:*

- **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.




Als Kontextdiagramm ist dieses Modell formal vollständig (!), hat jedoch weniger Aussagekraft als das Datenflußdiagramm.

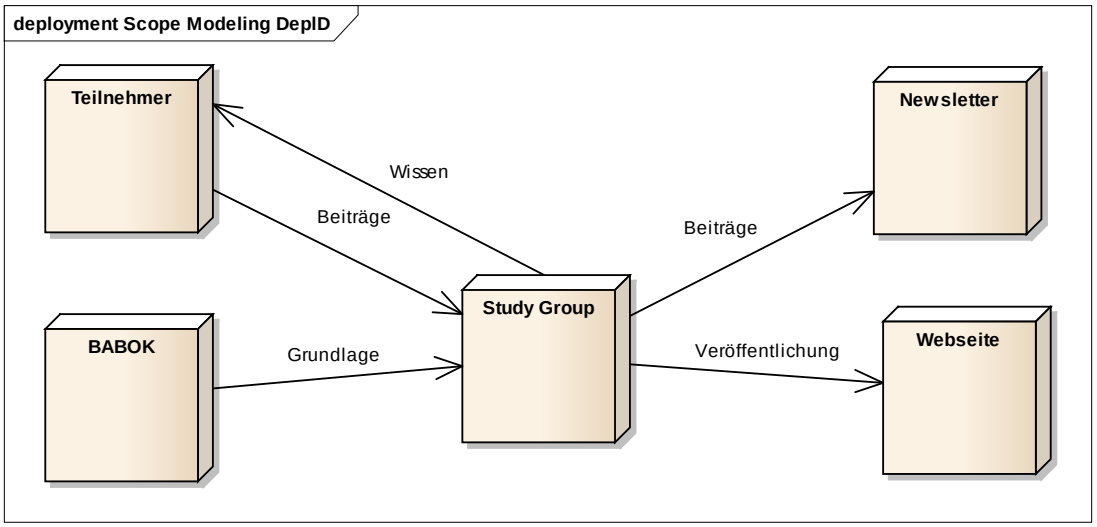
Ergänzt man das Kontextdiagramm um die Datenflüsse, so wird der Informationsgehalt gleich dem Datenflußdiagramm:




## Verteilungsdiagramme

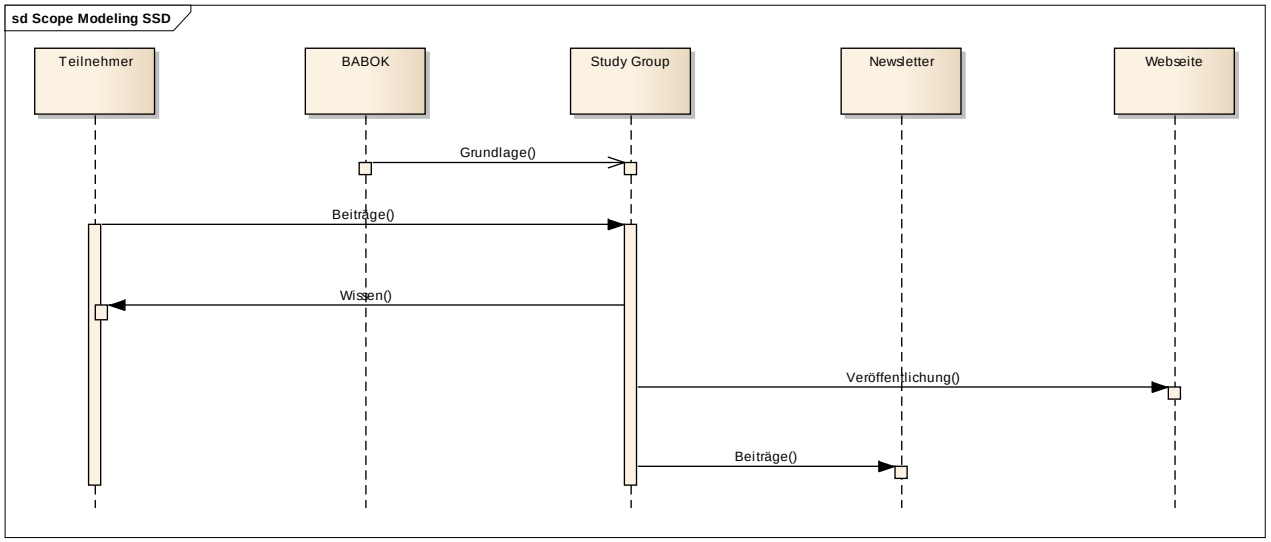
Auch Verteilungsdiagramme können zur physikalischen Kontextabgrezung eines Systems eingesetzt werden. Die meisten Systeme kommunizieren in irgendeiner Art und Weise mit Nachbar- und Fremdsystemen, empfangen wie oben gezeigt Eingaben und Daten und senden Daten und Informationen an Nachbarsysteme.

Mit Hilfe von Verteilungsdiagrammen lassen sich physikalische Fremdsysteme identifizieren und abgrenzen. Der Fokus liegt dabei hauptsächlich auf der Kommunikation des Systems mit seinen Nachbarn:



## Sequenzdiagramme

Sequenzdiagramme eignen sich zur Kontextabgrenzung vor allem dann, wenn Systeme ein wenig unterschiedliches und fest definiertes Interaktionsverhalten haben. In diesem Zusammenhang werden Sequenzdiagramme auch als „system sequence diagrams“ (SSD) bezeichnet:




Auch für die Darstellung von Kontextdiagrammen gilt der UML-Grundsatz: es wird (nur) das dargestellt, was gezeigt werden soll und dafür wird das jeweils geeignete Diagramm verwendet.

## Weiteres

Hat sich der BABOK in der Version 2 mit dem Scope Modelling noch auf die „klassische“ Definition der Kontextabgrenzung beschränkt („Scope models are used to describe the scope of analysis or the scope of a solution“), so hat sich dies im BABOK v3 – nicht zuletzt durch das hier neu eingeführte Core Competence Model – deutlich erweitert und geändert.

*BABOK v3:*

Scope models provide the basis for understanding the boundaries of:

- **Scope of Control**: what is being analyzed, roles and responsibilities, and what is internal and external to the organization.
- **Scope of Need**: stakeholder needs, value to be delivered, functional areas, and organizational units to be explored.
- **Scope of Solution**: requirements met, value delivered, and impact of change.
- **Scope of Change**: actions to be taken, stakeholders affected or involved, and events to cause or prevent.

Entsprechend ist das Core Concept „Context“ mit „The circumstances that influence, are influenced by, and provide understanding of the change. Changes occur within a context [...]“ definiert.

Die Definition des Scope Models mit „A model that defines the boundaries of a business domain or solution“ ist hingegen gleich geblieben.

## Vorteile

- Ein Scope Model zeigt, was innerhalb und was außerhalb des Systems liegt; dies insbesondere auch dafür, wenn sich Anforderungen ändern oder solche dazukommen

*BABOK v3:*

10.41.4 Usage Considerations

.1 Strengths

- A scope model facilitates agreement as a basis for:
- defining contractual obligations,
- estimating the project effort,
- justifying in-scope/out-of-scope decisions in requirements analysis, and
- assessing the completeness and impact of solutions.

## Nachteile

- Durch die High-Level-Sicht ist eine tiefere Detaillierung erforderlich

*BABOK v3:*

.2 Limitations

- An initial, high-level model can lack a sufficient level of granularity, particularly for boundary elements, that is needed to ensure clear scope identification.
- Once a scope is defined, changing it may be difficult due to political reasons and contractual obligations. Meanwhile, many factors can affect the scope validity before the targets are achieved. Such factors as wrong initial assumptions, situation change, evolution of stakeholder needs, or technology innovations may cause a need for revising the scope partially or entirely.
- Traditional scope models cannot address common complex boundaries, such as a horizon (a boundary that is completely dependent on the position of the stakeholder).

## Tasks in BABOK v3

BABOK empfiehlt die Verwendung von Scope Modelling in den folgenden Tasks:

### 3.1 Plan Business Analysis Approach

- **Scope Modelling**: used to determine the boundaries of the solution as an input to planning and to estimating.

### 3.2 Plan Stakeholder Approach

- **Scope Modelling**: used to develop scope models to show stakeholders that fall outside the scope of the solution but still interact with it in some way.

### 5.1 Trace Requirements

- **Scope Modelling**: used to visually depict scope, as well as trace requirements to the area of scope the requirement supports.

### 6.1 Analyze Current State

- **Scope Modelling**: helps define the boundaries on the current state description.

### 6.2 Define Future State

- **Scope Modelling**: used to define the boundaries of the enterprise in the future state.

### 6.4 Define Change Strategy

- **Scope Modelling**: used to define the boundaries on the solution scope and change scope descriptions.

### 7.1 Specify and Model Requirements

- **Scope Modelling**: used to visually show a scope boundary.

### 7.4 Define Requirements Architecture

- **Scope Modelling**: used to identify the elements and boundaries of the requirements architecture.

## Copyright-Note