# Masterarbeit

über das Thema

## Veränderung der Darstellung von Klassen- und Objektdiagrammen in USE durch die Anwendung von Auswahl- und Filteroptionen


**Autor:** Carsten Schipke <schipkca@uni-bremen.de>
Matrikelnummer 2790607


**Version vom:** 9. Januar 2019

**Erstgutachter:** Prof. Dr. Martin Gogolla
**Zweitgutachter:** Dr. Sabine Kuske

Universität Bremen

Nachname Schipke Matrikelnr. 2790607

Vorname/n Carsten

Diese Erklärungen sind in jedes Exemplar der Bachelor- bzw. Masterarbeit mit einzubinden.

## Urheberrechtliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.
Alle Stellen, die ich wörtlich oder sinngemäß aus anderen Werken entnommen habe, habe ich unter Angabe der Quellen als solche kenntlich gemacht.

Datum Unterschrift

## Erklärung zur Veröffentlichung von Abschlussarbeiten

Die Abschlussarbeit wird zwei Jahre nach Studienabschluss dem Archiv der Universität Bremen zur dauerhaften Archivierung angeboten.
Archiviert werden:

1) Masterarbeiten mit lokalem oder regionalem Bezug sowie pro Studienfach und Studienjahr 10 % aller Abschlussarbeiten
2) Bachelorarbeiten des jeweils der ersten und letzten Bachelorabschlusses pro Studienfach und Jahr.

- (●) Ich bin damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
- ( ) Ich bin damit einverstanden, dass meine Abschlussarbeit nach frühestens 30 Jahren (gem. §7 Abs. 2 BremArchivG) im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.
- ( ) Ich bin nicht damit einverstanden, dass meine Abschlussarbeit im Universitätsarchiv für wissenschaftliche Zwecke von Dritten eingesehen werden darf.

Datum Unterschrift

## Zusammenfassung

Diese Masterarbeit betrachtet die Auswahl- und Filteroptionen der Software *UML-based Specification Environment (USE)*, die zur Veränderung von Darstellungen eines Klassen- oder Objektdiagramms verwendet werden kann. Je mehr Elemente in einem Diagramm vorhanden sind, desto schwieriger wird eine übersichtliche Positionierung der enthaltenden Elemente. Auswahl- und Filteroptionen bieten eine komfortable Möglichkeit zur Erzeugung von Teildarstellungen. Die vorhandenen Interaktionen werden auf ihre Verwendung zur Veränderung von Darstellungen untersucht und das Ergebnis zeigt bisher ungenutzte Potenziale auf. Das Ziel ist die Schaffung eines mächtigen und effizienten Sets an Auswahl- und Filteroptionen.

## Abstract

This Master's thesis considers the selection and filtering options of the *UML-based Specification Environment (USE)* software, which can be used to modify representations of a class or object diagram. The more elements that are present in a diagram, the more difficult it becomes to have a clear positioning of the containing elements. Selection and filtering options provide a convenient way to create and focus on part representations. Existing interactions are examined for their use to alter representations, and the result reveals untapped potential up to now. The goal is the creation of a powerful and efficient set of selection and filtering options.

# Inhaltsverzeichnis

# 1. Einleitung

Durch die zunehmende Digitalisierung, den Umgang mit zunehmend größeren Datenmengen sowie neuen Anforderungen und Einsatzgebieten nimmt die Komplexität von Softwaresystemen immer weiter zu. Die erhöhte Komplexität spiegelt sich auch in der zugrundeliegenden Modellierung der Software wieder. Die Modelle bilden immer mehr Objekte, Beziehungen und Abhängigkeiten ab und werden durch die vielen Informationen und ihre Größe schwerer verständlich und unübersichtlich. Aber nicht nur Modelle von Softwaresystemen werden durch die Abbildung von komplexen Vorhaben oder Verhaltensweisen kompliziert. Auch bei der Abbildung von Ökosystemen, den Prognosen zum Wetter oder des demographischen Wandels werden viele Informationen erfasst und in Zusammenhang gesetzt, um auf deren Basis Vorhersagen zu berechnen. Alle diese Beispiele sind durch den enormen Informationsgehalt in ihrer Gänze schwer darstellbar und oftmals ist die Betrachtung eines ausgewählten Teils eines Modells ausreichend.

Die *UML-based specification environment* bietet Möglichkeiten, die Darstellung von Klassen- und Objektdiagrammen so anzupassen, dass die Betrachtung auf Teilszenarien eingegrenzt werden kann. Teilszenarien enthalten eine ausgewählte Menge an Informationen und können einzelne Teile eines großen Ganzen abbilden. Welche Informationen in einem Teilszenario enthalten sind, ist vom jeweiligen Vorhaben abhängig. Allgemein wird ein solches Szenario durch die Reduzierung der sichtbaren Elemente in der Darstellung erzeugt.

Diese Reduzierung kann durch den Einsatz von Auswahl- und Filteroptionen vorgenommen werden, welche die Sichtbarkeit von Elementen verändern. Im Rahmen dieser Ausarbeitung wird die Verfügbarkeit und Anwendung der Auswahl- und Filteroptionen in der *UML-based specification environment* mit den folgenden Fragestellungen untersucht:

- Sind die vorhandenen Auswahl- und Filteroptionen geeignet, um eine Reduzierung der Darstellung eines Klassen- und Objektdiagramms auf ein Teilszenario zu ermöglichen?

- Kann durch die Einführung neuer Benutzungsmöglichkeiten und die Verbesserung vorhandener Funktionen eine effizientere Veränderung der Darstellung erreicht werden?

- Hat die Anzahl der Elemente in einer Darstellung Auswirkungen auf die Nutzbarkeit der Auswahl- und Filteroptionen? Sollte es Einschränkungen geben, welche Funktionen sind davon betroffen und wie sehen die Einschränkung aus?

Im ersten Schritt ist zu ermitteln, welche Interaktionen vorhanden und welche Anpassungsmöglichkeiten einer Darstellung durch diese Optionen gegeben sind. Dabei ist zu untersuchen, auf welche Arten von Elementen die Aktionen angewendet werden können und wie Abhängigkeiten in der Darstellung behandelt werden. Mit den gewonnenen Erkenntnissen kann die zweite Fragestellung bearbeitet werden, ob die bereits implementierten Interaktionen korrekt funktionieren und ob diese verbessert werden können. Zusätzlich zu den möglichen Verbesserungen ist eine Ergänzung der Auswahl- und Filteroptionen in Betracht zu ziehen, sollten ungenutzte Potenziale vorhanden sein. Des Weiteren stellt der Umgang mit Darstellungen, die viele Elemente aufweisen, eine große Herausforderung dar. Hierbei ist zu evaluieren, ob die vorhandenen und möglicherweise neu geschaffenen Auswahl- und Filteroptionen für die Erstellung von Teilszenarien verwendet werden können und durch welche Faktoren die Anwendung erschwert wird.

Neben der Einleitung ist diese Ausarbeitung in sieben weitere Kapitel unterteilt. Im Anschluss an die Einleitung folgt das Kapitel *Grundlagen*, in dem Begriffe und Konzepte eingeführt werden, die für die Ausarbeitung von Bedeutung sind. Zur Beschreibung von Modellierungen werden Klassen- und Objektdiagramme aus der *Unified Modeling Language* verwendet. Daher werden unter anderem der Nutzen, die Einsatzmöglichkeiten und mögliche Notationen der Diagrammarten vorgestellt. Zusätzlich wird das Tool *USE - An UML-based specification environment* betrachtet, das im Kontext dieser Arbeit untersucht und erweitert wird.

USE kann nicht nur für die Darstellung von Klassen- und Objektdiagrammen verwendet werden, sondern bietet noch viele weitere Anwendungsmöglichkeiten. Im dritten Kapitel werden einige Ausarbeitungen vorgestellt, die unter Verwendung von USE erstellt wurden und Problematiken bei der Darstellung von Diagrammen aufweisen. Da in dieser Ausarbeitung die Darstellung durch die Anwendung von Auswahl- und Filteroptionen untersucht und gegebenenfalls verbessert werden soll, kann die Betrachtung der Problematiken bei der Darstellung von vorherigen Ausarbeitungen erste Erkenntnisse und Anregungen über die Darstellungsmöglichkeiten liefern.

In dem Kapitel *Analyse der USE-Version* wird eine Untersuchung der Benutzeraktionen durchgeführt, die Auswirkungen auf die Darstellung von Diagrammen haben. Der Fokus der Betrachtung liegt auf Aktionen, welche die Sichtbarkeit von Elemen-

ten in den Darstellungen verändern können. Das Ergebnis der Untersuchung gibt Aufschluss über die vorhandenen Benutzungsmöglichkeiten und deren Umsetzung in der USE-Version 5.0.1.

Auf Basis der Analyseergebnisse werden im fünften Kapitel neue Interaktionen konzipiert, die die Anpassung von Darstellungen zukünftig erleichtern sollen. Diese Aktionen verändern die Sichtbarkeit oder die Darstellung der vorhandenen Elemente. Bei der Veränderung der Sichtbarkeit ist sicherzustellen, dass die Wirkung von ausgeführten Interaktionen auch wieder aufgehoben oder rückgängig gemacht werden kann. Eine Veränderung der Darstellung einzelner Elemente darf nicht die Regeln zur Notation der *Unified Modeling Language* verletzen und ungültige Darstellungen erzeugen.

Im Zuge der Analyse und Konzeption werden Anforderungen definiert, die die vorhandenen Auswahl- und Filteroptionen verbessern und durch neue Optionen ergänzen. Das Kapitel *Implementierung und Anpassung der Benutzungsmöglichkeiten* beschreibt die Umsetzung der Anforderungen. Es wird auf die Benutzungsmöglichkeiten und die Auswirkungen der Interaktionen eingegangen. Die vollständige Umsetzung der Anforderungen führen zu einem mächtigen Set an Optionen, welche die Sichtbarkeit von Elementen und dadurch deren Darstellung verändern kann.

Die angepassten und neu implementierten Optionen sollen einen Anwender bei der Anpassung einer Darstellung unterstützen. Dazu werden die Benutzungsmöglichkeiten im Kapitel *Evaluation* auf ihre Nutzbarkeit untersucht. Für die Untersuchung werden zwei Darstellungen mit einer Vielzahl an Elementen erzeugt und die unterschiedlichen Auswahl- und Filteroptionen auf dieser Basis auf ihre Anwendbarkeit betrachtet. Das Ergebnis der Betrachtung gibt Aufschluss über die Güte der getätigten Änderungen und bildet die Grundlage für die Beantwortung der Forschungsfragen.

Das Kapitel *Zusammenfassung und Fazit* greift die in der Einleitung gestellte Problematik wieder auf und fasst die Ergebnisse dieser Ausarbeitung zusammen. Auf der Basis der Evaluation werden die Forschungsfragen beantwortet und ein Ausblick auf zukünftige Verbesserungen für die Darstellung von Diagrammen und die Nutzbarkeit der Interaktionen in USE gegeben.

# 2. Grundlagen

Im Zuge dieser Ausarbeitung werden verschiedene Frage- und Problemstellungen im Zusammenhang mit Darstellungen von Diagrammen bearbeitet. Eine erzeugte Darstellung eines Diagramms kann auf unterschiedlichste Weisen verändert und an gegebenen Anforderungen angepasst werden. Bei jeder Anpassung ist sicherzustellen, dass die vorliegende Darstellung gültig ist und nicht die Eigenschaften des zugrundeliegenden Modells oder der Notation verletzt werden. In diesem Kapitel werden Begriffe und Konzepte eingeführt, auf denen die nachfolgenden Kapitel aufbauen.

Bereits in der Einleitung dieses Kapitels wurden die Begrifflichkeiten Modell und Diagramm verwendet. Diese Begriffe werden im Kontext dieser Ausarbeitung häufiger vorkommen und daher voneinander abgegrenzt. Der Ausdruck Modell stammt von dem lateinischen Wort *modulus* ab und beschreibt die Abbildung eines vorhandenen Originals. Ein Modell ist eine abstrakte Form eines Originals und ist der Versuch die Eigenschaften des Originals zu erfassen und abzubilden. Folglich handelt es sich bei einem Modell um eine Abstraktion von Objekten aus der realen Welt, wobei die Betrachtung zu einem bestimmten Zeitpunkt und aus einer bestimmten Perspektive erfolgt. Der Detailgrad und die Informationsdichte sind im Vergleich zu dem originalen Objekt reduziert und je nach abzubildender Aufgabe unterschiedlich ausgeprägt. Allgemein unterstützen Modelle bei der Abbildung von komplexen Vorgängen, Problemen und Lösungen und bieten eine Grundlage für die Kommunikation über komplexe Sachverhalte. Ein Diagramm ist eine konkrete Ausprägung eines Modells und enthält unterschiedlichste Informationen über Eigenschaften und Verhalten von den modellierten Elementen. Einige Diagramme wurden standardisiert und mit Regeln für Notation und Gestaltung versehen, die bei der Verwendung einzuhalten sind. Eine inkorrekte Notation wäre eine Verletzung dieser Regeln und kann in einer fehlerhaften Darstellung resultieren (vgl. [RJB04] S.15ff., [KB18] S.18ff.).

Zusammengefasst sind Modelle und Diagramme Darstellungen oder Abbildungen eines Originals beziehungsweise eines Ausschnittes des Originals. Je nach dem Fokus der Betrachtung kann die Darstellung eines Ausschnittes ausreichend sein. Ein Wald kann ein sehr komplexes Objekt sein. Eine Vielzahl unterschiedlicher Lebewesen nutzen einen Wald als Rückzugsort, Lebensraum oder Nahrungsquelle. Zu den Lebewesen eines Waldes können Tiere und viele Arten von Vegetationen gehören. Die Erfassung aller Lebewesen, ihrer Eigenschaften und Verhaltensweisen resultieren in einer enormen Menge an Informationen. Zudem erhöht sich die Komplexität, wenn die Beziehungen der einzelnen Lebewesen eines Waldes zueinander abgebildet werden. Des Weiteren verfügt ein Wald selbst auch über Eigenschaften. Ein Wald hat beispielsweise eine Ausdehnung und umfasst ein Gebiet, das mit Koordinaten

beschrieben werden könnte. Die Fülle der Informationen für das komplexe Objekt Wald kann in unübersichtlichen Darstellungen resultieren. Für viele Betrachtungen ist allerdings nur ein Ausschnitt erforderlich, sodass mit Hilfe von Auswahl- und Filterfunktionen die Anzahl der Informationen reduziert werden muss, um eine übersichtliche Darstellung zu erzeugen.

In diesem Kapitel werden mit den Klassen- und Objektdiagramm zwei Arten von Diagrammen vorgestellt, auf deren Basis unterschiedliche Darstellungen erzeugt und im weiteren Verlauf verschiedene Auswahl- und Filterfunktionen angewendet werden. Die beiden Diagrammarten sind standardisiert, sodass bei deren Verwendung die vorhandenen Normen zur Notation einzuhalten sind. Dazu werden im Rahmen dieses Kapitels die Notationen von verschiedenen Elementen in den beiden Diagrammarten betrachtet. Die gezeigten Darstellungen werden mit dem Tool *USE - An UML-based specification environment* erzeugt.

## 2.1. UML: Unified Modeling Language

Die *Unified Modeling Language* (UML) ist eine Sprache zur Modellierung von Objekten aus der realen oder einer vorgestellten Welt. Die Ausdrucksweise entstand aus dem Vorhaben eine standardisierte Sprache zur objektorientierten Modellierung zu definieren, die die Spezifizierung, Erstellung und Dokumentation von Systemen ermöglicht. Dabei stehen zur Visualisierung verschiedene Arten von Diagrammen zur Verfügung, die sich in die zwei Kategorien *Struktur-* und *Verhaltensdiagramme* einteilen lassen. Strukturdiagramme werden für die Darstellung eines Systemzustands zu genau einem Zeitpunkt verwendet. Diese Diagramme können zur Modellierung von statischen und zeitunabhängigen Elementen genutzt werden und bilden den logischen Aufbau oder genau einen Zustand eines Systems ab. Die Abbildung *Taxonomie der Diagrammtypen in UML 2.5*, der *Object Management Group (OMG)*, benennt die sieben verschiedenen Arten der Strukturdiagramme. Die Verhaltensdiagramme ermöglichen eine dynamische Perspektive und sind für die Darstellung von zeitlichen Abläufen oder Interaktionen geeignet (vgl. [RJB04] S.3f., [KS15] S.20f, [OMG17] S.725).

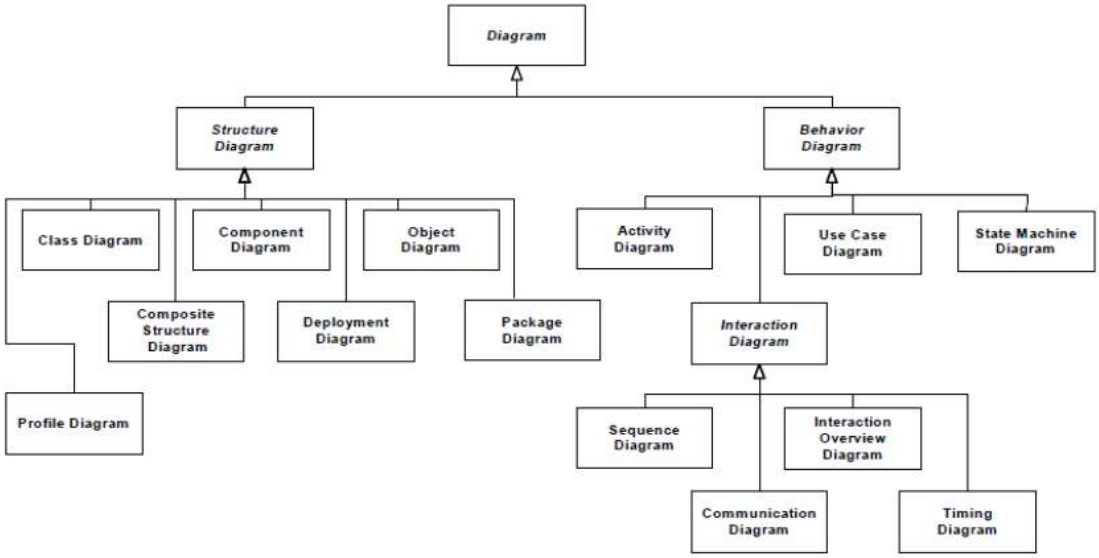


Abbildung 1: Taxonomie der Diagrammtypen in UML 2.5

In den weiteren Kapiteln sind nur die Klassen- und Objektdiagramme der Kategorie *Strukturdiagramme* von Bedeutung und werden in den folgenden Abschnitten näher betrachtet. Die Betrachtung erfolgt auf der Basis der UML-Spezifikation in der Version 2.5.1[1], für deren Pflege und Weiterentwicklung die *Object Management Group (OMG)* verantwortlich ist.

### 2.1.1. UML: Klassendiagramm

Klassendiagramme werden verwendet, um die Struktur eines vorhandenen oder zu entwerfenden Systems abzubilden. Es werden die zentralen Elemente eines Systems abstrahiert und in Form von Klassen dargestellt. Eine Klasse ist das zentrale Konzept eines Klassendiagramms und beschreibt die abstrakte Form eines Originals. Die Informationen können über Attribute und das Verhalten über Operationen abgebildet werden. Zudem können Klassen sich gegenseitig beeinflussen und in Beziehung zueinander stehen. Die Beziehung zwischen Klassen wird als Assoziation oder Generalisierung bezeichnet. Die UML ermöglicht unterschiedliche Arten von Beziehungen zwischen Klassen, die im Laufe dieses Abschnittes vorgestellt werden (vgl. [Koc15] S.47f., [RQZ12] S.108f).

Der Detail- und Reifegrad eines Diagramms ist je nach Bedürfnis und Anforderung unterschiedlich stark ausgeprägt. Klassendiagramme werden unter anderem in der Softwareentwicklung eingesetzt. Abhängig von der Phase der Entwicklung kann ein Klassendiagramm verschiedene Ausprägungen in Bezug auf den Detail- und Reifegrad besitzen. In einer frühen Entwicklungsphase ist die Visualisierung von zentralen Klassen und deren Beziehungen zu anderen Klassen von besonderer Bedeutung, um

[1] `https://www.omg.org/spec/UML/2.5.1/ (zuletzt aufgerufen am 27.12.2018)`

die Architektur der Software darstellen zu können. Die Eigenschaften und Verhaltensweisen von einzelnen Klassen sind zu diesem Zeitpunkt nebensächlich. Im Verlauf der Entwicklung werden die Klassen mit mehr Informationen versehen. Folglich wird das Diagramm detailreicher und wirkt ausgereifter. Ein ausgereiftes und detailreiches Klassendiagramm kann Softwareentwickler nicht nur bei der Dokumentation der Systemarchitektur unterstützen, sondern auch als Grundlage für die Generierung von Quellcode für das Softwaresystem verwendet werden (vgl. [RQZ12] S.113f).

Die UML stellt eine einheitliche Basis für die Erstellung von Klassendiagrammen zur Verfügung. Es ist definiert, welche Elemente in Klassendiagrammen zulässig sind und wie diese Elemente zu notieren sind. Eine Verletzung dieser formalen Regeln führt zu einer ungültigen Darstellung und bedeutet, dass die Notation oder die Struktur eines Modells nicht eingehalten wird (vgl. [Koc15], S43ff). Die Abbildung *Klassendiagramm StudentWorld* zeigt eine gültige Notation vieler Elemente, die die UML für ein Klassendiagramm vorsieht. Anhand dieses Beispiels werden viele Konzepte und Notationen in diesem und den folgenden Kapiteln erläutert. Das Beispiel beschriebt eine mögliche Welt eines Studenten und stammt aus der Vorlesung *Design of Information Systems*[2] an der Universität Bremen.

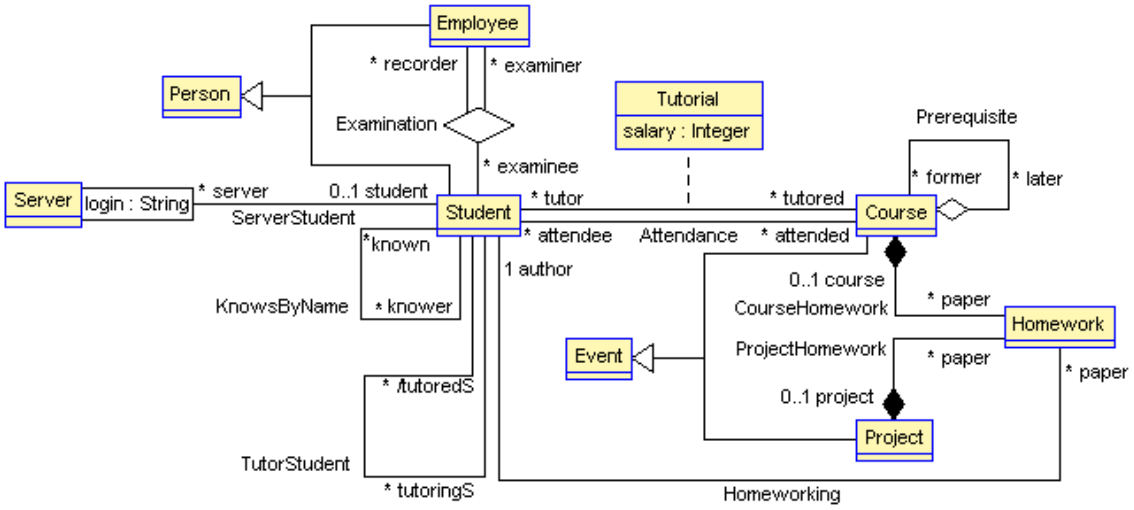


Abbildung 2: Klassendiagramm *StudentWorld*

Das zentrale Konzept eines Klassendiagramms ist die Klasse. Eine Klasse verfügt über einen Namen und kann Attribute, Operationen und Beziehungen aufweisen. Über die Attribute können Eigenschaften einer Klasse und über die Operationen ein mögliches Verhalten dargestellt werden. Während eine Klasse nicht zwingend über Attribute oder Operationen verfügen muss, setzt eine ordnungsgemäße Nota-

[2] `http://www.db.informatik.uni-bremen.de/teaching/courses/ss2018_eis/ (zuletzt aufgerufen am 27.12.2018)`

tion einen Namen für eine Klasse voraus. Eine Klasse wird in der UML durch ein Rechteck mit durchgezogen Linien dargestellt. Der verpflichtende Name ist zentral in diesem Rechteck anzuordnen. Unterhalb dieses ersten Rechtecks kann es weitere Rechtecke geben, in denen die Attribute und Operationen beschrieben werden. Bei der Definition der Attributnamen ist darauf zu achten, dass diese kleingeschrieben sind. Das Attribut *salary* der Assoziationsklasse *Tutorial*, eine spezielle Form einer Klasse, zeigt eine mögliche Darstellung einer Klasse inklusive eines Attributes. Neben den Namen des Attributs ist der zugehörige Datentyp anzugeben. In diesem Beispiel kann das genannte Attribut mit gültigen Zahlen aus dem Wertebereich des Datentyps *Integer* belegt werden. Da eine Klasse eine abstrakte Form eines Elementes ist, wird in einem Klassendiagramm das Attribut mit keiner konkreten Zahl belegt. Unterhalb der Attribute können Operationen angegeben werden, um Verhaltensweisen zu beschreiben. Eine Operation besteht aus dem Operationsnamen, Parametern und einem Rückgabetyp. In dem abgebildeten Klassendiagramm sind keine Operationen vorhanden und werden im weiteren Verlauf auch nicht vorkommen (vgl. [RQZ12] S.115ff).

Das *Klassendiagramm StudentWorld* umfasst neun Klassen, die durch die Verwendung von Linien und zum Teil weiterer Notationen in Beziehung gesetzt werden. Diese unterschiedlichen Beziehungstypen verbinden Klassen nicht nur miteinander, sondern können auch Anforderungen an die Klassen und deren Existenz stellen. Eine besondere Form der Beziehung zwischen Klassen ist die Generalisierung. An einer Generalisierung sind zwei Klassen beteiligt, die mit einer Linie verbunden werden. An einem Ende dieser Linie befindet sich eine nicht ausgefüllte dreieckige Spitze. Diese Notation ermöglicht die Unterscheidung in eine allgemeine und eine spezielle Klasse. Die Pfeilspitze zeigt auf die allgemeinere Klasse, die auch als Oberklasse bezeichnet wird. Das Ende der Verbindung ohne Pfeilspitze zeigt auf die speziellere Klasse beziehungsweise Unterklasse. In dem Beispiel existieren mehrere Generalisierungen. Die Klassen *Employee* und *Student* sind speziellere Formen der Klasse *Person*. Aus der Sicht der beiden Unterklassen stellt die Klasse *Person* eine Verallgemeinerung dar. Sollte die Oberklasse über Attribute oder Operationen verfügen, die nicht durch das Schlüsselwort *private* in ihrer Sichtbarkeit eingeschränkt sind, werden die Unterklassen ebenfalls über diese Attribute und Operationen verfügen. In der Programmierung wird von einer Vererbung gesprochen (vgl. [RQZ12] S.135ff).

Neben den Generalisierungen sind noch weitere Arten von Beziehungen in dem Beispiel abgebildet. Alle noch nicht beschriebenen Arten werden als Assoziationen bezeichnet. Eine Assoziation spezifiziert die Beziehung zwischen Klassen. Ähnlich wie bei den Klassen besitzt eine Assoziation auch einen Namen. Die Enden der Beziehungen können einen Rollennamen und eine Multiplizität aufweisen. Über den

Rollennamen kann angegeben werden, welche Rolle eine Klasse in der Beziehung einnimmt. Die Multiplizität schränkt die Anzahl der an einer Beziehung teilnehmenden Objekte ein. Der Begriff Assoziation wird als Sammelbegriff für eine Vielzahl spezieller Assoziationen verwendet, die nun vorgestellt werden:

- Binäre Assoziation
- Reflexive Assoziation
- N-äre Assoziation
- Abgeleitete Assoziation
- Aggregation
- Komposition
- Assoziation mit Assoziationsklasse
- Qualifizierte Assoziation

Eine binäre Assoziation verbindet zwei Klassen durch die Verwendung einer durchgezogenen Linie miteinander. Bei der Beziehung *Attendance*, die die Klassen *Student* und *Course* verbindet, wird von einer binären Beziehung gesprochen. Es wird die Teilnahme von Studenten an Kursen modelliert. Studenten können mehrere Kurse besuchen beziehungsweise Kurse können mehrere Teilnehmer in Form von Studenten haben. Ein weiteres Beispiel für eine binäre Assoziation wäre *Homeworking* zwischen den Klassen *Student* und *Homework*. Im Gegensatz zu der Beziehung *Attendance* wird bei *Homework* eine Eingrenzung der an der Beziehung teilnehmenden Elemente vorgenommen. Eine Hausarbeit darf nur von genau einem Autor in Form eines Studenten stammen. Ein Student darf jedoch eine Vielzahl von Hausarbeiten anfertigen, solange nur er der Autor der Ausarbeitung ist (vgl. [RQZ12] S.142f).

Eine besondere Form der binären Assoziation ist die reflexive Assoziation. Bei dieser Assoziation handelt es sich um eine zirkuläre Beziehung. Bei einer zirkulären Beziehung enden die beiden Endpunkte der Verbindung an der selben Klasse. Es ist nur eine Klasse an dieser Beziehung beteiligt. Bei der Assoziation *KnowsByName* handelt es sich um eine reflexive Assoziation. Es wird ausgedrückt, dass sich mehrere Studenten kennen. Dabei nimmt ein Student die Rolle des bekannten Studenten ein und der andere Student kennt diesen Studenten. Ein Student kann mehrere andere Studenten kennen und auch mehreren anderen Studenten bekannt sein (vgl. [RQZ12] S.149).

Die beiden bisher vorgestellten Assoziationen konnten maximal zwei Klassen miteinander verbinden. Die n-äre Assoziation wird als Sammelbegriff für Assoziationen verwendet, die über mehr als zwei Kanten verfügen. Das zentrale Element für die Notation stellt eine nicht ausgefüllte Raute dar, die mit durchgezogenen Linien mit den entsprechenden Klassen verbunden wird. Jede Verbindung von der Raute zu einer Klasse wird als abgehende Kante gewertet. Weist die Raute drei Verbindungen zu einer Klasse auf, wird von drei abgehenden Kanten gesprochen. Die Anzahl der Kanten ersetzt das *n* in n-äre und daher wird von einer ternären Assoziation gesprochen, wenn drei Kanten vorhanden sind. An der ternären Assoziation *Examination* sind drei Kanten und zwei Klassen beteiligt. Aus dem Wort ternär können keine Rückschlüsse auf die Anzahl der beteiligten Klassen gezogen werden, sondern nur auf die Anzahl der Verbindungen. Die Beziehung bildet eine Prüfungssituation ab, an der Prüflinge in Form von Studenten und Angestellte teilnehmen. Die Angestellten nehmen die Rolle des Prüfers und Protokollanten ein (vgl. [RQZ12] S.155f).

Eine abgeleitete Assoziation liegt vor, wenn eine Assoziation durch andere Assoziationen berechnet beziehungsweise dargestellt werden kann. Studenten können über *Tutorial* ein Tutorium geben und über *Attendance* an einem Kurs teilnehmen. Die abgebildete Beziehung *TutorStudent* gibt an, welche Studenten von einem anderen Studenten unterrichtet werden. Bei *tutoredS* liegt eine Ableitung vor, da *tutoredS* identisch mit *tutored.attendee* ist. Von einem Student kann über die Beziehung *Tutorial* zu der Klasse *Course* navigiert werden. Das Ergebnis sind alle Kurse für die der Student Tutor ist. Auf Basis dieses Zwischenergebnisses kann über *attendee* zurück zu der Klasse *Student* navigiert werden. Damit verändert sich das Ergebnis von der Menge aller Kurse, für die ein Student Tutor ist, zu einer Menge von Studenten, die ein Tutorium von diesem Studenten besuchen. Die Ergebnismenge ist identisch mit der Menge, die über *tutoredS* erzeugt wird (vgl. [Rum13] S.54f).

Bei einer Aggregation handelt es sich um eine spezielle Form der Assoziation, die eine sogenannte Teil-Ganzes-Beziehung ausdrücken kann. Sie wird verwendet, um ein Konstrukt darzustellen, das aus mehreren Teilen zusammengesetzt ist. Die Notation erfolgt durch eine durchgezogene Linie. An einem Ende der Verbindung befindet sich ein nicht ausgefüllter Diamant. Der Diamant zeigt auf die Klasse, die das Ganze darstellt, während das andere Ende der Verbindung auf die einzelnen Teile zeigt. Die Semantik der Aggregation lässt die Leseart *besteht aus* oder *ist Teil von* zu. In der *StudentWorld* existiert die Aggregation *Prerequisite*. Bei dieser Beziehung handelt es sich um eine spezielle Form der Aggregation, da sie zirkulär ist und beide Enden der Linie an der Klasse *Course* enden. Durch diese Assoziation wird eine Reihenfolge von Kursen angegeben. Ein Kurs kann einen Vorgänger und einen Nachfolger haben. Beispielweise kann modelliert werden, dass ein vertiefender

Kurs mehrere Grundkurse voraussetzt. Die Multiplizität ermöglicht auch, dass der vertiefende Kurs ohne zugehörige Grundkurse angeboten werden kann (vgl. [RQZ12] S.153).

Die Komposition bildet ebenfalls Teil-Ganzes-Beziehungen ab, formuliert aber strikte Regeln an die Existenz, der an der Beziehung beteiligten Elemente. Eine Komposition wird durch eine durchgezogene Linie dargestellt, wobei ein Ende mit einem schwarz ausgefüllten Diamanten versehen ist. Die Klasse, auf die der Diamant zeigt, nimmt die Rolle des Ganzen ein. Die Besonderheit der Komposition ist, dass die Teile ohne das Ganze nicht existieren dürfen. Die Lebensdauer der einzelnen Teile ist folglich von der Lebensdauer des Ganzen abhängig. Des Weiteren ist auf der Seite des Ganzen ausschließlich die Multiplizität 0..1 oder 1 erlaubt. In der *StudentWorld* stehen die Klassen *Course* und *Homework* in einer Kompositionsbeziehung. Werden Hausarbeiten im Kontext eines Kurses erstellt, ist die Lebensdauer der Hausarbeiten abhängig von der Existenz des Kurses. Wird der Kurs nicht angeboten oder findet nicht statt, können auch keine Hausarbeiten zu diesem Kurs existieren (vgl. [RQZ12] S.153f).

Eine Assoziation kann mit einer Assoziationsklasse erweitert werden. Eine Assoziationsklasse ist wie eine Klasse aufgebaut und kann über Attribute und Operationen verfügen. Sie wird verwendet um eine Beziehung mit zusätzlichen Eigenschaften zu versehen. Die Assoziationsklasse wird wie eine Klasse notiert und durch eine gestrichelte Linie mit der durchgezogenen Linie einer Assoziation verbunden. Die Namen der Assoziation und der Assoziationsklasse müssen identisch sein. Die Verwendung einer Assoziationsklasse ist nicht auf eine bestimmte Art von Assoziationen begrenzt. So können auch Assoziationsklassen für Aggregationen und Kompositionen gebildet werden. In dem gezeigten Klassendiagramm sind die Klassen *Student* und *Course* mit der Assoziation und der Assoziationsklasse *Tutorial* verbunden. Studenten können Tutor für Kurse werden und bekommen für die Durchführung von Tutorien ein Gehalt. Das Gehalt wird über das Attribut *salary* der Assoziationsklasse abgebildet (vgl. [RQZ12] S.157f).

Mit Hilfe einer Assoziationsklasse kann eine Assoziation mit Attributen versehen werden. Eine weitere Möglichkeit eine Assoziation mit Attributen zu erweitern, sind die qualifizierenden Attribute, die zu einer qualifizierten Assoziation führen. Eine qualifizierte Assoziation verbindet zwei Klassen miteinander. Die Verbindung erfolgt durch eine durchgezogene Linie, deren eines Ende nicht direkt an einer Klasse endet, sondern an einem zusätzlichen Rechteck, das zwischen dem Verbindungsende und der Klasse notiert wird. In diesem Rechteck werden die Attribute aufgelistet. Des Weiteren löst die Verwendung von qualifizierenden Attributen eine Herabsetzung

der Multiplizität aus. Die Kombination von einer Klasse mit ihrem qualifizierenden Attribut hat zur Folge, dass auf der anderen Seite der Beziehung die vorhandene Multiplizität auf 0..1 reduziert wird. Die Beziehung *ServerStudent* zwischen den Klassen *Server* und *Student* verfügt über das qualifizierende Attribut *login* vom Datentyp *String*. Die Anmeldung eines Studenten an einem Server ist nur erfolgreich, wenn der Student die korrekte Kombination von *Server* und *login* verwendet. Eine qualifizierte Assoziation kann auch durch eine Assoziation mit Assoziationsklasse ausgedrückt werden. Das qualifizierende Attribut wird über die Assoziationsklasse umgesetzt (vgl. [RQZ12] S.154), [Fow04] S.97f).

In diesem Abschnitt wurden die zentralen Konzepte der Klassendiagramme eingeführt und erläutert. Diese Konzepte prägen die Darstellung eines Klassendiagramms. Bei der Verwendung der Konzepte sind einige Regeln bei der Anwendung und deren Notation zu beachten. Dieses Verständnis ist für die folgenden Kapitel essentiell, da Darstellungen durch die Anwendung verschiedener Interaktionen verändert werden. Nach jeder Veränderung ist sicherzustellen, dass das Diagramm nur gültige Notationen enthält und alle Elemente korrekt dargestellt werden. Aus diesem Grund wurden die Konzepte und deren Notationen ausführlich und unter Verwendung von Beispielen vorgestellt. Die Erzeugung von ungültigen Zuständen durch die Veränderung eines Diagramms ist zu vermeiden.

### 2.1.2. UML: Objektdiagramm

Ein Objektdiagramm ist eine weitere Form eines Strukturdiagramms in der UML. Dieses Diagramm wird für die Darstellung eines konkreten Systemzustandes zu genau einem Zeitpunkt verwendet. Alle vorhandenen Instanzen und deren Beziehungen zueinander werden erfasst und abgebildet. Die zu verwendenden Notationen ähneln den Notationen in einem Klassendiagramm. Diese Ähnlichkeit ist auf die starke Verwandtschaft der beiden Diagramme zurückzuführen. Ein Objektdiagramm entsteht durch Instanziierungen der in einem Klassendiagramm vorhanden Elemente.

Das zentrale Konzept der Objektdiagramme ist das Objekt. Ein Objekt ist eine Instanz einer Klasse. Für ein besseres Verständnis der Zusammenhänge zwischen den beiden Diagrammen werden zuerst die beiden Konzepte Klasse und Objekt miteinander verglichen und voneinander abgegrenzt. Ein Objekt repräsentiert einen konkreten Gegenstand aus der realen Welt oder aus einer vorgestellten Welt. Dabei kann es sich um greifbare Elemente wie ein Blatt, Zweig oder Baum handeln oder um Sammelbegriffe wie zum Beispiel Mischwälder. Das Objekt wird durch seine Identität, seinem Zustand und Verhalten geprägt. Eine Klasse ist eine Art Schablone oder ein Bauplan für Objekte und gibt die Struktur der Objekte vor. Es wird festgelegt,

über welche Attribute und Operationen ein Objekt verfügt. Die Klasse *Baum* könnte beispielsweise definieren, dass jedes *Baum*-Objekt über ein Alter und eine Art verfügt. Für das Objekt *Eiche* der Klasse *Baum* werden die beiden Attribute mit Werten belegt werden. Die abstrakte Darstellung wird mit objektspezifischen Informationen angereichert, sodass eine konkrete Ausprägung entsteht.Die Abbildung *Klasse und Objekt* zeigt den Unterschied an einer Modellierung für einen Baum (vgl. [Koc15] S.43f, [RQZ12] S.108f).

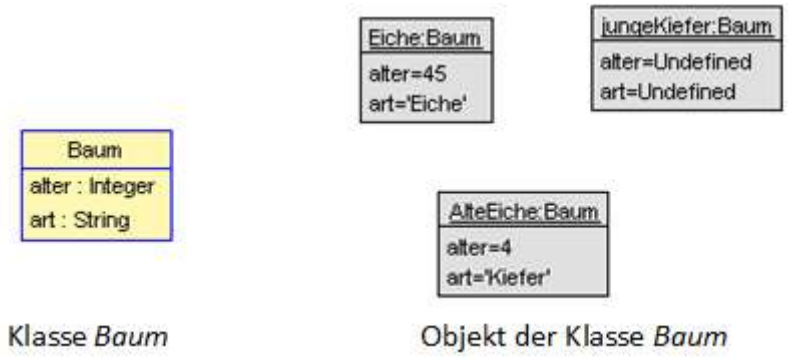


Abbildung 3: Klasse und Objekt

Eine Assoziationsklasse ist eine besondere Form einer Klasse. Die Instanziierung wird Linkobjekt genannt und ist mit einer Instanz einer Assoziation, einem Link, verbunden. In Objektdiagrammen werden Objekte, Linkobjekte und Links anstatt von Klassen, Assoziationsklassen und Assoziationen verwendet. Generalisierungen werden nicht explizit dargestellt. Ein Objekt, dessen Klasse über eine Oberklasse verfügt, besitzt die Attribute und Operationen der Oberklasse.

Die Abbildung *Beispiel eines Objektdiagramms* zeigt Objekte und Links, die auf Basis des Klassendiagramms aus dem vorherigen Abschnitt erstellt wurden. Die Instanzen stammen von den Klassen *Student*, *Homework*, *Course* und *Project* sowie von den Assoziationen *Homeworking*, *CourseHomework*, *ProjectHomework* und *KnowsByName*.

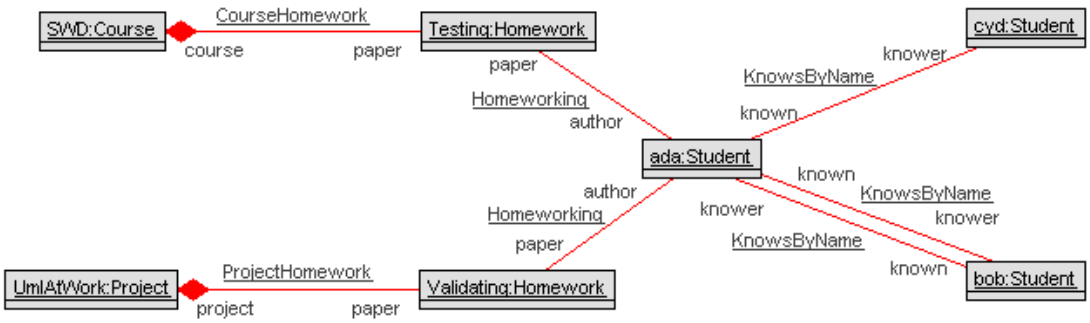


Abbildung 4: Beispiel eines Objektdiagramms

Jedes dargestellte Objekt hat einen Namen, der im oberen Rechteck der Darstellung eines Objektes angegeben ist. Auf dem Namen folgt ein Doppelpunkt und die Angabe der zugehörigen Klasse. Das Objekt *ada* ist eine Instanz der Klasse *Student* und verfügt über keine Attribute und Operationen. Zusätzlich zu der Studentin *ada* werden die Studenten *bob* und *cyd* abgebildet. Die bereits vorgestellte Assoziation *KnowsByName* setzt Studenten miteinander in Beziehung und ermöglicht die Darstellung von Bekanntschaften. Die Studenten *ada* und *bob* kennen sich gegenseitig, da zwischen den beiden Instanzen zwei Links vorhanden sind und jedes Objekt einmal die Rolle *knower* und *known* zugeordnet ist. Der Name des Links ist unterstrichen und entspricht dem Namen der Assoziation, in diesem Fall *KnowsByName*. Der Student *cyd* weist keine bilaterale Beziehung zu den anderen beiden Studenten auf. Daraus folgt, dass *cyd* zwar *ada* kennt, aber *ada* kennt *cyd* nicht. Während *bob* und *cyd* keine Beziehungen zu Instanzen der Klasse *Homework* haben, schreibt *ada* gleichzeitig an zwei Hausarbeiten. Die Hausarbeit *Testing* ist Bestandteil des Kurses *SWD*, während *Validating* zu dem Projekt *UmlAtWork* gehört.

Durch das Erzeugen oder Zerstören von Objekten sowie das Ausführen von Operationen, die die Belegung der Attribute eines Objektes verändern, wird die Momentaufnahme in Form eines Objektdiagrammes abgewandelt. Objektdiagramme gehören zu den Strukturdiagrammen und jede Änderung eines Zustandes resultiert in einem neuen Zustand, der über ein neues Objektdiagramm abzubilden ist.

Zusammengefasst ermöglicht ein Objektdiagramm die Darstellung von Instanzen. Die Elemente des Klassendiagramms dienen als Schablonen für die Instanzen und die Attribute der Instanzen werden mit Werten verstehen. Anstatt von Klassen, Assoziationsklassen und Assoziationen wird von Objekten, Linkobjekten und Links gesprochen. Die Tabelle *Klassen- und Objektdiagramme* verdeutlicht die Unterschiede und die entsprechende Verwendung der beiden Strukturdiagramme (vgl. [RQZ12] S.186).

| **Klassendiagramm** | **Objektdiagramm** |
|---|---|
| Darstellung in einer abstrahierten Form (Klassen, Assoziationen, Assoziationsklassen, Generalisierungen) | Verwendung konkreter Ausprägungen (Objekte, Links, Linkobjekte) |
| Übersichtliche Modellierung von abstrakten Elementen/Typen | Darstellung von konkreten Instanzen |
| Erlaubt keine Darstellung von Attributwerten | Abbildung eines Systemzustandes möglich |

Tabelle 1: Klassen- und Objektdiagramme

### 2.1.3. UML: Metamodell

Am Beispiel des Modells *StudentWorld* wurden Klassen- und Objektdiagramme eingeführt. Jede Art von Diagramm verfügt über unterschiedliche Elemente, die zur Modellierung verwendet werden können. Die UML beschreibt, welche Notationen anzuwenden sind und welche Konzepte für die Modellierung verwendet werden dürfen. In dieser Ausarbeitung wurde bisher nicht betrachtet, ob die *Unified Modeling Language* auch modelliert werden kann. Ist es möglich ein Diagramm zu erzeugen, dass die UML abbildet?

Die Antwort lautet Ja. Es existiert ein Modell, das die UML abbildet. Dieses Modell wird UML-Metamodell genannt und kann auch wieder durch ein anderes Modell beschrieben werden. Zuerst wird der Begriff Metamodell eingeführt, der sich aus den beiden Wörtern *Meta* und *Modell* zusammensetzt. Der Begriff des Modells wurde bereits hergeleitet, weshalb an dieser Stelle keine erneute Erklärung erfolgt. Das Wort *meta* kommt aus dem Griechischen und kann mit *über* übersetzt werden. Folglich ist ein Metamodell ein übergeordnetes Modell. Ein Modell ist ein Metamodell, wenn das Modell ein anderes Modell beschreibt.

Die Beschreibung von Modellen durch weitere Modelle kann prinzipiell beliebig fortgesetzt werden. Das Metamodell wird von einem anderen Modell beschrieben, welches wieder beschrieben werden kann und so kann eine endlose Abfolge von Modellen entstehen. Zur Unterbrechung dieser Verkettung von Modellen wurde von der *Object Management Group* eine oberste Instanz für die Metamodellierung festgelegt: Die *Meta Object Facility* (MOF) (vgl. [SG06] S.289).

Durch die Einführung der *Meta Object Facility* entsteht eine Hierarchie, die in der Abbildung *Schichten zur Modellierung am Beispiel der UML*[3] gezeigt wird. Die Darstellung verfügt über vier Schichten, die von *M0* bis *M3* nummeriert sind. Die Schicht *M0* beschreibt real existierende oder vorgestellte Objekte. Bei diesen Objekten kann es sich um ein konkretes Buch, die Studentin *ada* oder einen Baum in ihrer Nähe handeln. Auf der nächsten Ebene werden Objekte mit Hilfe von UML-Diagrammen abgebildet. Handelt es sich bei der gewählten Form um ein Klassen- oder Objektdiagramm, können die genannten Objekte durch die Verwendung von Klassen, Attributen und weiterer vorgestellter Konzepte beschrieben werden. Die Möglichkeiten wurden in den vorherigen Abschnitten beschrieben. Die Schicht Metamodell beschreibt formal die Elemente, die bei der Modellierung mit der UML verwendet werden können. In diesem Kapitel wurden diese Definitionen nicht durch

[3] `https://blog.sophist.de/wp-content/uploads/2016/03/Schichten_Modellbildung-e1456837046745.png (zuletzt aufgerufen am 27.12.2018)`

ein Modell dargestellt, sondern in Form von textuellen Beschreibungen und Beispielen vermittelt. Die MOF bildet den höchsten Grad der Abstraktion und wird für die Charakterisierung von Metamodellen verwendet. Im Kontext dieser Arbeit wird diese Ebene nicht verwendet und daher nicht weiter behandelt (vgl. [SG06] S.288 ff).

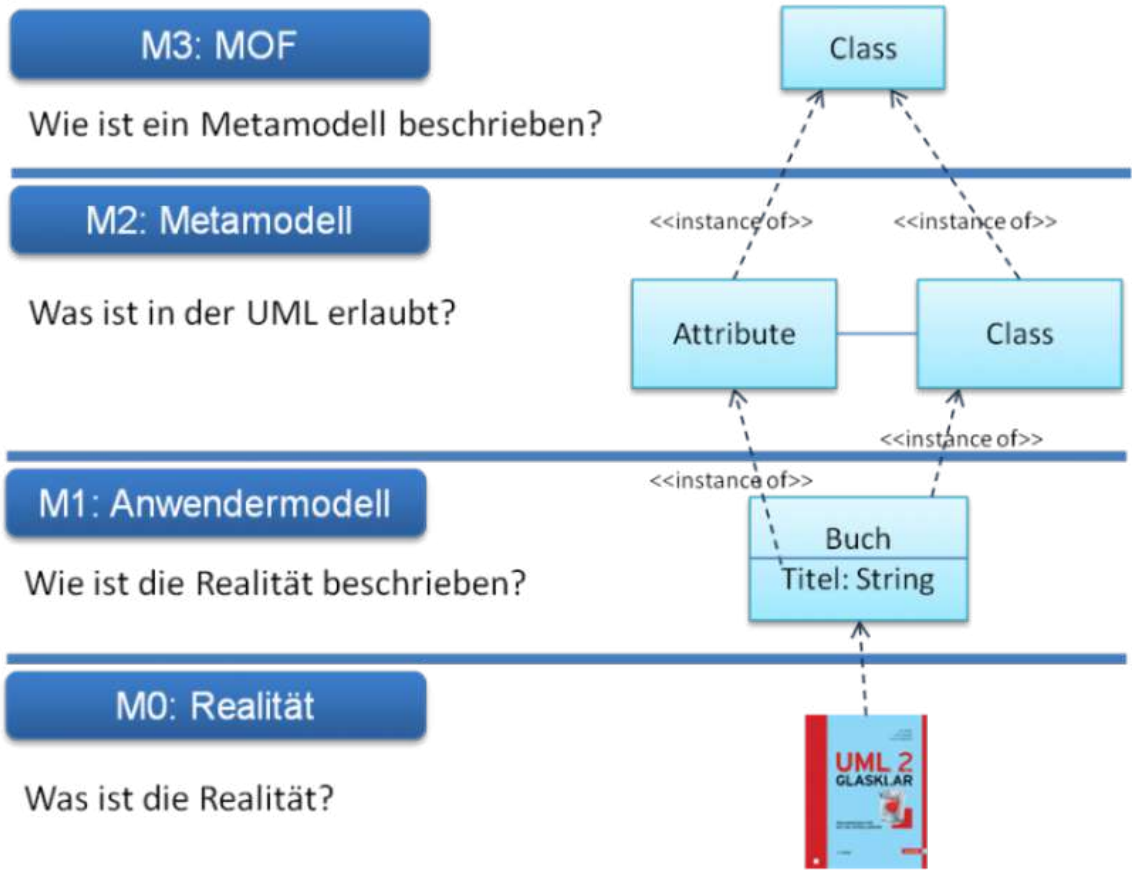


Abbildung 5: Schichten zur Modellierung am Beispiel der UML

Ein Modell des UML-Metamodells wird in dem Kapitel *Evaluation* als Grundlage für eine Überprüfung der getätigten Änderungen verwendet. Das Modell enthält eine Vielzahl unterschiedlicher Elemente und Beziehungen zwischen den einzelnen Elementen. Diese Vielfalt bildet eine hervorragende Basis für eine Evaluation und garantiert eine hohe Abdeckung von möglichen Kombinationen.

## 2.2. USE: An UML-based specification environment

Die Anwendung *UML-based Specification Environment*, kurz USE, wurde 1998 im Rahmen der Promotion von Mark Richters in der Arbeitsgruppe Datenbanksysteme an der Universität Bremen entwickelt und ermöglicht die Modellierung, Validierung und Verifikation von Informationssystemen. USE wird von der genannten Arbeitsgruppe bis heute gepflegt und weiterentwickelt. Im Laufe der Jahre veränderte sich die Anwendung durch zahlreiche wissenschaftliche Arbeiten, die USE mit verschie-

denen Anpassungen und Erweiterungen versahen. Auch im Kontext dieser Arbeit werden die vorhandenen Benutzungsmöglichkeiten der Software angepasst und durch neue Interaktionen ergänzt (vgl. [Ric02], S.1).

USE wird verwendet, um auf Basis einer textuellen Beschreibung eines Modells Klassen- und Objektdiagramme zu erzeugen. Anschließend werden die generierten Darstellungen durch die Verwendung verschiedener Interaktionen verändert. Die vorhandenen Benutzungsmöglichkeiten werden auf ihre Vollständigkeit und Korrektheit untersucht und gegebenenfalls durch neue Aktionen ergänzt. Das Listing *USE-Spezifikation* zeigt einen Ausschnitt aus der Beschreibung des Modells *StudentWorld*. Es werden drei Klassen, eine Assoziation und eine Generalisierung definiert.

```
model StudentWorld

class Person
end

class Student < Person
end

class Course
end

association Attendance between
  Student[*] role attendee
  Course[*] role attended
end

associationclass Tutorial between
  Student[*] role tutor
  Course[*] role tutored
attributes
  salary:Integer
end


aggregation Prerequisite between
  Course[*] role later
  Course[*] role former
end
```

Listing 1: USE-Spezifikation

Nachdem eine Spezifikation über ein Kommandozeilentool oder über die graphische Oberfläche eingelesen und geladen wurde, können verschiedene Diagramme erzeugt werden. Die Abbildung *UML-based Specification Environment* zeigt die Darstellung eines Klassen- und Objektdiagramms in USE. Am linken Rand der Abbil-

dung werden unter anderem die vorhandenen Klassen und Assoziationen aufgelistet. Wird eine davon ausgewählt, wird im darunter liegenden Abschnitt die Beschreibung aus der USE-Spezifikation angezeigt. In diesem Beispiel wird die Beschreibung der Assoziation *Attendance* gezeigt. Im oberen Bereich der Abbildung werden einem Anwender über die Menüs und Buttons verschiedene Möglichkeiten gegeben, mit der geladenen Spezifikation zu interagieren. Die Erzeugung eines Klassen- beziehungsweise Objektdiagramms stellen zwei mögliche Interaktionen dar (vgl. [GBR07] S.27ff).

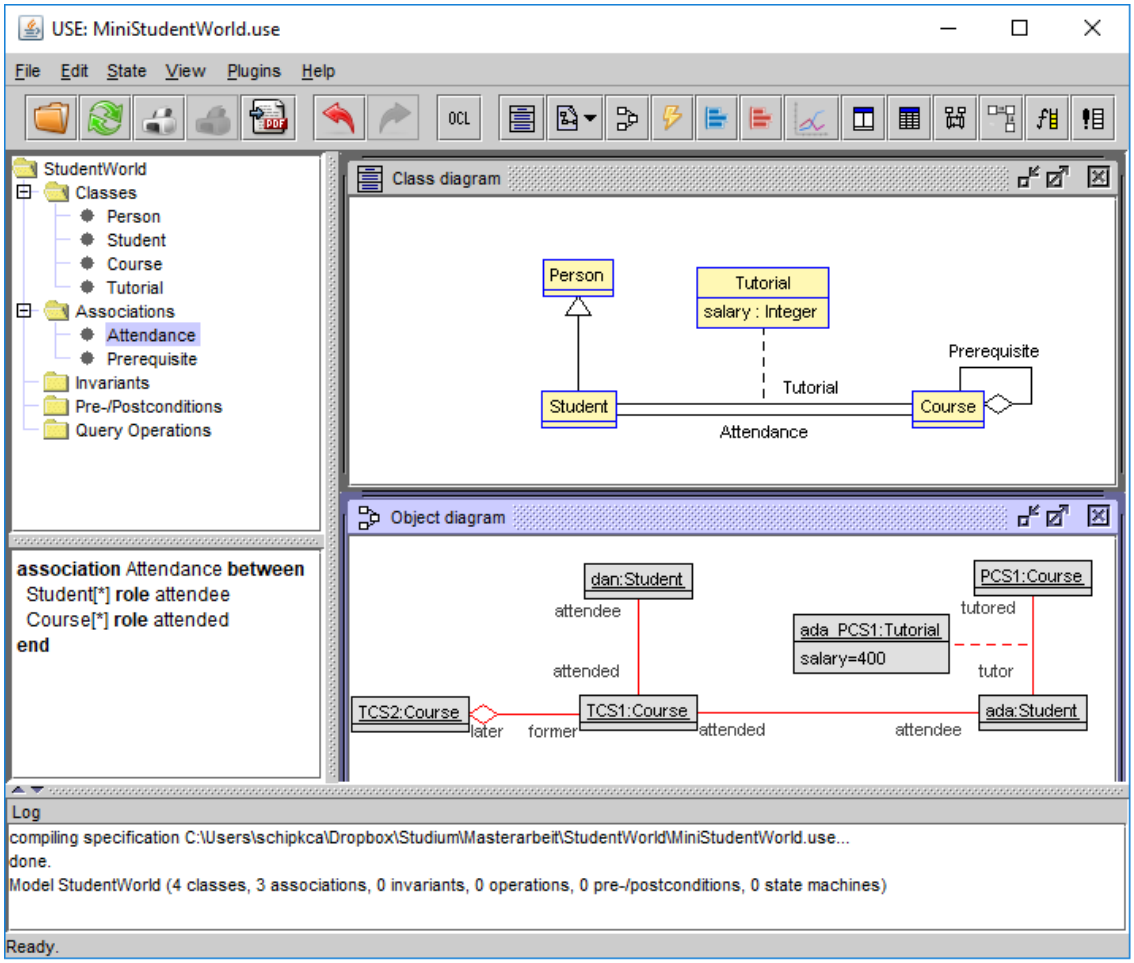


Abbildung 6: UML-based Specification Environment

Die Elemente in einem Objektdiagramm werden durch die Instanziierung der Elemente des zugehörigen Klassendiagramms erzeugt. USE ermöglicht das Erstellen von Instanzen über die graphische Oberfläche, über die Kommandozeile oder durch die Verwendung eines SOIL-Skriptes. SOIL steht für *Simple OCL-like Imperative programming* und dessen Sprachkonstrukute können direkt über die Kommandozeile eingeben oder über ein Skript eingelesen werden. Für das oben dargestellte Objektdiagramm wurde folgendes Skript verwendet (vgl. [BG11], S.3ff).

```
!new Student('ada')
!new Student('dan')
!new Course('PCS1')
!new Course('TCS1')
!insert (dan,TCS1) into Attendance
!insert (ada,TCS1) into Attendance
!ada_PCS1 := new Tutorial('ada_PCS1') between (ada,PCS1)
!ada_PCS1.salary := 400
!new Course('TCS2')
!insert (TCS2,TCS1) into Prerequisite
```

Listing 2: SOIL-Skript

Die vorhandenen und für diese Ausarbeitung relevanten Benutzungsmöglichkeiten von USE werden in dem Kapitel *Analyse der USE-Version* aufgezeigt. Dieser Abschnitt führte das Werkzeug ein und beschrieb auf welche Weise Diagramme erzeugt werden können. Alle nachfolgenden Kapitel setzen nach dem Einlesen eines Modells an und beschreiben primär die Benutzungsmöglichkeiten und die dadurch ausgelösten Änderungen.

# 3. Verwandte Arbeiten

Seit der ersten Veröffentlichung von USE sind Ausarbeitungen entstanden, die unter der Verwendung von USE verschiedene Aspekte der Modellierung, Transformation, Validierung und Verifikation untersuchten. Einige dieser Arbeiten führten zu Erweiterungen und neuen Funktionalitäten, während in andere Arbeiten USE eingesetzt wurde, um verschiedene Szenarien zu modellieren.

Ein Objektdiagramm stellt einen Systemzustand zu genau einem Zeitpunkt dar. Wird der Systemzustand verändert, führt das zu einem neuen Objektdiagramm. Je nach Anzahl der Änderungen in einem System können relativ schnell viele unterschiedliche Systemzustände eintreten, die zu einer Vielzahl von Objektdiagrammen führen. Jedes dieser Objektdiagramme kann einen Vorgänger und Nachfolger haben, ähnlich wie bei einem Film, der aus der Aneinanderreihung von einzelnen Momentaufnahmen besteht. Die entstehende Sequenz von Zuständen ist für die Darstellung eines *lebendigen* Systems geeignet. Um eine übersichtliche Darstellung zu ermöglichen, wurde untersucht, ob die Sequenz von unterschiedlichen Zuständen auch mit einem Objektdiagramm darstellbar ist und wie eine entsprechende Transformation aussieht (vgl. [GHH+14] S.274ff, [HHG14] S.175ff , [DGH17] S.89ff).

In Deutschland wurde das System *Toll Collect* eingeführt, um auf Basis von erfassten Wegpunkten eine Nutzungsgebühr für die Straßeninfrastruktur zu erheben. Für einen Lastkraftwagen wird erfasst, an welchem Wegpunkt der LKW mit der Nutzung der Straßeninfrastruktur begonnen hat und an welchem Wegpunkt die Nutzung endet. Für die Strecke zwischen den beiden Wegpunkten ist eine Gebühr zu entrichten (vgl. [GHHS14] S.32f). Wird die Anzahl der LKWs auf deutschen Straßen und die Anzahl an Autobahnauf- und abfahrten betrachtet, resultiert die Betrachtung in einer großen Anzahl von Objekten. Eine übersichtliche Darstellung ist aber stark abhängig von der Anzahl der modellierten LKWs und Wegpunkte. Diese Problematik kann auf weitere Anwendungsfälle übertragen werden und erfordert Möglichkeiten, die vorhandenen Informationen auf ein geeignetes Maß zu reduzieren, um überschaubare und nutzbare Darstellungen in Form von Diagrammen zu erzeugen.

USE kann nicht nur Objekt- und Klassendiagramme zu einer Spezifikation erzeugen, sondern unterstützt auch Kommunikations- und Sequenzdiagramme. Die Kommunikations- und Sequenzdiagramme können auch eine Fülle von Vorgängen enthalten und abbilden, sodass auch bei diesen Diagrammen Möglichkeiten vorhanden sein müssen, die Darstellung der Diagramme nach den Bedürfnissen des Anwenders zu verändern. Bei diesen Veränderungen wird ausschließlich die Visualisierung angepasst, nicht das Modell oder die vorhandenen Instanzen. Um einem Anwender

eine effizientere Bedienung zu ermöglichen, wurden die Interaktionsmöglichkeiten auf die beiden Diagramme erweitert (vgl. [Sch17] S.2f).

Des Weiteren enthalten beispielsweise die Ausarbeitungen *„Visualizing and Analyzing Discrete Sets with a UML and OCL Software Design Tool“* und *„Squeezer Graphs: Proposing a Simple Basis for Computing (Meta-)Model Features“* Darstellungen, die in der aktuellen USE-Version nicht oder nur umständlich erzeugt werden können. Die Verwendung der Darstellungen deuten auf ein Bedürfnis nach zusätzlichen Auswahl- und Filteroptionen hin und offenbaren bisher ungenutzte Potenziale (vgl. [GD18] S.3f, [Gog18] S.3).

Alle genannten Veröffentlichungen befassen sich mit der Darstellung von Klassen- oder Objektdiagrammen. Die dabei gesammelten Erfahrungen, aufgedeckten Probleme und gefundenen Lösungen können auch für die Beantwortung der Forschungsfragen dieser Arbeit relevant sein und die Gestaltung zukünftiger Interaktionen beeinflussen.

# 4. Analyse der USE-Version

USE bietet verschiedene Benutzeraktionen an, um erzeugte Darstellungen zu verändern. In dieser Arbeit stehen Funktionen im Vordergrund, die eine Änderung der Darstellung ermöglichen ohne das die zugrundeliegende Spezifikation verändert wird. Eine Darstellung ergibt sich aus den vorhandenen Elementen, deren Positionierung und Sichtbarkeit. Die Funktionen, die betrachtet werden, manipulieren ausschließlich die Ansicht und nicht die Existenz von Elementen.

Die Abbildung *Kontextmenü Ansicht Klassendiagramm* zeigt das zentrale Menü, über das Benutzeraktionen initiiert werden. Das Kontextmenü öffnet sich durch die Ausführung eines rechten Mausklicks, wenn sich der Anwender in der Ansicht *Class diagram* oder *Object diagram* befindet. Im weiteren Verlauf dieses Kapitels werden die vorhandenen Aktionen und deren Verhalten beschrieben sowie auf ihre Korrektheit geprüft. Eine weitere zentrale Komponente ist die Möglichkeit Elemente in einem Diagramm selektieren zu können. Eine Selektion kann zu zusätzlichen Aktionen im Kontextmenü führen, die ohne eine Selektion nicht vorhanden wären. Eine Auswahl eines Elements ist durch einen Klick auf das ausgewählte Element möglich. Durch die Verwendung der Umschalttaste können mehrere Elemente markiert werden. Alternativ kann mit dem Cursor ein Rechteck gezogen werden, mit dem Ziel alle Elemente in diesem Rechteck auszuwählen.

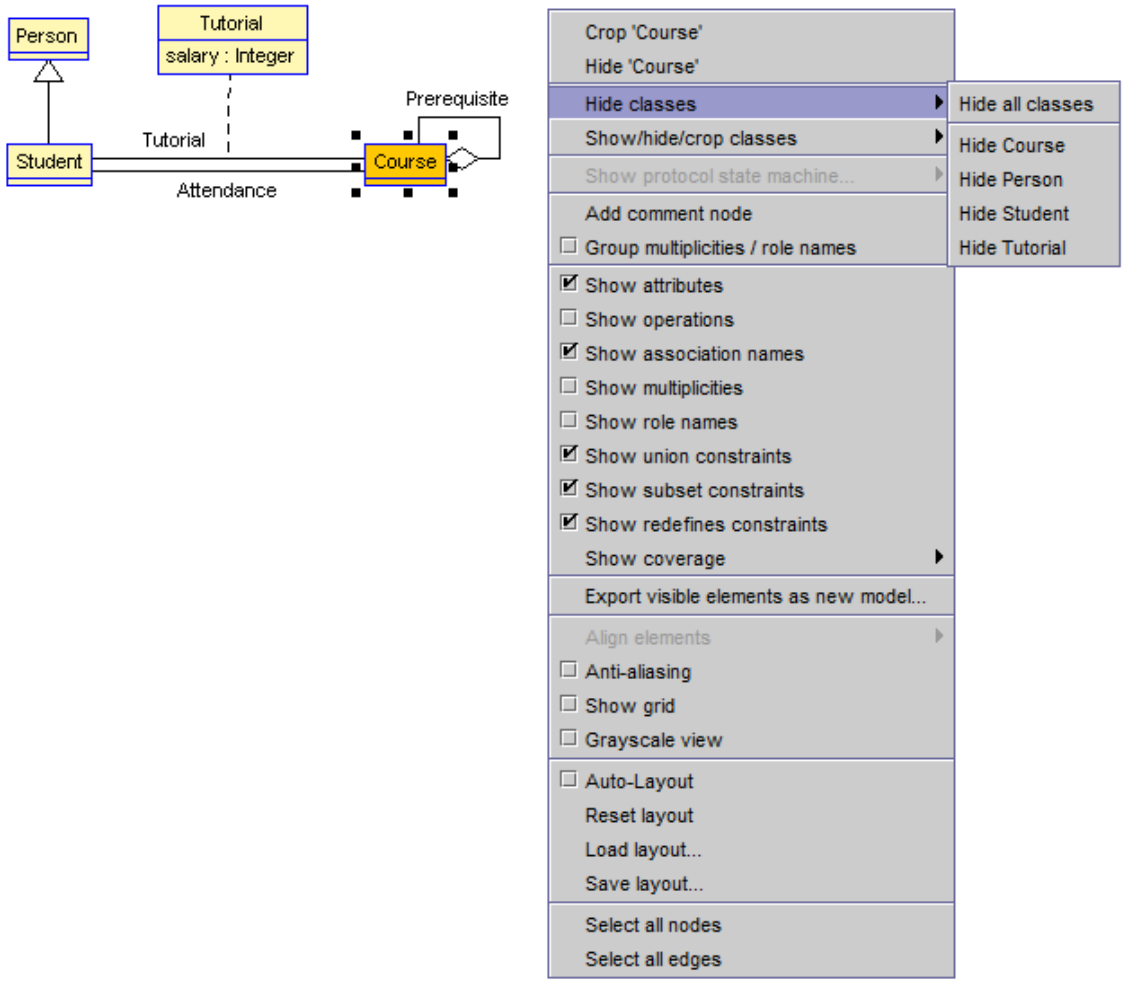


Abbildung 7: Kontextmenü Ansicht Klassendiagramm

In den folgenden Abschnitten wird untersucht, welche Benutzeraktionen einem Anwender angeboten werden und welche Veränderungen in der Darstellung die Benutzeraktionen ermöglichen. Dabei sollen die folgenden Fragestellungen bearbeitet und beantwortet werden, um Aufschlüsse über die vorhandene Auswahl- und Filteroptionen zu erhalten.

- Gibt es ein wiederkehrendes Schema, auf dem die Benutzeraktionen basieren?
- Sollte ein wiederkehrendes Schema vorhanden sein, ist das Schema für einzelne Konzepte wie zum Beispiel Klassen oder Assoziation vollständig und einheitlich umgesetzt?
- Gibt es Benutzeraktionen, die in der Ansicht Klassendiagramm vorhanden sind, aber in der Ansicht Objektdiagramm nicht unterstützt werden?
- Gibt es Benutzeraktionen, die in der Ansicht Objektdiagramm vorhanden sind, aber in der Ansicht Klassendiagramm nicht unterstützt werden?
- Können die vorhandenen Benutzeraktionen durch Änderungen verbessert werden?

- Ist es erforderlich, neue Benutzeraktionen einzuführen, um eine effektivere Manipulation von Darstellungen zu ermöglichen?

Die Analyse basiert auf der USE-Version 5.0.1[4] und Ausschnitte aus dem Beispiel der Studentenwelt werden als Szenario verwendet. Die verwendete USE-Spezifikation ist dem Anhang zu entnehmen.

## 4.1. Basisoperationen

Als Basisoperation wird in dieser Ausarbeitung eine Operation bezeichnet, die die Grundlage für eine Vielzahl unterschiedlicher Interaktionen bildet. In USE werden drei Basisoperationen verwendet, auf denen viele Benutzeraktionen beruhen. Bei den drei Basisoperationen, im späteren Verlauf auch als drei Aktionen bezeichnet, handelt es sich um die Funktionen *hide*, *show* und *crop*. USE bietet beispielsweise die Möglichkeit diese Aktionen auf Klassen anzuwenden, wodurch die Menge der angezeigten Elemente verändert wird und eine neue Darstellung entsteht. Alle drei Operationen verändern ausschließlich die Darstellung und nicht die Datenhaltung. Die Wirkung der Aktionen kann rückgängig gemacht werden.

Die Basisoperation *hide* ermöglicht das Verstecken von Elementen in einer Darstellung. Die Aktion kann auf ein Element oder eine Vielzahl von Elementen ausgeführt werden. Die Auswirkung der Operation hängt von dem zu versteckenden Element ab. Betroffene Elemente werden aus der Ansicht entfernt, sind jedoch weiterhin in der Datenhaltung vorhanden, sodass versteckte Elemente auch wieder sichtbar gemacht werden können. Die Aktion *hide* benötigt die Angabe von mindestens einem Element, auf das die Aktion ausgeführt werden kann. Prinzipiell kann *hide* auch auf ein bereits verborgenes Element ausgeführt werden. Allerdings wäre die Auswahl eines bereits versteckten Elements über die Darstellung problematisch, da es in der Darstellung nicht vorhanden und somit auch nicht ausgewählt werden kann. Des Weiteren führt ein *hide* auf ein bereits verstecktes Element zu keiner Veränderung in der Darstellung. Durch diese Aktion entsteht kein Mehrwert, weshalb dieser Anwendungsfall nicht weiter betrachtet wird.

Die Auswirkung der Funktion kann durch zwei Mengen beschrieben werden. Eine Menge umfasst alle Elemente, die in der Darstellung sichtbar sind. Diese Menge wird als *visibleData* bezeichnet. Nach dem Laden einer Spezifikation enthält diese Menge alle Elemente der Darstellung. Durch die drei Basisoperationen wird die Menge verändert. Je nach Aktion werden Elemente aus der Menge entfernt oder hinzugefügt. Die Aktion *hide* entfernt Elemente aus der Menge der sichtbaren Elemente und fügt

[4] `https://sourceforge.net/projects/useocl/ (zuletzt aufgerufen am 27.12.2018)`

diese Elemente einer zweiten Menge hinzu, der Menge der versteckten Elemente. Die zweite Menge wird auch als *hiddenData* bezeichnet und enthält alle Elemente, die in der aktuellen Darstellung nicht sichtbar sind. Ein Element kann immer nur in einer der beiden Mengen vorhanden sein.

Bei dem Verstecken von Elementen ist zwischen zwei Anwendungsfällen zu unterscheiden, die am Beispiel von Klassen und Assoziationen verdeutlicht werden. Die Darstellung verändert sich unterschiedlich, wenn ein *hide* auf eine Klasse ohne verbundene Assoziationen oder auf eine Klasse mit verbundener Assoziationen ausgelöst wird. Im einfachen Fall verfügt eine Klasse über keine ein- oder ausgehenden Kanten. Folglich steht die Klasse alleine und losgelöst in der Darstellung. Die Klasse wird aus der Menge *visibleData* entfernt und der Menge *hiddenData* hinzugefügt. Die Darstellung verwendet nur Elemente aus *visibleData*, sodass die Klasse nicht mehr angezeigt wird. Im zweiten Fall verfügt eine zu versteckende Klasse über eine verbundene Assoziation. Das Verbergen der Klasse führt zu einer Darstellung, bei der die verbundene Assoziation nicht an einer Klasse endet. Mindestens ein Endpunkt der Assoziation wäre nicht sichtbar und die daraus resultierende Darstellung bildet einen ungültigen Zustand ab. Zur Vermeidung einer fehlerhaften Darstellung ist in diesem Fall auch die Assoziation zu verstecken. Die Abbildung *Basisoperation hide* verdeutlicht den zweiten Fall. Es wurde die Operation *hide* auf die Klasse *Course* ausgeführt, die mit drei Assoziationen verbunden ist. Durch das Entfernen der Klasse aus der Menge der sichtbaren Elemente fehlt bei den drei Assoziationen mindestens ein Endpunkt. Die Aggregation *Prerequisite* in der Abbildung ist reflexiv auf die Klasse *Course* gebunden. Nach dem Entfernen der Klasse fehlen für diese Beziehung beide Endpunkte. Das ist ein ungültiger Zustand, weshalb auch die Aggregation *Prerequisite* versteckt wird. Bei den beiden anderen Assoziationen ist durch die verborgene Klasse *Course* ein Endpunkt der Verbindung nicht sichtbar. Folglich werden auch die beiden Assoziationen sowie die Assoziationsklasse *Tutorial* aus der Ansicht entfernt. Die Ausführung von *hide Course* hat die Menge der dargestellten Elemente von vier Klassen, drei Assoziationen und einer Generalisierung auf zwei Klassen und eine Generalisierung reduziert. Beim zweiten Fall werden immer mehrere Elemente versteckt, obwohl die Operation nur auf ein Element ausgeführt wird. Das Beispiel ist auf andere Elemente, wie Objekte und Links, übertragbar.

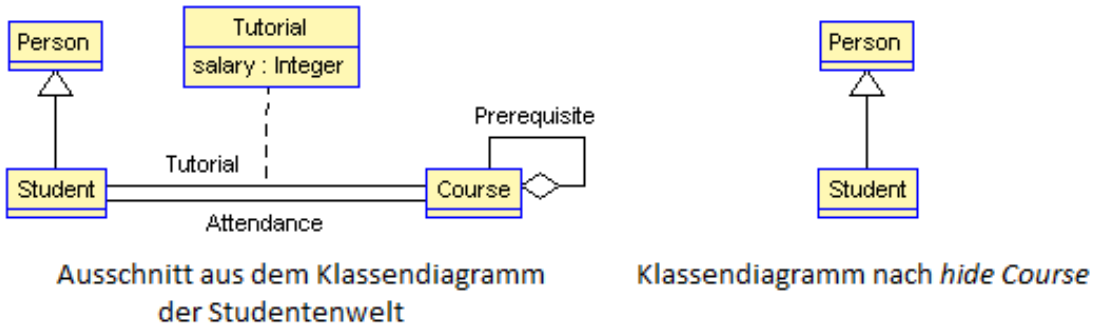


Abbildung 8: Basisoperation *hide*

Die zweite Basisoperation wird als *show* bezeichnet und kehrt die Wirkung von *hide* um. Versteckte Elemente werden durch *show* wieder in die Darstellung aufgenommen und *show* bildet das Komplement zu *hide*. Die Aktion kann auf ein oder mehrere Elemente ausgeführt werden. Dabei werden die betroffenen Elemente aus der Menge *hiddenData* entfernt und der Menge *visibleData* hinzugefügt. Damit die Aktion *show* eine Wirkung entfalten kann, sind Elemente in der Menge *hiddenData* erforderlich. Die Ausführung der Operation auf ein bereits sichtbares Element führt zu keiner Veränderung in der Darstellung und daher wird dieser Fall auch nicht weiter betrachtet.

Das Ergebnis der Anwendung der Aktion ist die vollständige Umkehrung der Wirkung von *hide*. Alle durch *hide* versteckten Elemente werden wieder sichtbar. Beim Verstecken von Elementen wurde zwischen zwei Fällen unterschieden, sodass die beiden Fälle auch bei der Ausführung von *show* beachtet werden müssen. Zur besseren Vergleichbarkeit werden wieder Klassen und Assoziationen zur Erklärung verwendet. Wird die Operation *show* auf eine versteckte Klasse angewendet, die über keine verbundenen Assoziationen verfügt, wird nur die Klasse wieder in die Darstellung aufgenommen. Dieses Verhalten kehrt das Verhalten aus dem ersten Fall der Aktion *hide* um. Im zweiten Fall wurden mehrere Elemente versteckt und die ursprüngliche Darstellung sollte durch die Ausführung von *show Course* wieder hergestellt werden. Dabei wird im ersten Schritt die Klasse *Course* sichtbar. Anschließend wird geprüft, ob zusätzlich Assoziationen angezeigt werden können. Aus der Mengen der versteckten Elemente werden bei der Überprüfung alle versteckten Assoziationen betrachtet und auf dieser Menge wird untersucht, ob Assoziationen vorhanden sind, deren Endpunkte sichtbar sind. Ist eine Assoziation gefunden, deren Endpunkte sichtbar sind, ist die Assoziation ebenfalls anzuzeigen. In der Abbildung wurde die Basisoperation *show* auf die Klasse *Course* ausgeführt. Durch die Anzeige der Klasse sind auch die Bedingungen für die Anzeige der versteckten Assoziationen *Attendance*, *Tutorial* und der Aggregation *Prerequisite* erfüllt und diese werden in die Darstellung aufge-

nommen. Die Aktion *show* kann mehrere Elemente wieder sichtbar werden lassen, auch wenn die Elemente nicht explizit angegeben wurden.

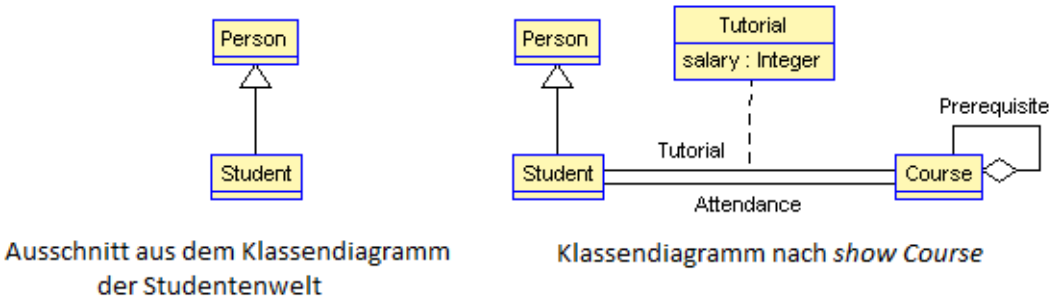


Abbildung 9: Basisoperation *show*

Die dritte Operation zur Veränderung der Darstellung ist *crop*. Die Aktion kann auf ein oder mehrere Elemente angewandt werden, benötigt aber sichtbare Elemente als Parameter. Die Elemente, auf die *crop* ausgeführt wird, bleiben in der Darstellung erhalten, während alle anderen Elemente versteckt werden. Dieses Verhalten entspricht einer Kombination von *hide* und *show*. Die gleiche Darstellung kann erzeugt werden, wenn der Anwender alle Elemente ausblendet und danach das gewünschte Element sichtbar werden lässt. Die Aktion verändert die beiden definierten Mengen, indem alle nicht selektierten Elemente in die Menge *hiddenData* überführt und damit aus der Darstellung entfernt werden.

Am Beispiel von Klassen und Assoziationen wird in der Abbildung *Basisoperation crop* das Verhalten der Aktion verdeutlicht. Die Aktion *crop* wird auf die Klasse *Course* ausgeführt, wodurch alle Elemente außer *Course* versteckt werden. Die Klassen *Person*, *Student* und *Tutorial* sowie die Assoziationen *Attendance*, *Tutorial*, *Prerequisite* und die Generalisierung zwischen den Klassen *Person* und *Student* wären auszublenden. Dabei ist *Prerequisite* genauer zu betrachten. Es handelt sich um eine reflexive Beziehung, deren beiden Enden mit der sichtbaren Klasse *Course* in Verbindung stehen. Bei der Operation *show* wurde beschrieben, dass Assoziationen sichtbar werden beziehungsweise sichtbar bleiben, wenn alle Endpunkte sichtbar sind und die Operation nicht direkt auf eine Assoziation ausgeführt wird. Nach der Ausführung von *crop* sind alle Endpunkte von *Prerequisite* sichtbar und *Prerequisite* bleibt ebenfalls sichtbar. Die verborgenen Elemente können durch *show* einzeln wieder in die Darstellung aufgenommen werden. Eine direkte Umkehrung der Operation, wie bei *hide* und *show*, ist nicht gegeben.

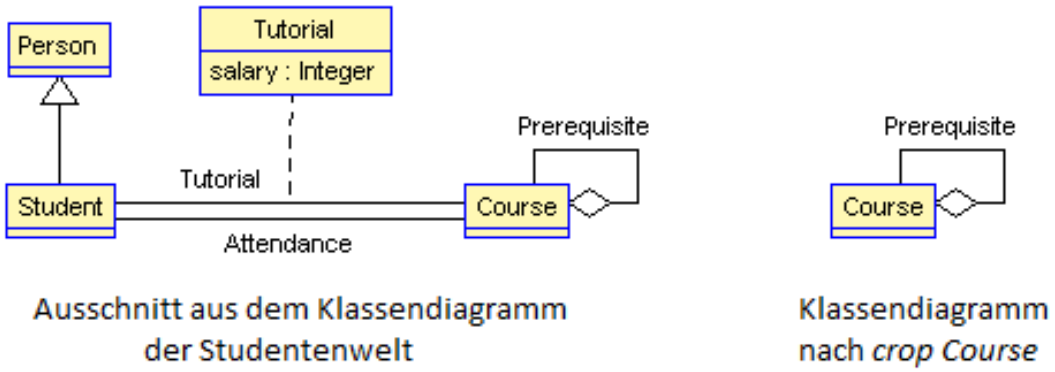


Abbildung 10: Basisoperation *crop*

Die Tabelle *Auswirkungen der Basisoperationen* fasst das Verhalten der drei Operationen zusammen und beschreibt die Auswirkungen der jeweiligen Aktion auf die Elemente in der Darstellung. Das Verhalten der Ausführung von *hide* auf ein bereits verstecktes Element beziehungsweise *show* auf sichtbare Elemente wird vermerkt, im weiteren Verlauf aber nicht weiter betrachtet. Ähnlich verhält es sich bei der Aktion *crop* auf verborgene Elemente.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **sichtbares Element** | Element wird versteckt | Element bleibt sichtbar | alle nicht ausgewählten Elemente werden versteckt |
| **verstecktes Element** | Element bleibt versteckt | Element wird sichtbar | Element wird sichtbar werden |

Tabelle 2: Auswirkungen der Basisoperationen

## 4.2. Ansicht Klassendiagramm

Im Zuge der Analyse soll die Frage beantwortet werden, ob die Basisoperationen einheitlich und vollständig auf die zentralen Konzepte eines Klassendiagramms umgesetzt wurden. In diesem Abschnitt wird untersucht, in wie weit die Operationen auf Klassen, Assoziationen und Generalisierungen ausgeführt werden können und ob Unterschiede in der Bedienung der entsprechenden Funktionen vorliegen. Wenn in diesem Abschnitt von *Aktionen im Klassendiagramm* gesprochen wird, sind die Basisoperationen in der Ansicht Klassendiagramm gemeint.

### 4.2.1. Basisoperationen auf Klassen

Als Erstes wird untersucht, ob die drei Basisoperationen auf Klassen angewendet werden können und wenn ja, welche Möglichkeiten zur Initiierung einer solchen Aktion gegeben sind beziehungsweise ob Voraussetzung oder Bedingungen erfüllt sein

müssen.

Das Verstecken von Klassen ist über das Kontextmenü möglich. Wenn die Darstellung mindestens eine sichtbare Klasse enthält, ist im Kontextmenü das Menü *Hide classes* vorhanden, wie in der Abbildung *Kontextmenü Ansicht Klassendiagramm* zu erkennen ist. Das Menü bietet die Möglichkeit alle Klassen oder gezielt eine einzelne Klasse unter Verwendung des Klassennamens auszublenden. Eine weitere Möglichkeit eine Klasse zu verstecken, wird über das Kontextmenü angeboten, wenn mindestens eine Klasse selektiert wurde. Die Selektion einer Klasse führt zur Erweiterung des Kontextmenüs um die Option *Hide <Klassenname>*. In der referenzierten Abbildung ist die Klasse *Course* selektiert und das Kontextmenü bietet die Option *Hide Course* an. Für die Operation *hide* wurden keine Abweichung zum beschriebenen Verhalten der Basisoperation festgestellt.

Die Basisoperation *show* auf Klasse wird über das Menü *Show classes* im Kontextmenü ermöglicht. Das Menü ist nur verfügbar, wenn mindestens eine versteckte Klasse vorhanden ist. Die Gestaltung orientiert sich an dem Menü *Hide classes*. Es ist eine Option vorhanden, die das Einblenden aller versteckten Klassen oder das gezielte Einblenden einer Klasse unter Verwendung des Klassennamen ermöglicht. Die Operation *show* verhält sich erwartungsgemäß.

*Crop* kann auf eine Klasse ausgeführt werden, nachdem eine Klasse selektiert wurde. Das Kontextmenü wird um die Option *Crop <Klassenname>* erweitert. Die Abbildung *Kontextmenü Ansicht Klassendiagramm* bietet für die Klasse *Course* eine *crop*-Aktion an. Das Verhalten entspricht den Erwartungen, wenn die Aktion auf eine Klasse ausgeführt wird, die keine Assoziationsklasse ist. Die Anwendung von *crop* auf eine Assoziationsklasse führt zu einer Darstellung ohne Elemente. Alle sichtbaren Elemente werden aus der Menge *visibleData* entfernt und der Menge *hiddenData* hinzugefügt. Der *Fehlerfall Crop < Assoziationsklasse >* in der gleichnamigen Abbildung verdeutlicht die Unterschiede zwischen dem aktuellen und dem erwarteten Ergebnis. Bei der Basisoperation *crop* soll nach der Ausführung mindestens das Element, auf das die Aktion ausgeführt wird, sichtbar sein. Im vorliegenden Fall ist keine Klasse mehr sichtbar und das Verhalten entspricht nicht den Erwartungen. Erwartet wurde eine Darstellung mit den Klassen *Student*, *Course* und *Tutorial* sowie den Assoziationen *Attendance*, *Tutorial* und *Prerequisite*.

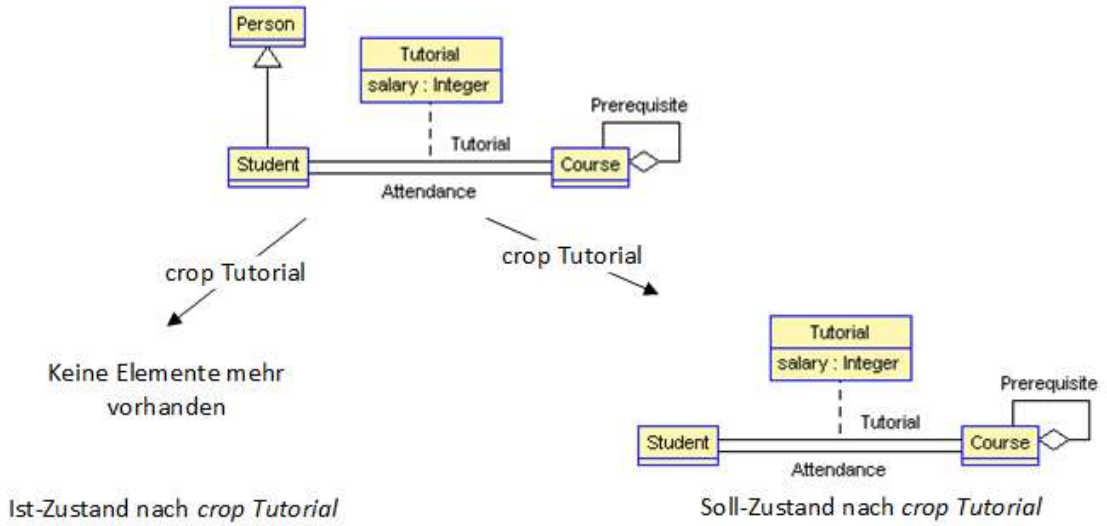


Abbildung 11: Fehlerfall *Crop* < *Assoziationsklasse* >

Wie entsteht diese korrekte Darstellung? Auf die Assoziationsklasse *Tutorial* wird die Aktion angewendet, wodurch diese Klasse in der Menge der sichtbaren Elemente bleibt. Alle anderen Elementen müssten ausgeblendet werden, es sei denn, die Elemente werden für die korrekte Anzeige der Assoziationsklasse benötigt. Die Assoziationsklasse gehört zu der gleichnamigen Assoziation und die Assoziation steht in Verbindung mit den Klassen *Student* und *Course*. Für die Darstellung der Assoziation und der zugehörigen Assoziationsklasse werden die genannten Elemente benötigt. *Attendance* und *Prerequisite* werden zusätzlich angezeigt, da alle Endpunkte der Assoziationen in der Darstellung vorhanden sind. Dieses Verhalten ist an dem Verhalten von *show* angelehnt und entspricht dem beschriebenen Verhalten im Abschnitt *Basisoperationen*.

Zusammenfasst sind alle Basisoperationen auf Klassen implementiert und das Verhalten entspricht den Erwartungen. Allein die Anwendung von *crop* auf Assoziationsklassen bildet eine Ausnahme. Die Implementierung der drei Aktionen auf Klassen wird als Referenzpunkt für eine vollständige Umsetzung angesehen und die Benutzungsmöglichkeiten der Aktionen auf Assoziationen und Generalisierungen werden mit diesem Zustand verglichen.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Klasse** | vorhanden | vorhanden | mit Einschränkungen |

Tabelle 3: Basisoperationen auf Klassen

### 4.2.2. Basisoperationen auf Assoziationen

Der Abschnitt *Basisoperationen auf Assoziationen* betrachtet, welche Basisoperation auf Assoziationen angewendet werden können und ob das Verhalten der Operationen korrekt implementiert ist. Des Weiteren wird darauf geachtet, ob Unterschiede zwischen den Aktionen auf Klassen und denen auf Assoziationen vorhanden sind und wenn ja, in welcher Weise diese Abweichungen eine Auswirkung haben. Im besten Fall können keine Abweichungen festgestellt werden, was einer vollständigen und einheitlichen Umsetzung der Operationen entspricht.

Das Verstecken von Assoziationen ist über das Kontextmenü möglich. Damit die Aktion *hide* einem Anwender angeboten werden kann, muss mindestens eine sichtbare Assoziation in der Darstellung vorhanden sein. Im Gegensatz zu den Klassen enthält das Kontextmenü kein Untermenü, das ein Verstecken von Assoziation ohne eine vorherige Selektion ermöglicht. Ein Menü mit der Bezeichnung *Hide associations* ist nicht vorhanden. Wird eine Assoziation gezielt selektiert, wird das Kontextmenü um die Aktion *Hide <Name der Assoziation>* erweitert. Das Verhalten von *hide* entspricht den Erwartungen und es konnte keine Abweichung festgestellt werden. Ausgewählte Assoziationen werden verborgen, während die zugehörigen Klassen weiterhin sichtbar bleiben. Bei der Ausführung auf eine Assoziation mit Assoziationsklasse wird die Assoziationsklasse ebenfalls ausgeblendet. Bei der Aktion *show* ist zu prüfen, ob diese das Verhalten umkehrt.

Die Operation *show* soll verborgene Assoziationen wieder sichtbar werden lassen. Die Selektion von versteckten Elementen ist nicht möglich, weshalb über das Kontextmenü das Untermenü *Show associations* bereitgestellt wird. Dieses Menü sollte nur angeboten werden, wenn eine Assoziation in der Menge *hiddenData* vorhanden ist. Die Abbildung *Fehlerfall Show associations* zeigt das Menü *Show associations* ohne Einträge an. In dem Szenario wurde die Klasse *Person* versteckt. Die Generalisierung zwischen den Klassen *Person* und *Student* kann nicht mehr korrekt dargestellt werden und wird folglich verborgen. Die verborgene Generalisierung führt zur Anzeige des *Show*-Menüs, obwohl eine Generalisierung anstatt einer Assoziation ausgeblendet wurde.

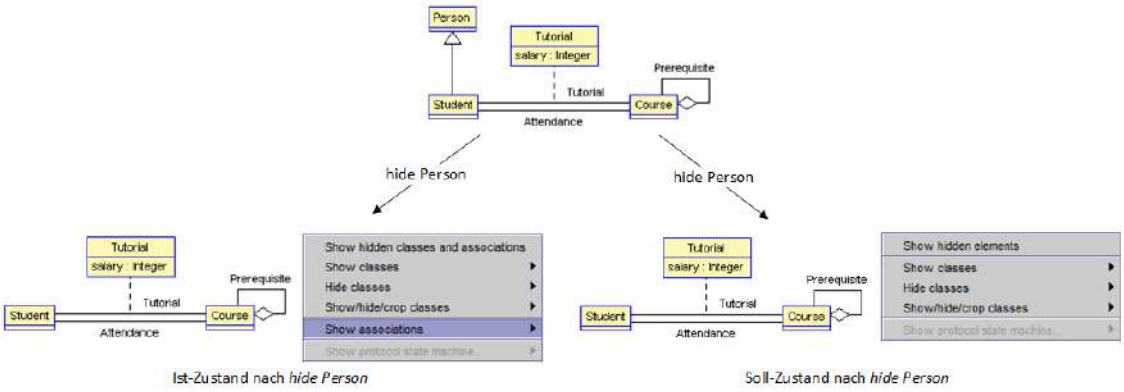


Abbildung 12: Fehlerfall *Show associations*

Die Anwendung von *show* auf eine versteckte Assoziation mit Assoziationsklasse entspricht nicht der erwarteten Darstellung, da die entsprechende *hide*-Aktion nicht umgekehrt wurde und eine Verbindungslinie nicht an einem Objekt endet. Die Abbildung *Fehlerfall Show < Assoziation > mit Assoziationsklasse* verdeutlicht das Fehlverhalten und zeigt die Unterschiede zwischen dem Ist- und dem Soll-Zustand. Die zugehörige Assoziationsklasse wird nicht in die Menge der sichtbaren Elemente überführt, wodurch die Verbindungslinie zu der Assoziationsklasse an einer unsichtbaren Klasse endet.

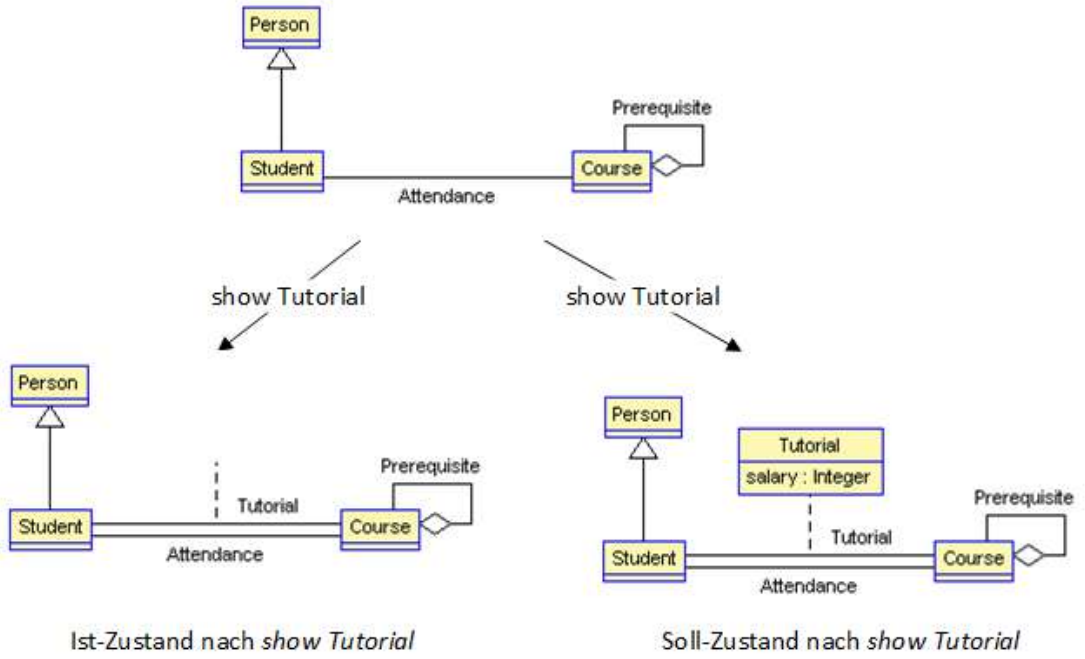


Abbildung 13: Fehlerfall *Show < Assoziation > mit Assoziationsklasse*

Das Verhalten von *show* auf eine Assoziation ohne Assoziationsklasse entspricht den Erwartungen. Für diesen Anwendungsfall konnten keine Abweichungen zu den beschriebenen Verhalten im Abschnitt *Basisoperationen* festgestellt werden.

Die Aktion *crop* ist in der aktuellen USE-Version nur auf Klassen anwendbar. Wird eine Assoziation selektiert, wird das Kontextmenü um die Aktion *hide <Name der Assoziation>* erweitert. Der Eintrag *crop <Assoziation>* ist jedoch nicht vorhanden, wodurch sich die Anwendung der Operation auf Klassen und Assoziationen unterscheidet.

Zusammengefasst sind nur zwei von drei Basisoperationen auf Assoziationen anwendbar, da für die Operation *crop* ein entsprechender Eintrag im Kontextmenü fehlt. Bei den zwei vorhandenen Operationen konnten Abweichungen im Verhalten oder bei der Menügestaltung festgestellt werden. Einige Menüeinträge sind nicht vorhanden oder werden fälschlicherweise angezeigt. Des Weiteren führt eine Störung im Verhalten zu einer fehlerhaften Darstellung. Der Vergleich der Basisoperation auf Klassen und Assoziationen zeigt Unterschiede auf, die in einer abweichenden Bedienung resultieren. Daher ist weder die Vollständigkeit noch die Einheitlichkeit gegeben.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Assoziation** | mit Einschränkungen | mit Einschränkungen | nicht vorhanden |

Tabelle 4: Basisoperationen auf Assoziationen

### 4.2.3. Basisoperationen auf Generalisierungen

In dem Kapitel *Grundlagen* wurde das Konzept der Generalisierung für Klassendiagramme beschrieben. In der aktuellen Version von USE können Generalisierungen selektiert werden, eine Erweiterung oder Anpassung des Kontextmenüs erfolgt daraufhin nicht. Es existiert auch kein Menü *Hide generalizations* oder *Show generalizations*. Folglich können auf Generalisierungen keine der drei Basisoperationen ausgeführt werden.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Generalisierung** | nicht vorhanden | nicht vorhanden | nicht vorhanden |

Tabelle 5: Basisoperationen auf Generalisierungen

## 4.3. Ansicht Objektdiagramm

Nach der Untersuchung, welche Basisoperationen in der Ansicht Klassendiagramm vorhanden sind, wird in diesem Abschnitt die Anwendung der Operationen auf Objekte und Links betrachtet. Wenn im Verlauf dieses Abschnitts von *Aktionen im Objektdiagramm* gesprochen wird, sind Basisoperationen in der Ansicht Objektdiagramm gemeint.

### 4.3.1. Basisoperationen auf Objekte

Ein Objekt kann als Instanz einer Klasse angesehen werden und auf Klassen konnten alle drei Basisoperationen ausgeführt werden. Auf Grund der Vergleichbarkeit von Klassen und Objekten wäre eine vollständige Implementierung und einheitliche Bedienung wünschenswert.

Das Kontextmenü bietet das Menü *Hide object* an, wenn in der Darstellung mindestens ein sichtbares Objekt vorhanden ist. Über dieses Menü können alle sichtbaren Objekte, alle sichtbaren Objekte einer Klasse oder gezielt ein Objekt versteckt werden. Das Menü besteht aus zwei Ebenen. Auf der ersten Ebene hat ein Anwender die Möglichkeit alle sichtbaren Objekte zu verstecken oder eine Klasse auszuwählen. Es werden alle Klassen aufgelistet, die mindestens eine sichtbare Instanz in der Darstellung aufweisen. Wird der Mauszeiger auf den Namen einer Klasse geführt, wird die zweite Ebene angezeigt. Auf dieser Ebene können alle sichtbaren Objekte der Klasse oder genau ein Objekt versteckt werden. Die vorhandenen Objekte einer Klasse werden sortiert aufgelistet. Die Abbildung *Kontextmenü Hide object* stellt den Aufbau des Menüs anhand eines Beispiels dar.

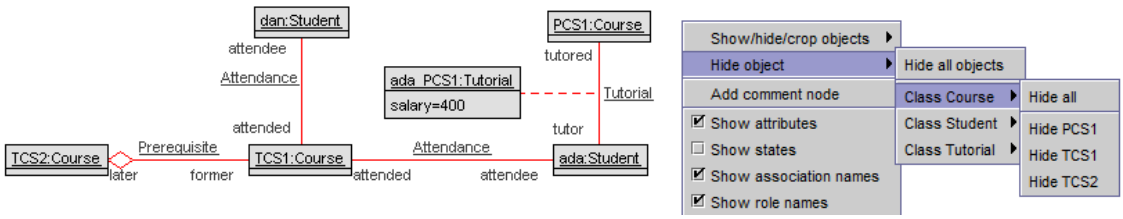


Abbildung 14: Kontextmenü *Hide object*

Nach der Selektion eines Objekts wird das Kontextmenü um den Eintrag *Hide <Objektname>* erweitert. Folglich kann die Operation *hide* auf identischer Weise auf Klassen und Objekten ausgeführt werden. Es wurden keine Abweichungen im Verhalten der Aktion und keine Unterschiede in der Bedienung zwischen Klassen und Objekten festgestellt.

Die Umkehrung der Wirkung von *hide* kann durch die Operation *show* initiiert werden. Ist mindestens ein verstecktes Objekt vorhanden, wird das Kontextmenü

um das Menü *Show object* erweitert. Der Aufbau dieses Menüs ist mit dem Aufbau des Menüs *Hide object* identisch. Der einzige Unterschied ist, dass die Basisoperation *show* anstatt *hide* verwendet wird. Daher können über das Menü alle versteckten Objekte, alle versteckten Objekte einer Klasse oder gezielt ein Objekt wieder sichtbar gemacht werden. Das Verhalten der Aktion *show* auf Objekte weicht nicht von dem beschriebenen Verhalten der Basisoperation ab.

Das Kontextmenü wird nach der Auswahl eines Objekts der Eintrag *Crop <Objektname>* hinzugefügt. Die Erweiterung gleicht den Operationen *hide* auf Objekte oder *hide* und *crop* auf Klassen. Bei dem Verhalten von *crop* auf Objekte wurde mit einer Ausnahme kein Fehlverhalten festgestellt. Wird die Aktion auf ein Linkobjekt angewendet, sind nach der Ausführung keine Elemente in der Darstellung vorhanden. Alle vorher sichtbaren Elemente befinden sich in der Menge *hiddenData*. Die Abbildung *Fehlerfall Crop < Linkobjekt >* verdeutlicht die fehlerhafte Veränderung der Darstellung. Ein *crop* auf das Linkobjekt *ada_PCS1* sollte das Linkobjekt, den benötigten Link sowie die beiden Objekte *PCS1* und *ada* sichtbar lassen. Es liegt eine Abweichung im Verhalten der Basisoperation vor, das auf eine fehlerhafte Implementierung hindeutet.

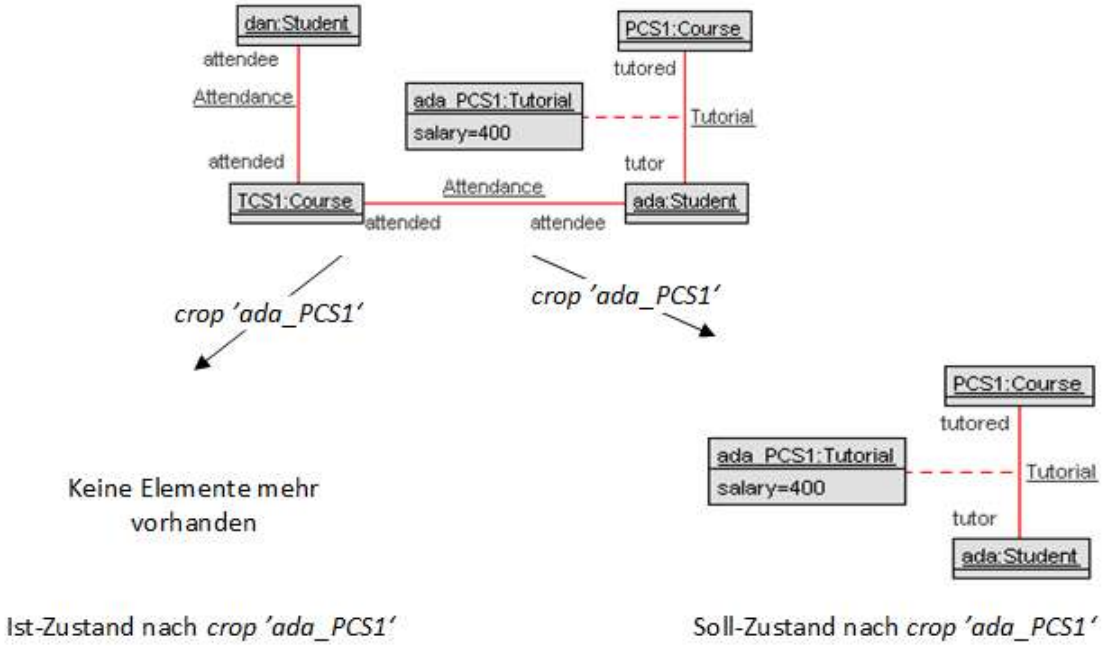


Abbildung 15: Fehlerfall *Crop < Linkobjekt >*

Insgesamt sind die Basisoperationen auf Objekte vollständig vorhanden. Der Fehler bei der Anwendung von *crop* auf Linkobjekte könnte auf Grund der Ähnlichkeiten zu dem Fehlerbild bei der Anwendung von *crop* auf Assoziationsklassen die gleiche Ursache haben. Dies ist im weiteren Verlauf zu untersuchen. Ansonsten können alle Basisoperationen auf Objekte verwendet werden und deren Bedienung ähnelt

der Verwendung von bereits vorgestellten Benutzungsmöglichkeiten. Das stärkt die Wiedererkennung von Interaktionen und resultiert in einer effizienteren und verständlicheren Benutzbarkeit.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Objekt** | vorhanden | vorhanden | mit Einschränkungen |

Tabelle 6: Basisoperationen auf Objekte

### 4.3.2. Basisoperationen auf Links

Nach den Objekten wird nun die Verfügbarkeit und Auswirkung der Basisoperationen auf Links untersucht. Zusätzlich wird betrachtet, ob Unterschiede zu den Aktionen auf Assoziationen und Objekten vorhanden sind. Da ein Link eine Instanz einer Assoziation ist, wäre eine einheitliche Handhabung erstrebenswert.

Das Verstecken von Elementen war bei vielen Konzepten über zwei Arten möglich. Das Kontextmenü wurde um ein Menü erweitert, wenn mindestens ein sichtbares Element vorhanden war, oder das Menü wurde bei einer Selektion um Optionen auf die ausgewählten Elemente erweitert. Ein Menü *Hide links* ist in der aktuellen USE-Version nicht vorhanden. Folglich gibt es keine einfache Möglichkeit alle Links oder alle Links einer Assoziation zu verstecken. Nach der Selektion eines Links wird das Kontextmenü um einen Eintrag erweitert, der das Verstecken des ausgewählten Links ermöglicht. Bei dem Verhalten der Aktion *hide* konnten keine Abweichung festgestellt und keine ungültigen Zustände erzeugt werden.

Die Basisoperation *show* auf Links steht einem Anwender nicht zur Verfügung. Das Kontextmenü enthält kein Menü, dass die Aktion *show* auf versteckte Links ermöglicht. Die Umkehrung der Operation *hide* ist umständlich. Versteckte Links werden nur wieder sichtbar, wenn *show* auf die Objekte eines Links ausgeführt wird. Die Operation überführt Objekte in die Menge der sichtbaren Elemente. Anschließend wird geprüft, ob versteckte Links vorhanden sind, bei denen alle Endpunkte sichtbar sind. Sind alle Endpunkte eines Links in der Darstellung enthalten, wird der entsprechende Link ebenfalls sichtbar. Dieses Verhalten ist ein Nebeneffekt der Aktion *show* auf Objekte und wird nicht als Ersatz für das fehlende Menü *Show links* angesehen.

Die Möglichkeit *crop* auf einen selektierten Link anzuwenden ist nicht vorhanden. Nach der Selektion eines Links findet keine Erweiterung des Kontextmenüs um die Option *crop <Name des Links>* statt. Das Fehlen dieser Funktionalität wirkt sich negativ auf die Vollständigkeit der Umsetzung der Basisoperationen.

Die Analyse der Anwendung der Basisoperationen resultiert in einem negativen Ergebnis. Zwei Operationen werden dem Anwender gar nicht angeboten und die Aktion *hide* nur über eine vorherige Selektion. Insgesamt wurden deutliche Mängel in der Vollständigkeit der Umsetzung ausgemacht und Abweichungen bei der Art der Benutzung festgestellt. Eine einheitliche Umsetzung im Vergleich zu der Ansicht Klassendiagramm oder auch zu den Anwendungsmöglichkeiten auf Objekte ist nicht gegeben.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Link** | mit Einschränkungen | nicht vorhanden | nicht vorhanden |

Tabelle 7: Basisoperationen auf Links

## 4.4. Übergreifende Aspekte

In diesem Abschnitt werden Verhaltensweisen, mögliche Verbesserungen und Fehler beschrieben, die sowohl in den Ansichten Klassen- als auch im Objektdiagramm auftreten. Die gefundenen Auffälligkeiten werden an einem Beispiel gezeigt. Dabei kann ein Klassen- oder Objektdiagramm verwendet werden. Sofern es nicht explizit anderes beschrieben wird, ist das Verhalten in beiden Diagrammen identisch und gilt für beide Diagrammarten.

Die Abbildung *Positionierung von Elementen* zeigt zwei Darstellungen eines Modells. Das linke Klassendiagramm weist teilweise und vollständige Überdeckungen von Klassen, Beziehungen und Detailinformationen auf. In der Nähe der Klassen *Student* und *Tutorial* werden Informationen zu Rollennamen nicht eindeutig dargestellt. Durch die Kollision von verschiedenen Elementen ist nicht nur die Beschriftung unklar zu erkennen, sondern auch die Zuordnung nicht zweifelsfrei möglich. Die abgebildete Darstellung entstand nachdem eine Spezifikation geladen und ein Klassendiagramm erzeugt wurde. Durch manuelle Verschiebungen per Tastatur oder Maus ist eine Änderung der Anordnung möglich, sodass eine Positionierung der Elemente ohne Überdeckungen entstehen kann. Das rechte Klassendiagramm zeigt eine mögliche Darstellung ohne Überdeckungen. Die Bezeichnungen der Rollen und Assoziationsnamen sind erkennbar und können zugeordnet werden. Um die wiederholte manuelle Platzierung zu vermeiden, bietet USE die Möglichkeit, die aktuelle Anordnung der Elemente zu speichern und diese Anordnung bei Bedarf zu laden.

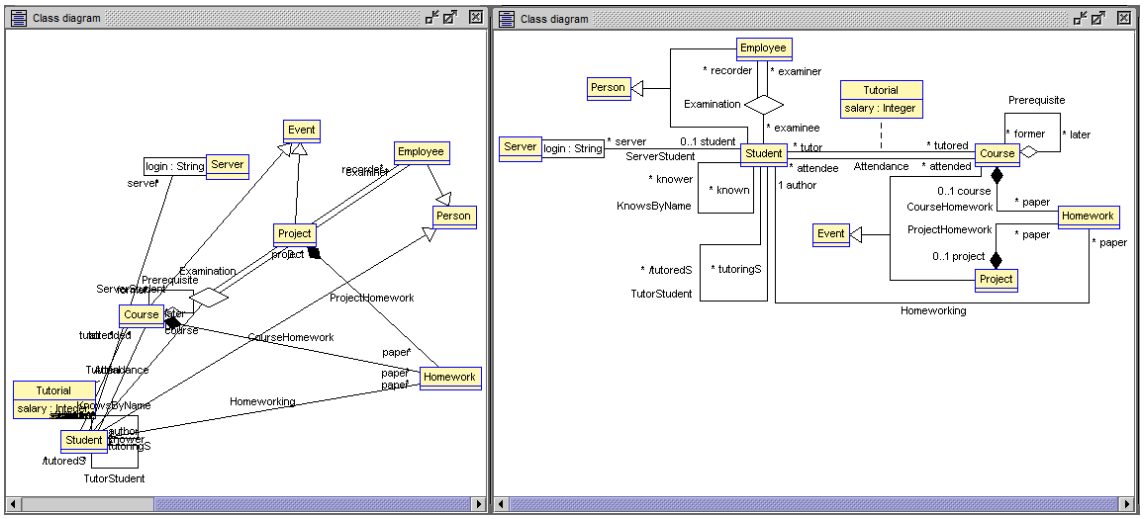


Abbildung 16: Positionierung von Elementen

Laut Schlobohm [Sch17] kollidieren bei der Erzeugung von Kommunikationsdiagrammen ebenfalls Elemente. Als Ursache für die zufällige Platzierung wird das Fehlen von formalen Regeln zur Platzierung genannt. Die Möglichkeit verschiedene Layouts zu erstellen und diese später verwenden zu können, verhindert nicht nur die wiederholte Platzierung der Komponenten, sondern ermöglicht dem Anwender zusätzlich verschiedene Darstellungen eines Modells vorhalten und bei Bedarf verwenden zu können.

Bei der Untersuchung eines erstellten Layouts fiel auf, dass unter anderem gespeichert wird, ob eine Darstellungskomponente sichtbar oder versteckt ist. Wird beispielsweise eine Klasse versteckt und anschließend ein Layout gespeichert, dann ist die ausgeblendeten Klasse in dem Layout-Datei mit <hidden>**true**</hidden> versehen. Das Laden einer gespeicherten Darstellung führt jedoch dazu, dass trotz der *hidden*-Markierung versteckte Elemente wieder sichtbar sind. Daraus resultiert, dass Informationen über die Sichtbarkeit gespeichert, aber beim Laden nicht korrekt ausgewertet oder verwendet werden. Nach Rücksprache mit Prof. Dr. Martin Gogolla ist es nicht Bestandteil dieser Arbeit diesen Fehler zu korrigieren. Eine Erweiterung des Funktionalität darf aber nicht zu einer weiteren Verschlechterung der Funktion *Layout* führen.

Die vorhandenen Menüs und Einträge im Kontextmenü können je nach Systemzustand variieren. Aber nicht nur die reine Existenz variiert, sondern auch die Inhalte von einigen Menüs. Am Beispiel des Menüs *Hide Classes* soll der Aufbau eines Menüs verdeutlicht werden. Der erste Eintrag *Hide all classes* ist statisch und ist immer vorhanden. Unterhalb dieses Eintrages werden verschiedene Klassen aufgelistet, die in der Darstellung enthalten sind. Die Abbildung *Struktur dynamische Menüeinträge* veranschaulicht den allgemeinen Aufbau und die Struktur. Die Anzahl der aufgelis-

teten Klassen ist abhängig von der Anzahl der sichtbaren Klassen und ist nicht begrenzt. Bei einer großen Anzahl von Klassen wird die Auflistung länger, bis die Liste den unteren Rand des Bildschirms erreicht hat. Die verwendete Auflösung des Bildschirms begrenzt die Anzahl der dargestellten Klassen in dieser dynamischen Liste. Sollte die Liste mehr Elemente enthalten, als angezeigt werden können, werden die überzähligen Elemente außerhalb des sichtbaren Bereiches des Bildschirms dargestellt. Für einen Anwender sieht es aus, als ob das Menü abgeschnitten wurde. Durch die fehlende Anzeige von einigen Elementen können auch keine Operationen auf diese ausgeführt werden und die Aufzählung wirkt unvollständig. Die Reproduktion dieses Fehlverhaltens ist abhängig von der verwendeten Auflösung und der Anzahl der Elemente. Alle Menüs, die Aktionen auf eine variable Anzahl von Elementen anbieten, sind von diesem Effekt betroffen. Zu nennen wären beispielweise die Menüs *Hide classes*, *Show classes*, *Hide associations* und *Show associations*.

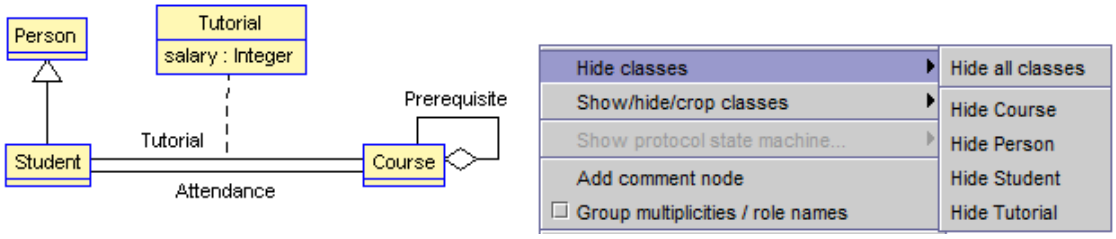


Abbildung 17: Struktur dynamische Menüeinträge

Des Weiteren ist die Auflistung von Elementen nicht nur von der Anzahl der Einträge abhängig, sondern auch von der verwendeten Sortierung. In der aktuellen USE-Version wird eine alphabetisch aufsteigende Sortierung verwendet. Das Listing *Unterschied alphabetische und alphanumerische Sortierung* zeigt eine solche Sortierung. Das Ergebnis der Sortierung ist für den Anwender möglicherweise auf den ersten Blick nicht verständlich, da trotz aufsteigender Reihenfolge der Eintrag *Klasse10* vor *Klasse2* eingereiht wird. Die Verwendung einer alphanumerische Sortierung würde Abhilfe schaffen. Zur Vereinheitlichung wäre die Sortierung auf alle dynamischen Einträge anzuwenden.

```
Alphabetische Sortierung:    Klasse1, Klasse10, Klasse2, Klasse21, Klasse3
Alphanumerische Sortierung: Klasse1, Klasse2,  Klasse3, Klasse10, Klasse21
```

Listing 3: Unterschied alphabetische und alphanumerische Sortierung

Zudem wurden bei der Untersuchung der Anwendungsmöglichkeiten der Basisoperationen Fälle betrachtet, bei denen ein Element selektiert wurde und anschließend das Kontextmenü um Optionen erweitert wurde. In den Beispielen wurde immer genau ein Element ausgewählt. Prinzipiell ist aber auch die Auswahl von mehreren Elementen möglich. Werden mehrere Elemente selektiert, ist die Basisoperation auf die gesamte Auswahl anzuwenden. Die Abbildung *Fehlerfall Basisoperation auf*

*mehrere Elemente* zeigt ein Klassendiagramm, in dem die beiden Klassen *Student* und *Person* sowie die Assoziation *Attendance* selektiert sind. Das dargestellte Kontextmenü wurde um die zwei Optionen *Crop 2 classes* und *Hide 2 classes* erweitert. Die getätigte Auswahl umfasst auch eine Assoziation, für die keine Aktionen angeboten wird. Die Ansicht Objektdiagramm weist ein vergleichbares Verhalten auf. Bei der Selektion von Objekten und Links werden nur Aktionen auf die ausgewählten Objekte angeboten. Die angebotenen Funktionen im Kontextmenü entsprechen nicht den Erwartungen, da nicht nachvollziehbar ist, warum die Aktionen nur auf Klassen beziehungsweise Objekte verfügbar sind und die Auswahl der Assoziationen beziehungsweise Links nicht berücksichtigt wird.

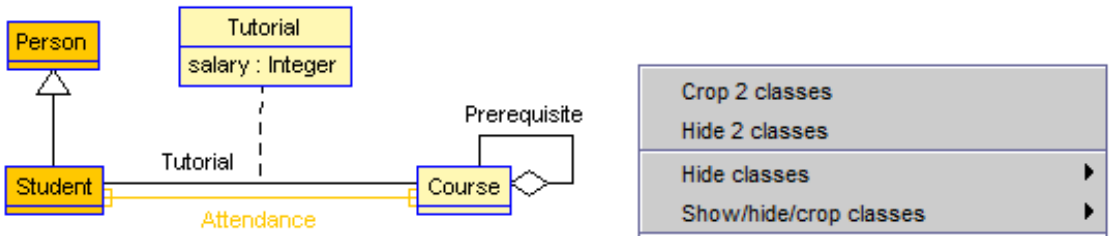


Abbildung 18: Fehlerfall Basisoperation auf mehrere Elemente

Versteckte Elemente sollten mit der Operation *show* wieder sichtbar gemacht werden können. Die Analyse zeigte, dass die Operation *show* über ein entsprechendes Menü auf verborgene Elemente angewandt werden kann. Das Kontextmenü enthält noch eine weitere Option, die Elemente aus der Menge *hiddenData* entfernt und der Menge *visibleData* hinzufügt. Die Option *Show hidden classes and associations* ist in der Ansicht Klassendiagramm verfügbar, wenn mindestens eine Klasse oder Assoziation versteckt wurde. In der Ansicht Objektdiagramm ist eine Option mit dem Namen *Show hidden objects* vorhanden. Die Bezeichnung *Show hidden classes and associations*, wie in der gleichnamigen Abbildung *Show hidden classes and associations* gezeigt, signalisiert das versteckte Klassen und Assoziationen sichtbar werden. Sollte es nur versteckte Klassen und keine versteckten Assoziationen geben, signalisiert der Name der Option das sowohl Klassen als auch Assoziationen sichtbar werden. Da keine verborgenen Assoziationen vorhanden sind, können diese auch nicht sichtbar werden. Des Weiteren nimmt diese Aktion auch ausgeblendete Generalisierungen wieder in die Darstellung auf, was aus dem Namen nicht ableitbar ist. *Show hidden objects* im Objektdiagramm wird im Kontextmenü nur angeboten, wenn mindestens ein Objekt verborgen ist. Versteckte Links finden, im Gegensatz zu den Assoziationen, keine Berücksichtigung, obwohl durch das Einblenden aller Objekte auch alle Links sichtbar werden. Eine passendere Bezeichnung dieser Option, die das ausgelöste Verhalten präziser erfasst, wäre wünschenswert.

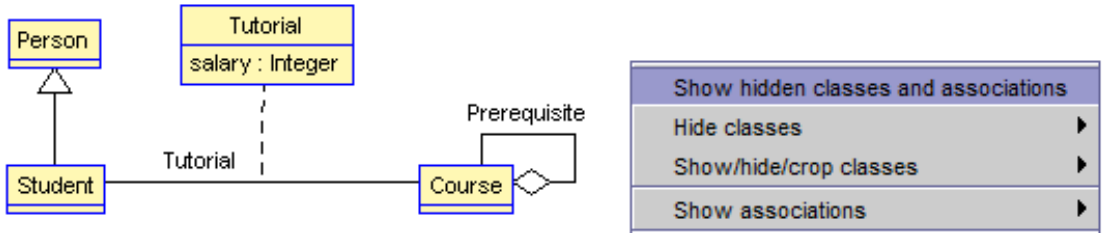


Abbildung 19: *Show hidden classes and associations*

## 4.5. Ergebnisse der Analyse

In dem letzten Abschnitt dieses Kapitels werden die Ergebnisse der Analyse zusammengefasst und auf die zu Beginn gestellten Fragen zurückgekommen. Die Zusammenfassung bildet die Grundlage für zukünftige Verbesserungen der vorhandenen Funktionen und Benutzungsmöglichkeiten. Zusätzlich ermöglicht das Ergebnis die Konzeption und Implementierung zusätzlicher Funktionalitäten.

Die drei Basisoperationen *hide*, *show* und *crop* stellen die Basis für eine Vielzahl von Interaktionen dar. Diese drei Operationen sind sowohl in der Ansicht Klassendiagramm als auch in der Ansicht Objektdiagramm auf eine Vielzahl von Elementen anwendbar. Jedoch sind die Aktionen nicht auf alle Konzepte vollständig nutzbar und teilweise fehlerhaft implementiert. Das hat zur Folge, dass die Bedienung je nach Ansicht und Konzept abweicht und keine Einheitlichkeit vorhanden ist. Der Wiedererkennungswert eines übergeordneten Schemas, dass durch eine konsequente Umsetzung der Basisoperationen erreicht werden könnte, ist beeinträchtigt. Das Fehlen oder die fehlerhafte Umsetzung einiger Aktionen führt zu einer Einschränkung der Gestaltungsmöglichkeiten. Das effiziente Erzeugen von gewünschten Darstellungen ist erschwert.

Die Tabelle *Implementierte Basisoperationen* fasst die einzelnen Untersuchungsergebnisse zusammen und verdeutlicht welche Benutzungsmöglichkeiten vorhanden sind. Dabei wird auch ersichtlich, an welchen Stellen Nachbesserungen vorzunehmen sind, um ein übergeordnetes und einheitliches Schema umzusetzen.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Klasse** | vorhanden | vorhanden | mit Einschränkungen |
| **Assoziation** | mit Einschränkungen | mit Einschränkungen | nicht vorhanden |
| **Generalisierung** | nicht vorhanden | nicht vorhanden | nicht vorhanden |
| **Objekt** | vorhanden | vorhanden | mit Einschränkungen |
| **Link** | mit Einschränkungen | nicht vorhanden | nicht vorhanden |

Tabelle 8: Implementierte Basisoperationen

Aus den aufgedeckten Fehlverhalten und gefundenen Verbesserungen der bestehenden Funktionen werden Anforderungen gebildet. Diese Anforderungen werden im Kontext dieser Arbeit umgesetzt, um dem Anwender eine effizientere Anpassung von Darstellungen zu ermöglichen. Die zu schaffenden Auswahl- und Filterfunktionen zielen auf eine vollständige und einheitliche Umsetzung der Basisoperationen ab. Die entstehenden positiven Effekte in Bezug auf Wiedererkennung und Diagrammgestaltung sollen die Nutzungsmöglichkeiten von USE stärken und vereinfachen. Die Tabelle *Anforderungen an eine neue Version* fasst die umzusetzenden Kriterien zusammen und bildet den Arbeitsauftrag für das Kapitel *Implementierung und Anpassung der Benutzungsmöglichkeiten*. Jede Anforderung wird mit einer Nummer zur Identifizierung und einer Beschreibung versehen.

| Nr. | Ansicht | Beschreibung |
|---|---|---|
| 1 | Klassendiagramm | Zur einheitlichen Menüführung ist die Einführung des Menüs *Hide Associations* notwendig. Das Menü wird in das Kontextmenü eingebunden und ermöglicht die Ausführung der Basisoperation *hide*. |
| 2 | Klassendiagramm | Die Anwendung der Operation *crop* auf eine Assoziationsklasse resultiert in einer fehlerhaften Darstellung. Alle sichtbaren Elemente werden versteckt, sodass eine Darstellung ohne Elemente entsteht. |
| 3 | Klassendiagramm | Das Menü *Show associations* soll nur verfügbar sein, wenn mindestens eine versteckte Assoziation vorhanden ist. Aktuell wird dieses Menü auch eingeblendet, wenn eine Generalisierung versteckt wurde. |
| 4 | Klassendiagramm | Die Operation *show* auf eine Assoziation mit Assoziationsklasse führt zu einer unvollständigen und fehlerhaften Darstellung, da die Assoziationsklasse nicht sichtbar wird. |
| 5 | Klassendiagramm | Nach der Selektion einer Assoziation wird das Kontextmenü nicht um die Option *crop <Name der Assoziation>* erweitert. |
| 6 | Klassendiagramm | Die Anwendung der Basisoperationen auf Generalisierungen ist nicht vorhanden. Die Aktion *hide*, *show* und *crop* sind vollständig zu implementieren. |
| 7 | Objektdiagramm | Die Anwendung der Operation *crop* auf ein Linkobjekt resultiert in einer fehlerhaften Darstellung. Alle sichtbaren Elemente werden versteckt, sodass eine Darstellung ohne Elemente entsteht. |
| 8 | Objektdiagramm | Zur einheitlichen Menüführung ist die Einführung des Menüs *Hide links* notwendig. Das Menü wird in das Kontextmenü eingebunden und ermöglicht die Ausführung der Basisoperation *hide*. |
| 9 | Objektdiagramm | Zur einheitlichen Menüführung ist die Einführung des Menüs *Show links* notwendig. Das Menü wird in das Kontextmenü eingebunden und ermöglicht die Ausführung der Basisoperation *show*. |
| 10 | Objektdiagramm | Nach der Selektion eines Links wird das Kontextmenü nicht um die Option *crop <Name des Links>* erweitert. |
| 11 | Übergreifend | Dynamische Menüeinträge können zu einer zu langen Auflistung führen. Enthält die Auflistung zu viele Elemente, werden einige Elemente nicht dargestellt. |
| 12 | Übergreifend | Die Sortierung der Auflistung erfolgt alphabetisch und soll auf alphanumerisch geändert werden. |
| 13 | Übergreifend | Sind unterschiedliche Elemente selektiert, werden über das Kontextmenü nur die Aktionen *hide* und *crop* auf Klassen beziehungsweise Objekte angeboten. |
| 14 | Übergreifend | Die Bezeichnung für die Aktionen *Show hidden classes and associations* und *Show hidden objects* soll präzisiert werden. |

Tabelle 9: Anforderungen an eine neue Version

# 5. Konzeption neuer Benutzungsmöglichkeiten

In dem vorherigen Kapitel wurde die Frage behandelt, ob neue Benutzeraktionen eingeführt werden müssen, um eine effektivere Anpassung der Darstellung zu ermöglichen. Das Ergebnis der Analyse war, dass das übergeordnete Schema mit den drei Basisoperationen gestärkt und vollständig implementiert werden muss. Zur Erreichung der einheitlichen Anwendung von Operationen und dessen Gestaltung werden neue Interaktionen geschaffen. Alle zu schaffenden Benutzungsmöglichkeiten orientieren sich an Interaktionen auf Klassen, da die Umsetzung der Basisoperationen auf Grund der Vollständigkeit der Aktionen als Referenzpunkt gewählt wurde. Dieses Kapitel greift die Fragestellung unter Betrachtung einer anderen Perspektive erneut auf. Der Fokus ist auf die Schaffung neuer Benutzungsmöglichkeiten in Form von neuen Filter- und Auswahlfunktionen gelegt.

- Welche zusätzlichen Möglichkeiten können für eine Veränderung der Darstellung hilfreich sein?
- Gibt es neben den Basisoperation weitere Aktionen, die zur Gestaltung von Darstellungen verwendet werden können?
- Kann es andere Zustände als sichtbar oder unsichtbar für Elemente geben?

## 5.1. Ansicht Klassendiagramm

Dieser Abschnitt beschreibt neue Nutzungsmöglichkeiten, die für die Ansicht Klassendiagramm konzeptioniert wurden. Die Analyse ergab, dass in der aktuellen USE-Version keine der drei Basisoperationen auf Generalisierungen angewendet werden können und empfahl die vollständige Implementierung der Funktionen. Eine Generalisierung ist eine gerichtete Beziehung und die Richtung wird durch die Verwendung einer Pfeilspitze notiert. Klassen können durch eine Generalisierung in Beziehung zueinander gesetzt werden. Eine solche Beziehung versieht die beteiligten Klassen mit einer Rolle. Eine Klasse nimmt die Rolle der Oberklasse ein und die andere Klasse wird als Unterklasse bezeichnet. In den bisherigen USE-Versionen werden keine Operationen auf Ober- und Unterklassen angeboten und dies wird durch eine neue Funktionalität geändert.

Die Abbildung *Basisoperationen auf Unterklassen* zeigt ein abstraktes Beispiel für die Kombination von Unterklassen und den Basisoperationen. Das Beispiel besteht aus sechs Klassen, wobei die grau hinterlegten Klassen zum aktuellen Zeitpunkt versteckt sind. Die Klasse *C* ist die Unterklasse von den Oberklassen *SUP1* und *SUP2*. Zugleich ist die Klasse *C* auch die Oberklasse von den Klassen *SUB1* und *SUB2*. Die Klasse *D* steht in keiner Beziehung zu den anderen Klassen. Ausgehend von der

Klasse *C* wird das Verhalten der neuen Funktionen beschrieben, der Anwendung der Basisoperation auf Unterklassen.

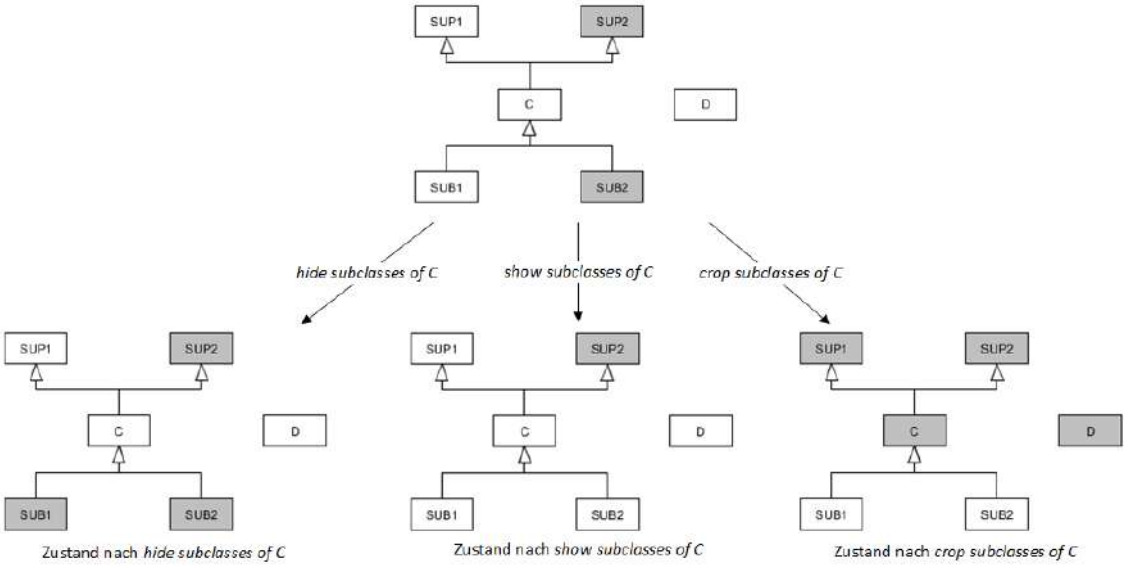


Abbildung 20: Basisoperationen auf Unterklassen

Die Aktion *hide subclasses of C* kombiniert die Basisoperation *hide* mit den Unterklassen der Klasse *C*. Die Unterklasse *SUB2* ist bereits versteckt, weshalb keine Veränderung der Sichtbarkeit erfolgt. Die sichtbare Unterklasse *SUB1* wird verborgen, sodass nur noch die Klassen *SUP1* ,*C* und *D* in der Darstellung enthalten sind. Die drei genannten Klassen sind nicht in der Menge der Unterklassen von *C* vorhanden und daher nicht von der Aktion betroffen.

Die Basisoperation *show* auf die Unterklassen von *C* verändert die Sichtbarkeit der Klasse *SUB2* von verborgen auf sichtbar. Weitere Klassen sind von dieser Aktion nicht betroffen. Zu beachten ist, dass die Aktion *show subclasses of C* die Aktion *hide subclasses of C* nicht vollständig umkehrt. Der Ausgangszustand mit einer versteckten Unterklasse kann durch die Interaktion nicht wiederhergestellt werden. Die Ausführung von *show* resultiert in der Sichtbarkeit aller Unterklassen. Zur Wiederherstellung einer ursprünglichen Darstellung kann die Verwendung der Aktionen *hide* und *show* auf Klassen notwendig sein.

Die beiden genannten Operationen verändern ausschließlich die Sichtbarkeit der Unterklassen. Anders verhält sich die Aktion *crop subclasses of C*, bei der die Sichtbarkeit aller Klassen verändert werden kann. Die beiden Unterklassen *SUB1* und *SUB2* werden in die Menge *visibleData* aufgenommen beziehungsweise bleiben in der Menge, während alle anderen Klassen in die Menge *hiddenData* überführt werden. Erstmals sind auch die Klassen *SUP1*, *SUP2*, *C* und *D* betroffen. Durch das

Verbergen der Klasse $C$ verschwindet sogar die Klasse, auf die die Aktion ausgeführt wird, aus der Darstellung. Die Umkehrung und Rückkehr in den Ausgangszustand ist durch die mehrfache Anwendung von *hide* und *show* auf Klassen möglich.

Die neuen Anwendungsmöglichkeiten ermöglichen die Veränderung der Darstellung auf eine Art und Weise, die bisher nur durch eine mehrfache Ausführung von Aktionen erreicht werden konnte. Die konzeptionierte Lösung kann Darstellungen mit Unterklassen effizient und schnell verändern. Neue Darstellungen, die den Fokus der Betrachtung auf die Bedeutung von Unterklassen setzen, sind mit einer einfachen Interaktion möglich. Zusammenfassend kombinieren die neuen Interaktionen mehrere vorhandene Aktionen und bieten dadurch eine effizientere Benutzungsmöglichkeit.

Die Kombination der Basisoperationen im Zusammenspiel mit Oberklassen ähnelt dem beschriebenen Vorgehen und ermöglicht ebenfalls neue Gestaltungsoptionen. Aus diesem Grund sind nicht nur die Basisoperationen auf Unterklassen, sondern auch auf Oberklassen umzusetzen. Weitere zusätzliche Interaktionen in der Ansicht Klassendiagramm sind nicht geplant.

## 5.2. Ansicht Objektdiagramm

Nachdem die neuen Funktionen in der Ansicht Klassendiagramm beschrieben wurden, wird nun die Konzeption neuer Anwendungsmöglichkeiten in der Ansicht Objektdiagramm behandelt. Die neuen Interaktionen ermöglichen eine zusätzliche Darstellung von Objekten in Bezug auf ihre Sichtbarkeit und die Anwendung der Basisoperationen *hide* und *show* auf Assoziationstypen.

Bisher wird bei der Sichtbarkeit von Elementen zwischen den Zuständen *Versteckt* und *Sichtbar* unterschieden. Folglich ist ein Element entweder in der Darstellung vorhanden und wird angezeigt oder es ist in der Menge *hiddenData* enthalten. Im Kontext dieser Arbeit soll ein dritter Zustand auf Objekte eingeführt werden. Der neue Zustand ermöglicht das Ausgrauen von Objekten, um eine schwächere Form der Sichtbarkeit zu ermöglichen. Ein ausgegrautes Objekt ist weiterhin sichtbar und folglich in der Menge der sichtbaren Elemente enthalten. Daher können auf diese Objekte weiterhin alle Interaktionen ausgeführt werden, die auf sichtbare Objekte angewandt werden können. Die Darstellung eines Objektes in USE wird so verändert, dass der Kontrast zwischen der Darstellung des Objektes und dem Hintergrund der Ansicht abgeschwächt wird. Die dunkelgraue Hintergrundfarbe eines Objektes wird zu einem hellen Grau modifiziert und der schwarze Rahmen des Objektes ersetzt. Durch diese beiden Maßnahmen unterscheidet sich die Darstellung eines ausgegrauten Objekts nicht mehr so stark vom weißen Hintergrund und der Schwerpunkt der

Betrachtung liegt nicht auf diesen Objekten. Der zusätzliche Zustand soll eine Abschwächung der Fokussierung beziehungsweise Stärkung der Fokussierung auf andere Objekte ermöglichen. In einem Szenario kann eine Vielzahl von Objekten existieren. Je nach Anwendungsfall werden allerdings nicht alle Objekte benötigt. Diese können mit der Operation *hide* ausgeblendet werden. Was ist aber mit Objekten, die für das Verständnis von Zusammenhängen von Bedeutung sind, aber nicht für den Kern der Betrachtung? Genau dieser Anwendungsfall soll durch den dritten Zustand der Sichtbarkeit abdeckt werden.

Objekte sind Instanzen von Klassen und in vielen Anwendungsfällen existieren mehr Objekte als Klassen. Folglich sind in einem Objektdiagramm viele Elemente vorhanden, wobei einige Objekte nicht immer von Bedeutung sind. Um die wahrscheinlich größere Anzahl an Objekten im Vergleich zu Klassen, Assoziationen, Generalisierungen und Links zu nutzen, wird der dritte Zustand für Objekte eingeführt, um Erfahrungen sammeln zu können. Eine Verbreitung und vollständige Implementierung auf alle Konzepte ist erst nach einer positiven Evaluation sinnvoll.

Zusätzlich zum Ausgrauen von Objekten muss eine Möglichkeit geschaffen werden, den Zustand eines ausgegrauten Objektes in ein normal sichtbares Objekt zu überführen. Die Anwendung der Aktion *hide*, gefolgt von einem *show*, auf ein solches Objekt würde den gewünschten Zustand herbeiführen. Allerdings wird dieser Vorgang als umständlich angesehen und eine eigene Aktion zur Umkehrung bevorzugt. Des Weiteren könnte das Ausführen von *hide* und *show* zu ungewollten Nebeneffekten führen, indem verborgene Assoziationen sichtbar werden. Die Veränderung der Anzahl der sichtbaren Elemente, also der Anzahl der in der Menge *visibleData* enthaltenen Elemente, ist nicht Ziel des neuen Zustands.

Nun wird die angekündigte Interaktion, die die beiden Basisoperationen *hide* und *show* mit Assoziationstypen verbindet, beschrieben. In der Ansicht Objektdiagramm werden Objekte und Links zur Darstellung verwendet. Ein Link ist eine Instanz einer Assoziation und zu einer Assoziation kann es mehrere Links geben. Die Bestimmung der Assoziation zu einem Link ist möglich. Das Listing *Ebenen im Kontextmenü* verdeutlicht die Zuordnung und zeigt die Entstehung einer zusätzlichen Ebene durch die Instanziierung. Der Link *(ada,TCS1)* verbindet die Studentin *ada* mit dem Kurs *TCS1* und ist eine Instanz der Assoziation *Attendance*.

```
Klassendiagramm:
Aufbau:          Menü -> <Name der Aktion> <Name der Assoziation>
Beispiel:        Hide associations -> Hide Attendance

Objektdiagramm:
Aufbau:          Menü -> <Name der Assoziation> ->
                         <Name der Aktion> <Name der Instanz>
Beispiel:        Hide links -> Attendance -> Hide (ada,TCS1)
```

Listing 4: Ebenen im Kontextmenü

In dem Kapitel *Grundlagen* wurden verschiedene Arten oder Typen von Assoziationen vorgestellt. Bei einer Assoziation kann es sich beispielsweise um eine Aggregation, Komposition oder binäre Assoziation handeln. Die neue Funktion soll im Rahmen eines Menüs über Kontextmenü realisiert werden und die Operationen *hide* und *show* auf Assoziationstypen ermöglichen. Zu einem Link wird die zugehörige Assoziation bestimmt und zu der Assoziation wird der Assoziationstyp ermittelt. Die zugehörige Assoziation zu dem Link *(ada,TCS1)* ist *Attendance* und *Attendance* ist eine binäre Assoziation. In dem neuen Menü wird nicht die Assoziation in der ersten Ebene des Menüs verwendet, sondern der Assoziationstyp. Es entsteht eine Auflistung aller Assoziationstypen, die über eine Instanz verfügen. In dem Listing wäre der Platzhalter *<Name der Assoziation>* durch den *<Name des Assoziationtyp>* zu ersetzen. Unterhalb des jeweiligen Typs werden die Instanzen aufgezählt und die Anwendung der beiden Basisoperationen ermöglicht.

Diese Filter- und Auswahloption ermöglicht das Verstecken oder die Anzeigen von Links auf Basis des Typs der Assoziation. Bestimmte Arten von Assoziationen können effizient ein- und ausgeblendet werden und fördern die Fokussierung auf bestimmten Arten von Beziehungen. Dem Anwender wird ein weiteres Werkzeug zur Erstellung und Anpassung der Darstellung bereitgestellt. Auf die Operation *crop* wird bewusst verzichtet. Die Operation ist bisher nur verfügbar, wenn Elemente selektiert wurden und eine Selektion findet hier nicht statt. Weitere neue Funktionen sind in dieser Ansicht nicht geplant.

## 5.3. Zusammenfassung

Die in diesem Kapitel entworfenen neuen Benutzungsmöglichkeiten erweitern die Auswahl- und Filteroptionen und ermöglichen neue Wege Diagramme zu gestalten. Die Wiederverwendung einiger Basisoperationen unterstreicht erneut den Wert dieser Operationen und verstärkt die Dringlichkeit die Operationen vollständig und fehlerfrei zu implementieren. Das Fehlen einiger Interaktionen in der aktuellen USE-Version verhindert die Umkehrung der neuen Funktionen und die Wiederherstellung

des Ausgangszustands.

Insgesamt weisen die neuen Funktionen das Potenzial auf, eine effizientere Gestaltung der Darstellung zu ermöglichen. Eine Überprüfung, ob die Aktionen geeignet sind, kann jedoch erst nach der Implementierung erfolgen. Die beschriebenen Abhängigkeiten, also das Fehlen von Interaktionen und die damit verbundene Einschränkung der Umkehrung der neuen Funktionen, kann negative Auswirkungen auf die Akzeptanz und Nutzbarkeit der neuen Möglichkeiten haben. Eine erfolgreiche Umsetzung würde die Veränderung der Darstellung vereinfachen und die Möglichkeiten von USE erweitern. Die geplanten Funktionen sind als spezifische Erweiterungen anzusehen und gefährden nicht die einheitliche Umsetzung.

Die Tabelle *Neue Benutzungsmöglichkeiten* fasst die beschriebenen Erweiterungen zusammen und dient als Referenz in der Umsetzungsphase. Dabei werden die genannten Features über eine Nummer identifiziert. Die Implementierung erfolgt im Kapitel *Implementierung und Anpassung der Benutzungsmöglichkeiten*.

| **Nr.** | **Ansicht** | **Beschreibung** |
|---|---|---|
| 1 | Klassendiagramm | Die Basisoperationen *hide*, *show* und *crop* sollen auf Unterklassen angewendet werden können. Das Verhalten der Aktionen wurde in diesem Kapitel beschrieben. |
| 2 | Klassendiagramm | Die Basisoperationen *hide*, *show* und *crop* sollen auf Oberklassen angewendet werden können. Das Verhalten der Aktionen und die Menügestaltung sollen mit den Aktionen auf Unterklassen harmonisieren. |
| 3 | Objektdiagramm | Ein neuer Zustand für Objekte soll eingeführt werden, der das Ausgrauen ermöglicht. Dieser Zustand stellt eine Zwischenstufe zwischen den beiden Zuständen *Versteckt* und *Sichtbar* dar. |
| 4 | Objektdiagramm | Es muss eine Möglichkeit geschaffen werden, um ausgegraute Objekte wieder in ihre ursprüngliche Darstellung zu versetzen. |
| 5 | Objektdiagramm | Vorhandene Links sollen nach ihrer Assoziationsart sortiert und aufgelistet werden. Zusätzlich sollen alle Links einer Assoziationsart versteckt oder angezeigt werden können. Die Unterstützung von *hide* und *show* auf Links ist ebenfalls zu implementieren. |

Tabelle 10: Neue Benutzungsmöglichkeiten

# 6. Implementierung und Anpassung der Benutzungsmöglichkeiten

In der Analysephase wurde die Anwendung der Basisoperationen auf Vollständigkeit und Korrektheit geprüft. Anschließend wurden im Kapitel *Konzeption neuer Benutzungsmöglichkeiten* Ideen für neue Interaktionen entwickelt. Die Ergebnisse der beiden Kapitel wurden in Form von Anforderungen an eine neue USE-Version festgehalten. Im Verlauf dieses Kapitels wird die Umsetzung der Anforderungen beschrieben, die in einer neuen USE-Version resultieren.

Die Änderungen am Quellcode zur Realisierung der Anforderungen werden in dieser Ausarbeitung nicht dargestellt, aber der Arbeitsgruppe *Datenbanksysteme* zur Verfügung gestellt. Alle nachfolgenden Darstellungen wurden mit einer neuen Version von USE erzeugt, die zum aktuellen Zeitpunkt noch über keine offizielle Versionsnummer verfügt. Es handelt sich nicht um die Version 5.0.1.

## 6.1. Ansicht Klassendiagramm

Viele Änderungen erfordern eine Anpassung des Kontextmenüs in der Ansicht Klassendiagramm. Zur Vermeidung von doppelten Darstellungen werden die Änderungen nicht nach Klassen, Assoziationen und Generalisierungen gruppiert. Zuerst werden Anforderungen betrachtet, die eine Veränderung des Kontextmenüs auslösen ohne das ein Element selektiert wurde. Anschließend wird die Umsetzung der Anforderungen beschrieben, bei denen die Selektion von einem oder mehreren Elementen erforderlich ist.

Die Abbildung *Neues Kontextmenü Klassendiagramm* zeigt drei mögliche Ausprägungen des Kontextmenüs. In den oberen beiden Klassendiagrammen sind eine Generalisierung und drei Assoziationen sichtbar, die versteckt werden können. Das Verstecken von Elementen kann über die Menüs *Hide associations* und *Hide generalizations* ausgelöst werden. Beide Menüs bieten die Möglichkeit alle sichtbaren Elemente oder gezielt ein Element zu verbergen. Um eine einheitliche Gestaltung der Menüs umzusetzen, ist der Aufbau an das Menü *Hide classes* angelehnt. In der untersten Darstellung wurde die Klasse *Person* und die entsprechende Generalisierung versteckt. Die verborgene Generalisierung resultiert in der Anzeige des Menüs *show generalizations*, das genutzt werden kann, um versteckte Generalisierungen wieder einzublenden. Bei diesem Szenario wurde in der USE-Version 5.0.1 das Menü *show associations* angezeigt, obwohl keine versteckte Assoziation vorhanden war. Dieses Verhalten wurde, wie die Abbildung aufzeigt, behoben. Zudem belegt die Abbildung

die Umsetzung von Menüs, die das Verstecken von Assoziationen und Generalisierung sowie die Anzeige von Generalisierungen ermöglichen.

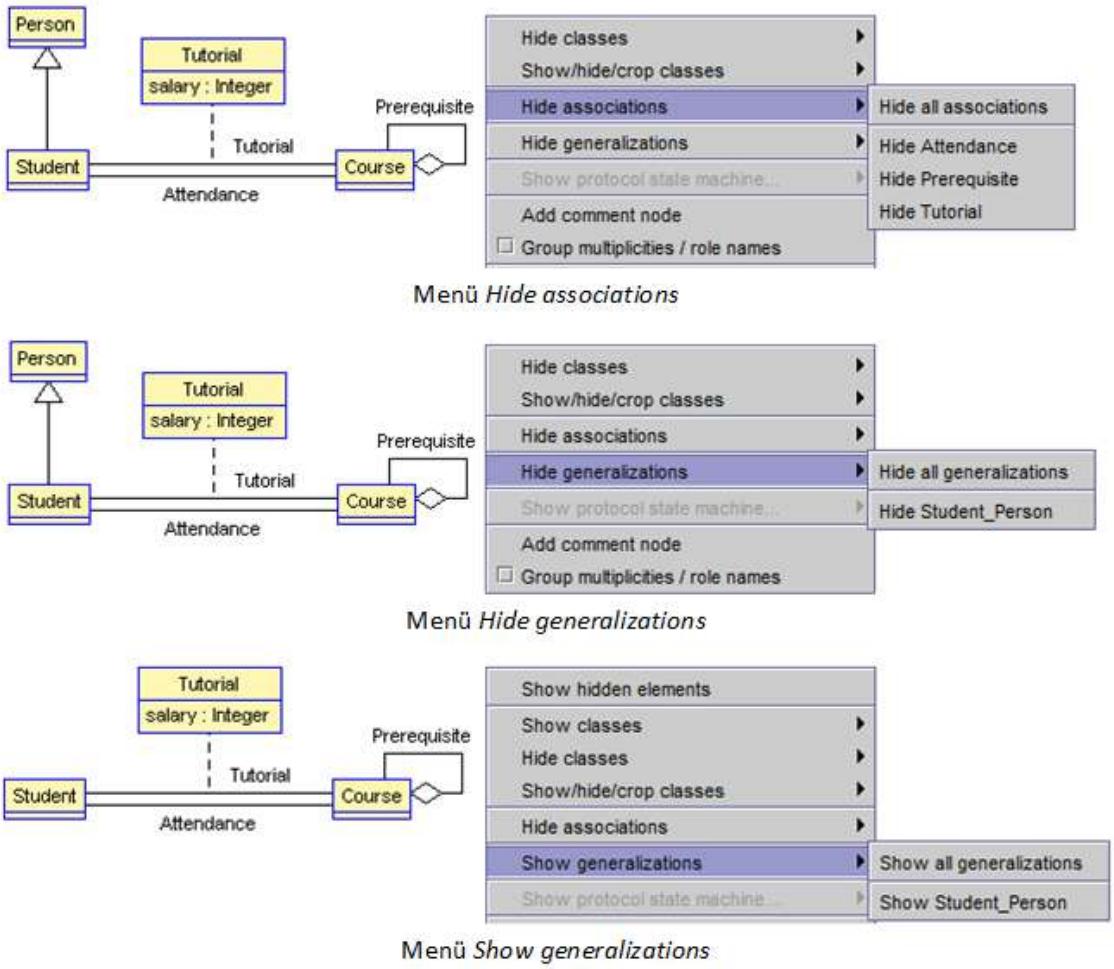


Abbildung 21: Neues Kontextmenü Klassendiagramm

Die Operation *hide* auf Assoziationen wurde bereits auf ihr korrektes Verhalten geprüft. Da keine Anpassungen an dem internen Verhalten der Operationen erfolgte, wird von einer erneuten Untersuchung abgesehen. Die Anwendung von *hide* und *show* auf Generalisierungen war bisher nicht möglich, sodass die Überprüfung des Verhaltens der Operationen auf Generalisierungen notwendig ist. Die Abbildung *Aktionen hide und show auf Generalisierungen* stellt die Auswirkungen dar. Die Generalisierung zwischen den Klassen *Person* und *Student* wird als *Student_Person* bezeichnet. In dem linken Diagramm ist diese Generalisierung sichtbar und kann durch die neu geschaffene *hide*-Aktion verborgen werden. Die beteiligten Klassen bleiben weiterhin sichtbar, während die Kante aus der Darstellung entfernt wird. Das Verhalten ist mit dem Verhalten der Aktion *hide* auf Assoziationen vergleichbar und entspricht den Erwartungen.

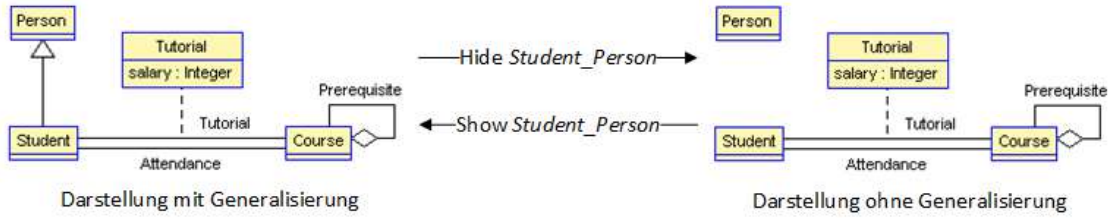


Abbildung 22: Aktionen *hide* und *show* auf Generalisierungen

Zur vollständigen und einheitlichen Umsetzung der Basisoperation wurde definiert, dass die Operationen *hide* und *crop* auf selektierte Assoziationen und Generalisierungen verfügbar sein müssen. Das angepasste Kontextmenü nach einer Selektion einer Assoziation beziehungsweise Generalisierung wird in der Abbildung *Aktionen hide und crop auf Assoziationen und Generalisierungen* gezeigt. Nach der Auswahl der Assoziation *Attendance* wird nicht nur eine *hide*-Aktion, sondern auch eine *crop*-Aktion auf *Attendance* angeboten. Wird anstatt einer Assoziation eine Generalisierung selektiert, in diesem Beispiel *Student_Person*, werden ebenfalls die beiden Basisoperationen bereitgestellt.

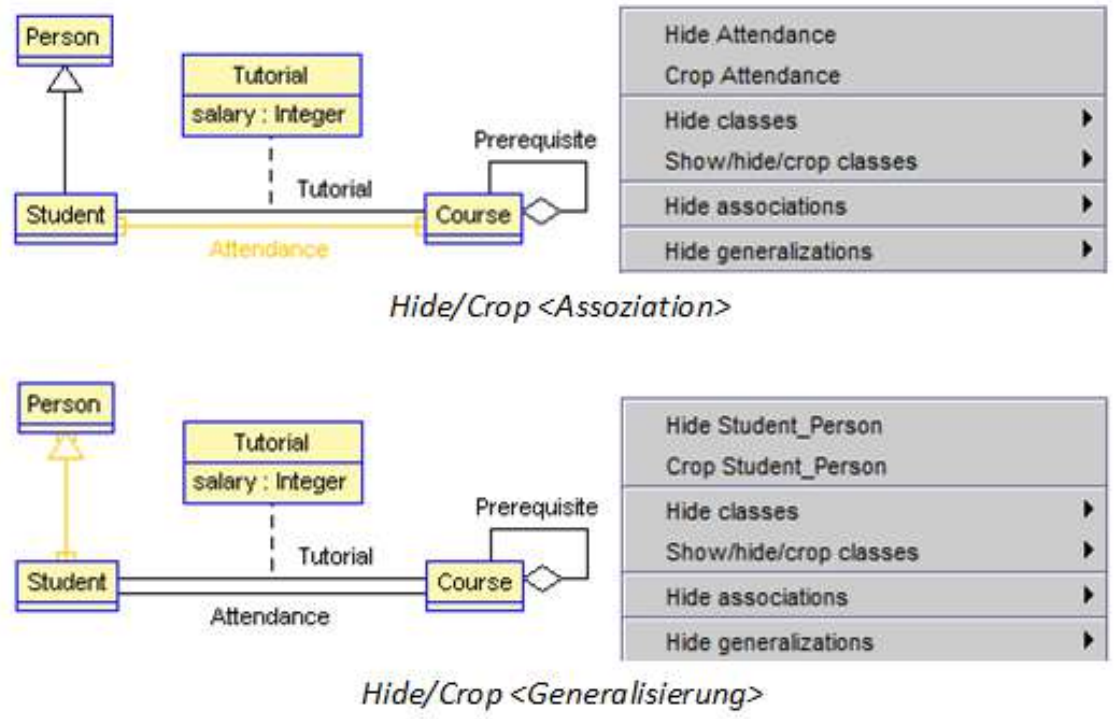


Abbildung 23: Aktionen *hide* und *crop* auf Assoziationen und Generalisierungen

Die Anwendung von *hide* auf Generalisierungen wurde bereits betrachtet. Offen ist die Kontrolle der Darstellung nach der Ausführung von *crop* auf Generalisierungen. Die Abbildung *Aktion crop auf Generalisierungen* liefert den Nachweis der korrekten Darstellung. Die Aktion *crop* wird auf die Generalisierung *Student_Person* ausgeführt. Damit die Generalisierung korrekt dargestellt werden kann, bleiben die beiden

verbundenen Klassen sichtbar und alle anderen Elemente werden ausgeblendet. Die Darstellung entspricht den Erwartungen. Die Implementierung der Basisoperationen auf Generalisierungen, wie sie im Kapitel *Analyse der USE-Version* gefordert wurde, ist somit erfolgreich abgeschlossen.

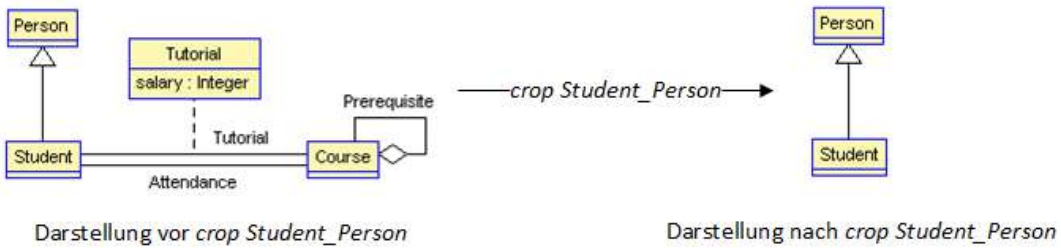


Abbildung 24: Aktion *crop* auf Generalisierungen

Das Verhalten der Aktion *crop* auf Assoziationen wurde neu implementiert, da es in der aktuellen USE-Version nicht verfügbar ist. Die Abbildung *Aktion crop auf Assoziationen* behandelt zwei Anwendungsfälle der Operation *crop*. Die linke untere Darstellung entsteht durch die Ausführung von *crop Attendance*. Um die Assoziation anzeigen zu können, werden die Klassen *Student* und *Course* benötigt. Durch die sichtbare Klasse *Course* wird zusätzlich die reflexive Assoziation *Prerequisite* eingeblendet. Insgesamt führt die Ausführung der Aktion zu zwei sichtbaren Klassen und Assoziationen. Bei der rechten unteren Darstellung wird ebenfalls *crop* auf eine Assoziation ausgeführt. Diese Assoziation verfügt über eine Assoziationsklasse. Für die Darstellung von *Tutorial* werden wieder die beiden Klassen *Student* und *Course* sowie die Assoziationsklasse *Tutorial* benötigt. Folglich werden auch die beiden Assoziationen aus der linken Darstellung angezeigt und die Aktion *crop Tutorial* versteckt nur die Klasse *Person* und die Generalisierung *Student_Person*.

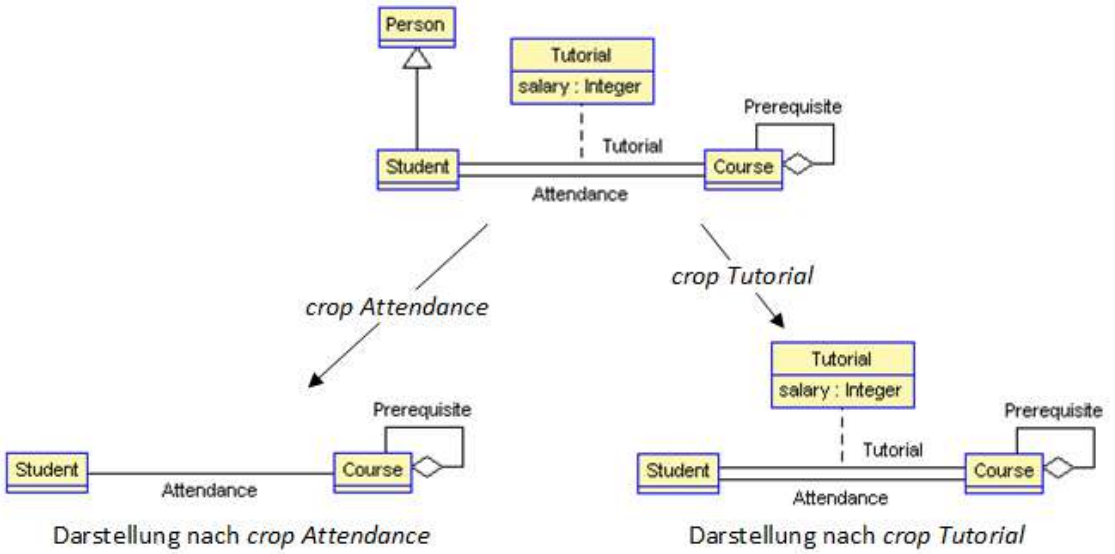


Abbildung 25: Aktion *crop* auf Assoziationen

Im Kapitel *Konzeption neuer Benutzungsmöglichkeiten* wurde die Kombination von Basisoperation und Unter- beziehungsweise Oberklassen ausgearbeitet. Die nachfolgenden Abbildungen veranschaulichen die Umsetzung dieser Anforderungen. Um Unter- oder Oberklassen bestimmen zu können, muss mindestens eine Klasse selektiert sein, die eine Generalisierung aufweist. Die neuen Benutzungsmöglichkeiten werden über das Menü *Sub/Superclasses* bereitgestellt, wenn eine Klasse mit mindestens einer Generalisierung ausgewählt wurde. Welche Aktionen angeboten werden, ist von dem Szenario abhängig. Sind beispielsweise keine Unterklassen vorhanden, werden auch keine Aktionen auf Unterklassen angeboten. Sind alle Unterklassen versteckt, ist die Aktion *Hide subclasses of <Name der Klasse>* nicht verfügbar. Die Abbildung *Menü Sub/Superclasses* präsentiert verschiedene Ausprägungen des neuen Menüs. Um das Kontextmenü nicht weiter zu überladen, werden die Aktionen auf Unter- beziehungsweise Oberklassen über ein gemeinsames Menü realisiert. Sollte sich im Verlauf der Ausarbeitung herausstellen, dass Gründe gegen eine Zusammenlegung vorhanden sind, ist eine Aufsplittung in separate Menüs denkbar.

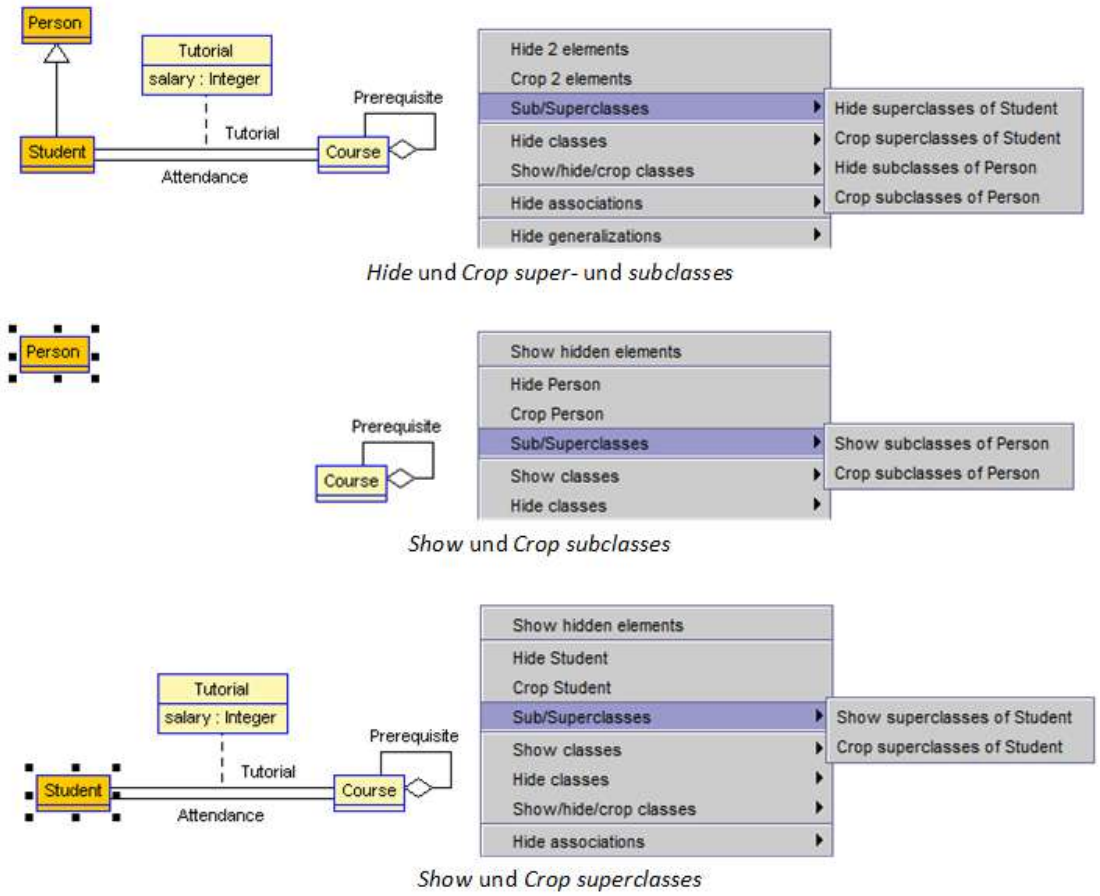


Abbildung 26: Menü *Sub/Superclasses*

Die Abbildungen *Aktionen auf Unterklassen* und *Aktionen auf Oberklassen* zeigen die Veränderungen der Darstellungen, die die Ausführung der Basisoperationen auf Unter- oder Oberklassen erzeugen. Die Operationen *hide* und *show* kehren sich in

diesen Beispielen um. Die Umkehrung der beiden Operationen ist aber nicht allgemeingültig. Zu beachten ist, dass die Aktion *crop* versteckte Elemente sichtbar werden lässt, wenn versteckte Unter- oder Oberklassen vorhanden sind. Die Klasse *Student* ist eine Unterklasse von der Klasse *Person*. In der rechten unteren Darstellung ist die Unterklasse versteckt. Nach der Ausführung von *crop* ist die Klasse wieder sichtbar. Die Aktion *Crop subclasses* nimmt alle Unterklassen in die Darstellung auf, auch wenn diese vorher versteckt waren. Die erzeugten Darstellungen entsprechen in allen Fällen dem definierten Verhalten.

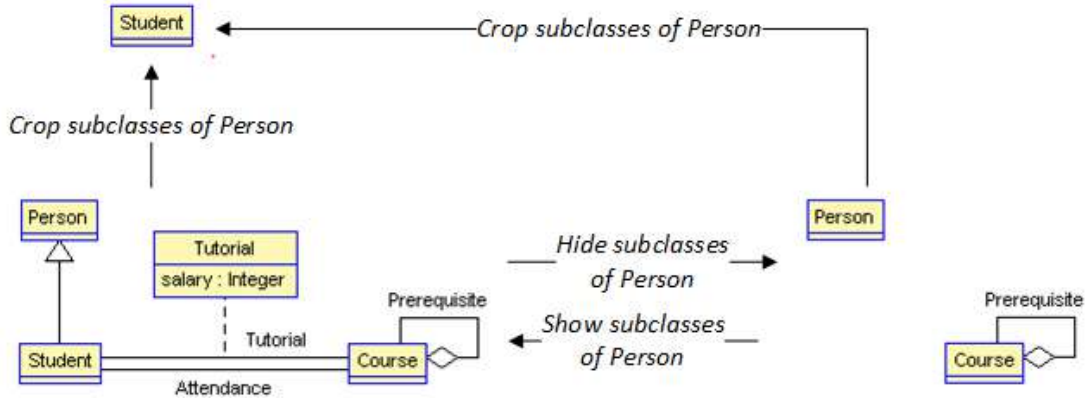


Abbildung 27: Aktionen auf Unterklassen

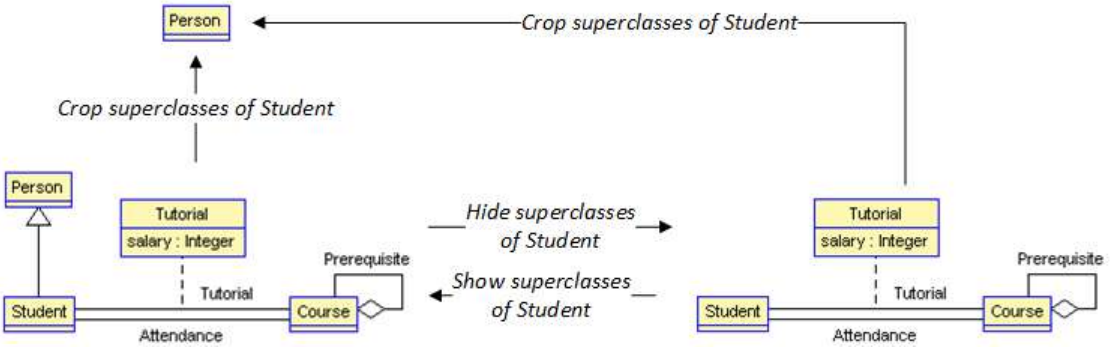


Abbildung 28: Aktionen auf Oberklassen

Von allen Anforderungen an die Ansicht Klassendiagramm sind noch zwei Fälle mit Fehlverhalten nicht korrigiert. Die Anwendung von crop auf eine Assoziationsklasse resultiert in einer Darstellung ohne Elemente. Die Ursache für dieses Verhalten konnte ermittelt und der Fehler behoben werden. Die Ausführung der Operation *crop* erzeugt, den in der Abbildung *Fehlerfall Crop < Assoziationsklasse >* im Kapitel *Analyse der USE-Version* beschriebenen Zustand. Anstatt eines leeren Diagramms werden die erwarteten Elemente angezeigt wie es im Soll-Zustand beschrieben wurde. Zudem wurde das Verhalten von *show* auf eine versteckte Assoziation mit Assoziationsklasse angepasst. Bisher führte die Aktion zu einer unvollständigen Darstellung. Die Assoziationsklasse wurde nicht sichtbar, während die Verbindungslinie zu dieser

Klasse gezeichnet wurde. Folglich endete die Linie an einer unsichtbaren Klasse und es lag eine ungültige Notation vor. Nach der Korrektur erzeugt *show* den Sollzustand, der in der Abbildung *Fehlerfall Show < Assoziation > mit Assoziationsklasse* im Kapitel *Analyse der USE-Version* zu sehen ist. Diese beiden Korrekturen schließen die Umsetzung der definierten Anforderungen ab.

Die Realisierung der Anforderungen hat zu einer vollständigen und fehlerfreien Implementierung der Basisoperationen auf Klassen, Assoziationen und Generalisierungen geführt. Die vorhandenen Fehler bei der Anwendung der Aktionen auf Klassen und Assoziationen wurden korrigiert und die vorhandenen Benutzungsmöglichkeiten auf Assoziationen und Generalisierungen erweitert. Alle Benutzungsmöglichkeiten, die auf Klassen möglich sind, sind nun auch auf Assoziationen und Generalisierungen vorhanden. Die Gestaltung der unterschiedlichen Menüs und das Verhalten der Aktionen ist vergleichbar und führt zu einer einheitlichen Darstellung und der Wiedererkennung von Strukturen. Die Tabelle *Basisoperationen Klassendiagramm nach Abschluss der Implementierung* fasst die Auswirkungen der Änderungen zusammen. Des Weiteren wurden neue Benutzungsmöglichkeiten (Aktionen auf Unter- und Oberklassen) geschaffen, die zusätzliche Optionen bei der Veränderung von Darstellungen ermöglichen.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Klasse** | vorhanden | vorhanden | vorhanden |
| **Assoziation** | vorhanden | vorhanden | vorhanden |
| **Generalisierung** | vorhanden | vorhanden | vorhanden |

Tabelle 11: Basisoperationen Klassendiagramm nach Abschluss der Implementierung

## 6.2. Ansicht Objektdiagramm

Nach der Umsetzung der Anforderungen in der Ansicht Klassendiagramm werden die Anpassungen und Erweiterungen in der Ansicht Objektdiagramm behandelt. Die Gruppierung der Umsetzung der Anforderungen entspricht der Reihenfolge aus dem vorherigen Abschnitt. Eine Gruppierung nach Objekten und Links erfolgt nicht.

Das Kontextmenü in der Ansicht Objektdiagramm wird erweitert, um weitere Interaktionen bereitstellen zu können. Die Abbildung *Neues Kontextmenü Objektdiagramm* zeigt die Implementierung der Menüs *Hide links* und *Show links*, die benötigt werden, um das Ziel der einheitlichen Anwendung der Basisoperationen zu ermöglichen. Das Menü *Hide Links* ist verfügbar, wenn in der Darstellung mindestens ein sichtbarer Link vorhanden ist und bietet das Verstecken aller sichtbaren Links, aller sichtbaren Links einer Assoziation oder eines Links an. Die Anzahl der aufgeliste-

ten Assoziationen und deren Instanzen ist abhängig von den sichtbaren Links in der Darstellung. Die Aktion *show* auf Links ist in der aktuellen USE-Version nicht verfügbar, was die Umkehrung der Operation *hide* erschwert. Dieser Missstand wird mit der Einführung des Menüs *Show links* behoben. Das Menü wird nur angeboten, wenn mindestens ein versteckter Link vorhanden ist. Bei der Gestaltung orientiert sich das Menü an dem Menü *Hide classes*, um die Wiedererkennung von Strukturen und eine einfache Bedienung zu fördern.

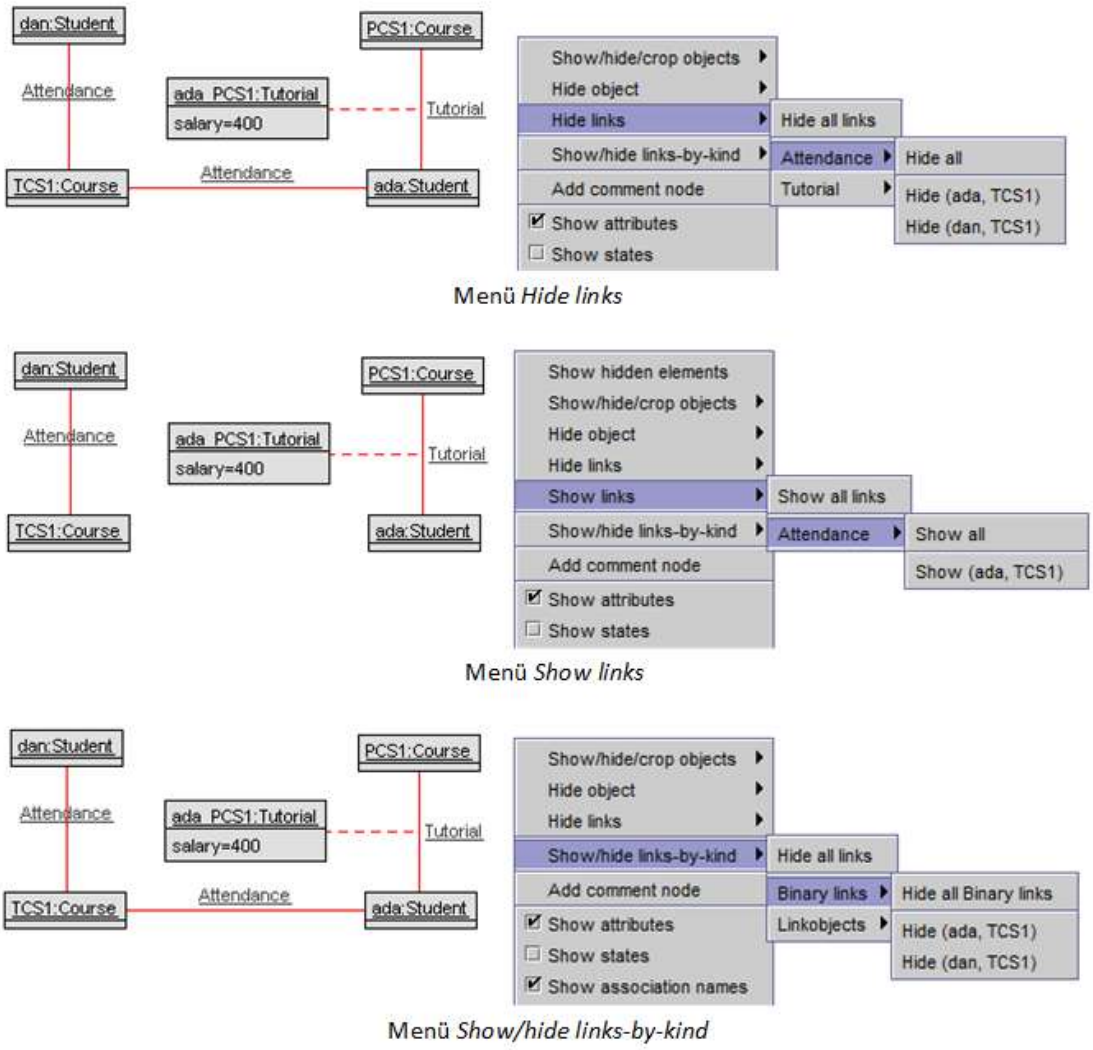


Abbildung 29: Neues Kontextmenü Objektdiagramm

Das Verstecken von Links wurde bereits in der Analysephase auf korrektes Verhalten untersucht. Die Umkehrung durch die Ausführung von *show* auf versteckte Links war bisher nicht möglich. Daher ist zu prüfen, ob *show* die Wirkung von *hide* umkehrt und somit korrekt implementiert wurde. Die Abbildung *Aktionen hide und show auf Links* zeigt das Verhalten von *show*. Die Aktion *show* kann fehlerfrei auf Links und Links mit Linkobjekten ausgeführt werden. Damit ist die Implementierung von *show* auf Links abgeschlossen.

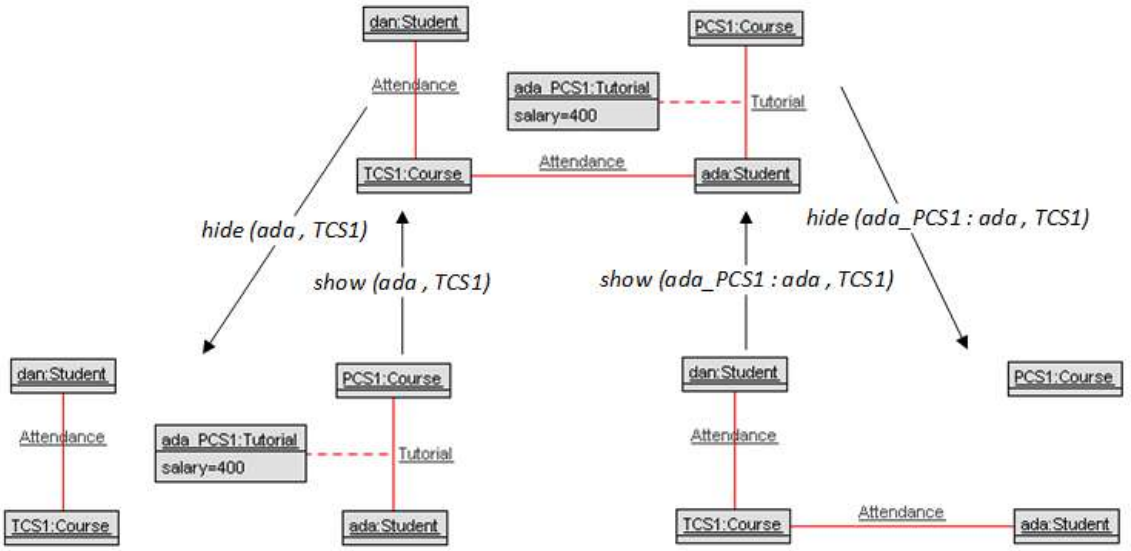


Abbildung 30: Aktionen *hide* und *show* auf Links

Die Abbildung *Neues Kontextmenü Objektdiagramm* zeigt noch ein weiteres Menü: *Show/hide links-by-kind*. Über dieses Menü können ebenfalls *hide*- und *show*-Aktionen auf Links ausgelöst werden. Die Gliederung unterscheidet sich allerdings von den in diesem Abschnitt bereits vorgestellten Menüs. Einzelne Links werden nicht unter ihrer Assoziation zusammengefasst, sondern unter der Art der Assoziation. Der Link *(ada, TCS1)* ist eine Instanz der Assoziation *Attendance*. *Attendance* wiederum ist eine binäre Assoziation. Folglich wird der Link *(ada, TCS1)* der Assoziationsart *Binary links* zugeordnet. Weitere binäre Links würden ebenfalls diesem Typ zugeordnet werden. Dabei müssen die weiteren Links nicht unbedingt Instanzen von *Attendance* sein. Das Menü bietet die Möglichkeit Aktionen auf alle Links, alle Links einer Assoziationsart oder gezielt auf einen Link auszuführen. Diese neue Funktion ordnet Links in die folgenden sieben Kategorien ein:

- *Derrived links*: Abgeleitete Links werden unter *Derrived links* zusammengefasst.
- *Linkobjects*: Links, die mit einem Linkobjekt verbunden sind, werden in dieser Kategorie eingeordnet.
- *nAryLinks*: Links, die über mehr als zwei Endpunkte verfügen, werden unter *nAryLinks* kategorisiert.
- *Reflexiv links*: Instanzen von zirkulären Assoziationen werden diesem Typ zugeordnet.
- *Binary links*: Links, die zwei Objekte miteinander verbinden und keine Aggregation oder Komposition sind, werden unter *Binary links* gruppiert.
- *Aggregation*: Links, die eine Aggregationsbeziehung abbilden.

- *Compositon*: Links, die eine Kompositionsbeziehung abbilden.

Wenn keine Instanz eines Typs vorhanden ist, wird der Typ im Menü nicht angezeigt. In der Abbildung *Neues Kontextmenü Objektdiagramm* sind zwei Links der Assoziation *Attendance* und ein Link der Assoziation *Tutorial* vorhanden. Bei *Tutorial* handelt es sich um eine Assoziation mit Assoziationsklasse. Da zwei unterschiedliche Arten von Assoziationen vorliegen, werden zwei und nicht sieben Assoziationsarten im Menü angezeigt. Diese neue Funktion ergänzt die vorhandenen Interaktionen auf Links und ermöglicht neue Auswahl- und Filteroptionen.

Nach der Vorstellung von verschiedenen Möglichkeiten die Basisoperationen *hide* und *show* auf Links nutzen zu können, wird nun die Anwendung von *crop* behandelt. Diese Operation auf Links ist in der aktuellen Version von USE nicht vorhanden. Daher wird das Kontextmenü um die Aktion *crop <Name des Links>* erweitert, wenn ein Link selektiert wurde. Die neue Interaktion entspricht von der Benennung und der Menüeinbindung den bereits vorhandenen *hide*- und *crop*-Aktionen und wird daher nicht dargestellt. Die Abbildung *Aktion crop auf Links* verdeutlicht das implementierte Verhalten. Da bei einigen Anwendungen von *crop* in der Analyse ein Fehlverhalten festgestellt wurde, werden zwei Anwendungsfälle betrachtet. Beim ersten Fall wird *crop* auf einen Link ohne Linkobjekt ausgeführt. Die Aktion *crop (ada, TCS1)* wird als Beispiel verwendet. In der entstehenden Darstellung sind neben dem Link, die beiden Linkenden *TCS1* und *ada* sichtbar. Im zweiten Fall wird *crop* auf einen Link mit Linkobjekt angewendet. Neben den Links sind die drei Klassen für eine vollständige Notation sichtbar. Die beiden erzeugten Diagramme weisen keine Abweichungen zu den Erwartungen auf. Die Implementierung von *crop* auf Links wird als abgeschlossen angesehen.

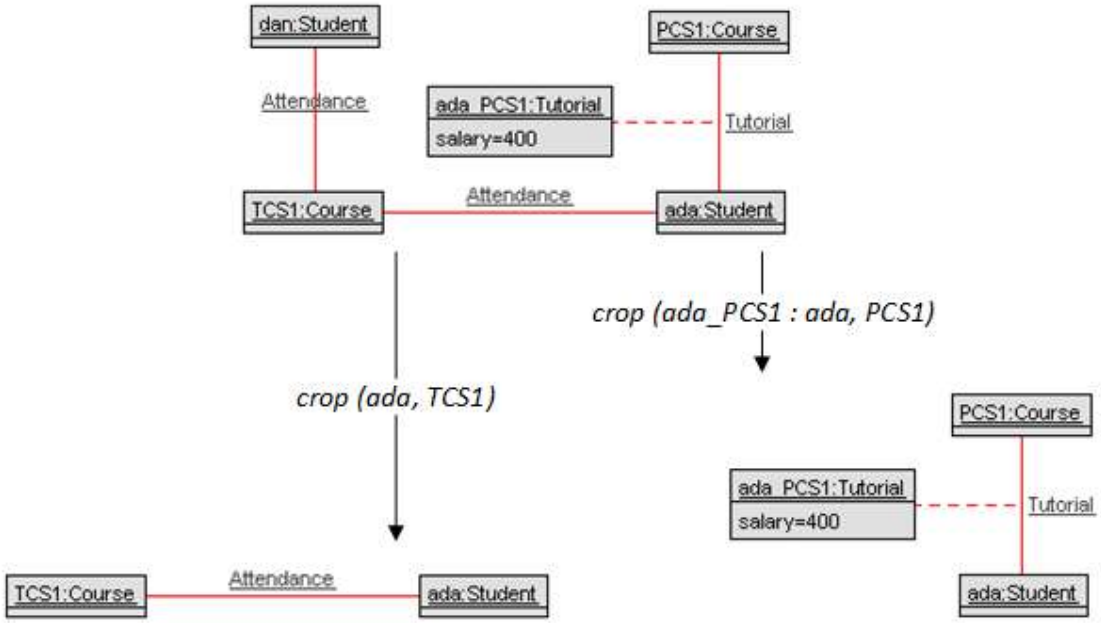


Abbildung 31: Aktion *crop* auf Links

Neben den Erweiterungen der Anwendungsmöglichkeiten auf Links wurde in der Konzeptionsphase die Einführung eines dritten Zustands angeregt. Der neue Zustand soll eine Zwischenstufe zwischen den bisherigen Sichtbarkeiten *Versteckt* und *Sichtbar* bilden und das Ausgrauen von Objekten ermöglichen. Laut Anforderung soll die neue Aktion nur auf Objekte möglich sein, daher wird das Menü nur erweitert, wenn ein Objekt selektiert wurde. Die neue Aktion wird unter der Bezeichnung *Grey out <Name des Objekts>* in das Kontextmenü eingebunden. Die Abbildung *Ausgrauen von Objekten* zeigt die Veränderung der Darstellung und deren Umkehrung. Der Kontrast der Hintergrundfarbe wird bei einem ausgegrauten Objekt im Vergleich zu dem Hintergrund in der Ansicht Objektdiagramm abgeschwächt und die Rahmenlinie entfernt. Die Darstellung des Objekts hebt sich nicht stark vom Hintergrund ab, wodurch der Fokus auf dem Objekt mit der normalen Darstellung liegt. Auf ein ausgegrautes Objekt wird die Aktion *Grey out* nicht angeboten, da die Aktion keine Veränderung der Darstellung zur Folge hat. Statt der Aktion wird die Aktion *Grey in <Name des Objekts>* bereitgestellt, um das Objekt in den ursprünglichen Zustand zu überführen.

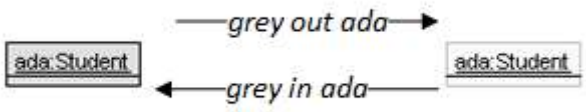


Abbildung 32: Ausgrauen von Objekten

Die letzte Anforderung für die Ansicht Objektdiagramm, abgesehen von den übergreifenden Änderungen, ist die Korrektur des Fehlverhaltens der Aktion *crop* bei der Anwendung auf Linkobjekte. In der aktuellen Version resultiert eine entsprechende Anwendung in einer Darstellung ohne sichtbare Elemente. Die Ursache für das fehlerhafte Verhalten konnte gefunden und korrigiert werden, sodass *crop <Name des Linkobjekts>* die gewünschte Darstellung aus der Abbildung *Fehlerfall Crop < Linkobjekt >* im Kapitel *Analyse der USE-Version* erzeugt.

Die Realisierung der Anforderungen vervollständigt und vereinheitlicht die Benutzungsmöglichkeiten der Basisoperationen auf Objekte und Links. Die zuvor fehlerhaften Aktionen wurden korrigiert und durch neue Interaktionen ergänzt. Speziell die Anwendungsmöglichkeiten der drei Operationen auf Links wurden verbessert. Die Tabelle *Basisoperationen Objektdiagramm nach Abschluss der Implementierung* fasst die implementierten Basisoperationen zusammen und deutet die vollständige Implementierung der Basisoperationen auf Objekte und Links an. Die neu geschaffenen Funktionen sind in der Tabelle nicht erfasst, erweitern aber zusätzlich die Möglichkeiten eine Darstellung verändern zu können.

| | **Aktion hide** | **Aktion show** | **Aktion crop** |
|---|---|---|---|
| **Objekt** | vorhanden | vorhanden | vorhanden |
| **Link** | vorhanden | vorhanden | vorhanden |

Tabelle 12: Basisoperationen Objektdiagramm nach Abschluss der Implementierung

## 6.3. Übergreifende Änderungen

In diesem Kapitel wurde bereits die Umsetzung einiger Erweiterungen und Fehlerkorrekturen beschrieben. Alle Veränderungen hatten Auswirkungen auf Interaktionen oder Verhaltensweisen in der Ansicht Klassendiagramm oder in der Ansicht Objektdiagramm. Die Realisierung der Anforderungen, die in beiden Ansichten umzusetzen sind, steht noch aus und wird in diesem Abschnitt nachgeholt. Die folgenden Anforderungen werden implementiert:

- Das Kontextmenü verfügt über einige Untermenüs, deren Einträge dynamisch wachsen können. Die Anzahl der Einträge ist abhängig von der Anzahl der Elemente. Bei einer großen Anzahl von Einträgen können nicht alle Einträge angezeigt werden. Das Menü erscheint abgeschnitten.

- Die bisher verwendete alphabetische Sortierung soll durch eine alphanumerische Sortierung ersetzt werden.

- In den bisher gezeigten Beispielen wurde oftmals nur ein Element selektiert und das Kontextmenü auf die korrekte Anzeige von Aktionen untersucht. Bei der

Auswahl mehrerer Elemente von unterschiedlichen Typen werden Aktionen, die eine Selektion erfordern, nicht vollständig angezeigt. Werden Assoziationen und Klassen selektiert, bietet das Kontextmenü die Aktionen *hide* und *crop* nur auf Klassen an. In der Ansicht Objektdiagramm werden nur Aktionen auf Objekte bereitgestellt, auch wenn Objekte und Links selektiert wurden.

- Die Bezeichnung der Aktionen *Show hidden classes and associations* in der Ansicht Klassendiagramm und *Show hidden objects* in der Ansicht Objektdiagramm sollen durch *Show hidden elements* ersetzt werden.

Zuerst werden die untersten beiden Anforderungen aus der Aufzählung umgesetzt, da für die anderen beiden Anforderungen ein neues Szenario benötigt wird. Mit den bisher verwendeten Szenarien kann das Fehlverhalten nicht reproduziert und die Korrektur getestet werden. Das fehlerhafte Verhalten bei der Auswahl mehrerer Elemente wird in zwei Schritten gelöst und an der Ansicht Klassendiagramm verdeutlicht. Im ersten Schritt werden die Bezeichnungen der Aktionen angepasst. Anstatt *Hide x classes* und *Crop x classes* werden die Interaktionen *Hide x elements* und *Crop x elements* angeboten. Das $x$ ist ein Platzhalter für natürliche Zahlen, die gleich oder größer als die Zahl 2 sein können. Durch die Verwendung des Begriffs *elements* ist eine Unterscheidung und separate Auflistung von Klassen, Assoziationen und Generalisierungen nicht notwendig. Bei der Selektion eines Elements, wird weiterhin der Name des Elements verwendet. Im zweiten Schritt wird das Verhalten der Aktionen angepasst, um die gewählte Aktion auf alle selektierten Elemente auszuführen. Die Abbildung *Aktionen auf mehrere selektierte Elemente* zeigt die Umsetzung der Anforderung für die Aktionen *hide* und *crop* in der Ansicht Klassendiagramm. Es wurde die Klasse *Course* und die Assoziation *Attendance* selektiert. Die Ausführung von *Hide 2 elements* beziehungsweise *Crop 2 elements* verändert die Darstellung. Im Gegensatz zur vorherigen USE-Version wird die ausgewählte Assoziation bei der Veränderung der Darstellung beachtet und eine entsprechende Operation ausgelöst. Das dargestellte Verhalten bei der Benennung von Menüeinträgen und bei der Ausführung entspricht der gestellten Anforderung und wird ohne Abweichungen in die Ansicht Objektdiagramm übertragen. Die Implementierung in der Ansicht Objektdiagramm wird nicht dargestellt, da sie der gezeigten Methodik entspricht.

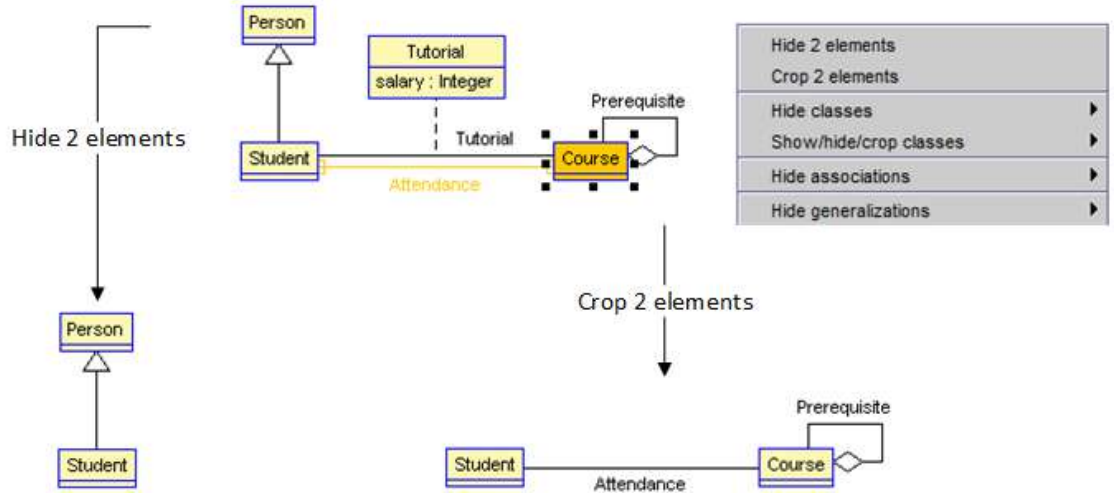


Abbildung 33: Aktionen auf mehrere selektierte Elemente

Des Weiteren wurden die beiden Aktionen *Show hidden classes and associations* in der Ansicht Klassendiagramm und *Show hidden objects* in der Ansicht Objektdiagramm durch *Show hidden elements* ersetzt. Die Aktion überführt alle versteckten Elemente in die Darstellung. Je nach Diagramm und Szenario können Klassen, Assoziationen, Assoziationsklassen, Generalisierungen, Objekte, Links oder Linkobjekte wieder sichtbar werden. Sobald mindestens ein verborgenes Element vorhanden ist, wird diese Interaktion angeboten. Die Abbildung *Show hidden elements* zeigt mögliche Ausprägungen eines Kontextmenüs. Es wurden alle Elemente aus den vorherigen Beispielen für Klassen- und Objektdiagramme versteckt. Durch das Verstecken aller Elemente ist die Bedingung, dass mindestens ein Element verborgen sein muss erfüllt. Des Weiteren bieten die beiden gezeigten Kontextmenüs dem Anwender ausschließlich die Operation *show* an. Das Verhalten von *show* auf die unterschiedlichen Elemente wurde bereits geprüft und auf eine erneute Prüfung wird verzichtet.

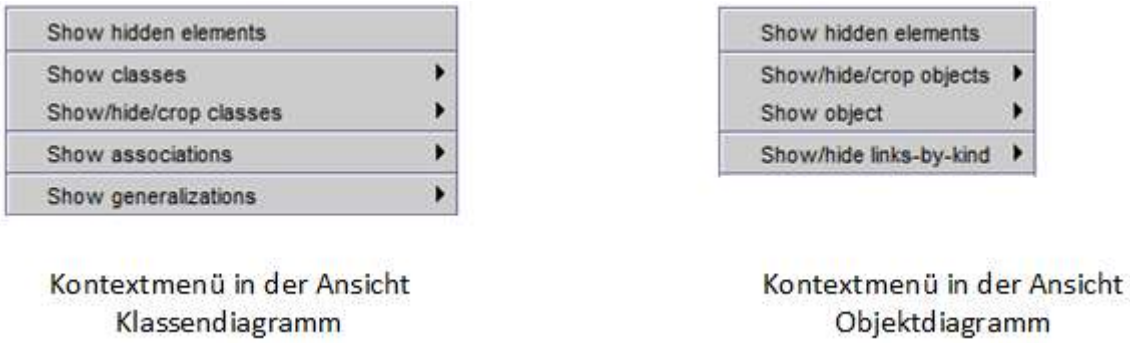


Abbildung 34: *Show hidden elements*

Für die Implementierung und den Test der nächsten beiden Funktionen wird ein angepasstes Beispiel mit mehreren Elementen benötigt. Das neue Beispiel dient als Testszenario für die Umsetzungen der alphanumerische Sortierung und den Korrekturen von den Menüdarstellungen, wenn eine Vielzahl von Einträgen vorhanden ist.

Die Umsetzung der Anforderungen wird in der Ansicht Objektdiagramm vorgestellt. Dazu werden 30 Objekte der Klasse *Student* erzeugt, die fortlaufend als *Student1* bis *Student30* bezeichnet werden. Die Abbildung *Sortierung und Scrolling* zeigt das Menü *Hide object* für die Klasse *Student*. Mit den Pfeilen kann in dem Menü nach oben oder unten navigiert werden. Dieses Feature wird aktiv, wenn ein Menü über 20 oder mehr Einträge verfügt und verhindert das Einträge abgeschnitten werden. Die Implementierung erfolgt für alle Menüs, die dynamische Einträge enthalten und somit die Anzahl der Einträge nicht vorhersehbar ist. Die technische Umsetzung basiert auf einer Lösung von *Darryl Burke*[5] und wird in die Ansicht Klassendiagramm übertragen.

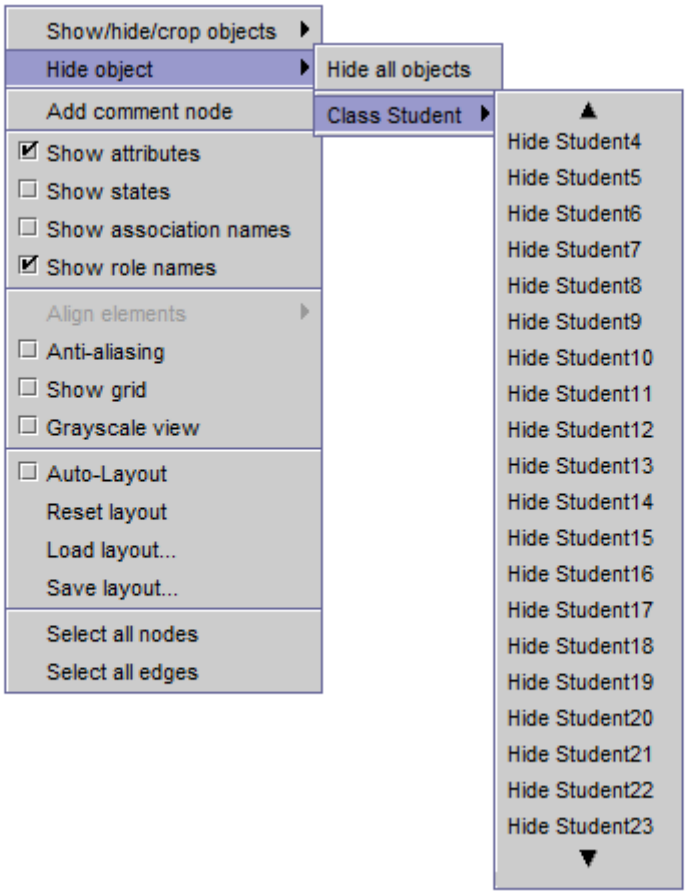


Abbildung 35: Sortierung und Scrolling

Des Weiteren dient die Abbildung als Nachweis für die Implementierung der alphanumerischen Sortierung. Die Reihenfolge der Aufzählung der einzelnen Studenten-Objekte entspricht der Anforderung. Als Beispiel für die neue Sortierung kann die Reihenfolge der Objekte *Student9* und *Student10* betrachtet werden. Das Objekt *Student10* wird nach dem Objekt *Student9* gelistet. Bei einer alphabetischen Sortierung müsste *Student9* als letztes Element der 30 erzeugten Objekte gelistet werden.

[5] `https://tips4java.wordpress.com/2009/02/01/menu-scroller/ (zuletzt aufgerufen am 27.12.2018)`

Für die Realisierung wurde auf einem Comparator von *Dave Koelle*[6] zurückgegriffen, der angepasst wurde und für die Sortierung von Elementen in USE verwendet werden kann. Die Reihenfolge im Kontextmenü bleibt unberührt, da dieses Funktion nur auf Einträge in Untermenüs angewendet wird.

## 6.4. Zusammenfassung

Die einzelnen Abschnitte in diesem Kapitel beschreiben die Umsetzung der definierten Anforderungen. Es wurden Fehlerkorrekturen, Verbesserungen vorhandener Funktionen und neu entwickelte Benutzungsmöglichkeiten vorgestellt. Die vorgestellten Implementierungen führten zu Veränderungen in den Ansichten Klassendiagramm und Objektdiagramm. In diesem Abschnitt werden die Einstiegsfragen aus dem Kapitel *Analyse der USE-Version* aufgegriffen und unter Berücksichtigung der getätigten Änderungen erneut betrachtet.

Das wiederkehrende Schema, das durch die drei Basisoperationen vorgegeben wird, wurde durch die durchgeführten Anpassungen gestärkt. Das fehlerhafte Verhalten bei der Anwendung einiger Operationen wurde korrigiert, sodass die Operationen die erwarteten Darstellungen erzeugen. Zusätzlich wurde das Schema durch die Einführung und Anpassung der Benutzungsmöglichkeiten im Kontextmenü vereinfacht, vervollständigt und angeglichen. Auf alle betrachteten Konzepte (Klassen, Assoziationen, Generalisierungen, Objekte und Links) sind die drei Basisoperationen in gleicher Weise verfügbar. Nach der Selektion eines Elements werden die Operationen *hide* und *crop* angeboten. Die Unterschiede, dass einige Aktionen nur auf einige Konzepte vorhanden waren, wurden aufgehoben. Des Weiteren existieren *hide*- und *show*-Menüs im Kontextmenü, die unabhängig von einer Selektion sind und die genannten Operationen anbieten. Durch die vollständige Implementierung wurde sichergestellt, dass eine veränderte Darstellung in ihren ursprünglichen Zustand überführt werden kann. Je nach Zustand ist das über eine oder eine Kombination von mehreren Aktionen möglich.

Zusätzlich zur einheitlichen und vollständigen Anwendung wurden neue Benutzungsmöglichkeiten entwickelt, die mehr Optionen bei Anpassungen von Diagrammen ermöglichen. Einige der neuen Interaktionen greifen auf die Basisoperationen zurück, was die Bedeutung der drei Operationen unterstreicht. Bei den neu entworfenen und implementierten Funktionen handelt es sich um spezialisierte Funktionen für eine Ansicht, die nicht auf die andere Ansicht übertragen werden konnte. Beispielsweise setzt die Anwendung der Basisoperation auf Unterklassen eine vorhandene Generalisierung voraus. Bei Generalisierungen handelt es sich um ein Konzept,

[6] `http://www.davekoelle.com/alphanum.html (zuletzt aufgerufen am 27.12.2018)`

das in Klassendiagrammen vorhanden ist, aber nicht in gleicher Form in Objektdiagrammen. Daher wird das Feature der Anwendung der Basisoperationen auf Unterklassen nur in der Ansicht Klassendiagramm angeboten. Diese Problematik betrifft bis auf eine Ausnahme alle neu konzeptionierten Funktionen. Das Ausgrauen von Objekten ist auch auf andere Elemente übertragbar, wurde im Kontext dieser Ausarbeitung aber nur für Objekte umgesetzt. Bevor der neu geschaffene Zustand für andere Elemente verfügbar ist, ist festzustellen, ob diese neue Funktion eine sinnvolle Ergänzung zu den vorhandenen Zuständen *Versteckt* und *Sichtbar* darstellt.

Es wurden nicht nur vorhandene Funktionen korrigiert und neue Interaktionen geschaffen, sondern auch vorhandene Benutzungsmöglichkeiten verbessert. Zwei Beispiele dafür sind die Umsetzung der alphanumerischen Sortierung und die Einführung des Scrolling in einigen Menüs. Die alphanumerische Sortierung listet Elemente in einer leicht nachvollziehbaren Reihenfolge auf. Die Implementierung des Scrollings in einigen Menüs verhindert nicht nur abgeschnittene Menüs durch die Reduzierung der gleichzeitig angezeigten Einträge, sondern unterbindet auch lange Auflistungen.

Zusammengefasst wurde durch die Umsetzungen der Anforderungen ein Mehrwert geschaffen. Es sind weniger Fehler in den Operationen vorhanden, neue Funktionen geschaffen und vorhandene Interaktionen verbessert worden. Offen ist jedoch, ob die Anpassungen und vor allem die neuen Benutzungsmöglichkeiten eine effizientere Veränderungen von Darstellungen ermöglichen, wenn eine Vielzahl von Elementen vorliegt. Die in diesem Kapitel verwendeten Darstellungen verfügen über maximal 30 Elemente. Modellierungen können durchaus eine vielfache Anzahl an Elementen aufweisen. In dem Kapitel *Grundlagen* wurde ein stark vereinfachtes Modell von einem Wald erwähnt. Ein Wald verfügt über eine Vielzahl unterschiedlicher Bewohner und Vegetationen sowie weitere Elemente und Eigenschaften. Die Anzahl der zu erzeugenden Instanzen, um einen Wald zu modellieren, wären um ein Vielfaches größer als die abstrakte Darstellung. Die Frage, ob die geschaffenen Benutzungsmöglichkeiten auch für die Veränderungen von größeren Szenarien geeignet und effizient anwendbar sind, wurde bisher nicht beantwortet.

# 7. Evaluation

In den bisherigen Kapiteln wurden übersichtliche Darstellungen von Diagrammen für die Illustration von Gestaltungen und Verhaltensweisen verwendet. In diesem Kapitel werden zwei Beispiele genutzt, die über eine Vielzahl von Elementen, im Vergleich zu den bisherigen Beispielen, verfügen. Diese neuen Beispiele bilden die Grundlage für die Evaluation, bei der Kenntnisse gewonnen werden sollen, ob die geschaffenen Auswahl- und Filteroptionen für Darstellungen mit vielen Elementen geeignet sind.

Bei der Evaluation steht die Nutzbarkeit der Funktionen im Vordergrund. Für die Evaluation in der Ansicht Klassendiagramm wird das bereits vorgestellte UML-Metamodell verwendet, während in der Ansicht Objektdiagramm eine Darstellung auf Basis der *StudentWorld* generiert wird. Die erzeugten Darstellungen sind durch teilweise und vollständige Überdeckungen von Elementen geprägt. Ein Element wird als überdeckt bezeichnet, wenn die Notation eines Elements durch die Notation eines anderen Elements verdeckt wird. Mit steigender Anzahl von Elementen auf einer gleichbleibenden Fläche zur Darstellung, steigt die Wahrscheinlichkeit der Existenz von überdeckten Elementen. Während in den vorherigen Kapiteln auf Grund der Übersichtlichkeit und Nachvollziehbarkeit der Beispiele die Positionierung der Elemente verändert wurde, sodass keine Überdeckungen vorliegen, wird in diesem Kapitel darauf verzichtet. Das Ergebnis der Teilevaluationen der Ansichten Klassen- und Objektdiagramm wird in dem Abschnitt *Ergebnisse der Evaluation* zusammengeführt.

## 7.1. Ansicht Klassendiagramm

Viele Benutzungsmöglichkeiten in USE basieren auf den drei Basisoperationen *hide*, *show* und *crop*. Im Kontext dieser Arbeit wurde die Anwendung dieser Operationen auf Assoziationen und Generalisierungen an die bestehenden Möglichkeiten auf Klassen angeglichen. Das Ziel dieser Maßnahme ist die einheitliche und vollständige Umsetzung der drei Aktionen auf die genannten Konzepte. Zusätzlich zur Übertragung der Interaktionen der Klassen auf die anderen Konzepte wurden zwei neue Anwendungsmöglichkeiten (Operationen auf Unter- und Oberklassen) geschaffen. Bisher wurden jedoch noch keine Erkenntnisse gewonnen, ob die Stärkung des Konzeptes der Basisoperationen und die neu geschaffenen Möglichkeiten geeignete Auswahl- und Filteroptionen darstellen. Dies soll mit Hilfe des UML-Metamodels untersucht werden.

Die Abbildung *UML-Metamodell* zeigt das Modell in Form eines Klassendiagramms. Die Darstellung enthält unter anderem 63 Klassen und 98 Beziehungen. Allerdings sind nicht alle Elemente in der Abbildung sichtbar, da einige Elemente von anderen Elementen überdeckt werden. Die vorhanden Invarianten, Operationen sowie die Vor- und Nachbedingungen sind für die Evaluation nicht von Bedeutung, da die geschaffenen Benutzungsmöglichkeiten ausschließlich auf die Darstellung der Klassen und ihrer Beziehungen Auswirkungen haben. Das zugrundeliegende Modell wurde nicht im Kontext dieser Ausarbeitung angefertigt. Es ist in der USE-Version 5.0.1 als Beispiel enthalten und modelliert die UML-Spezifikation in der Version 2.3. Die Abweichungen zu der aktuellen UML-Spezifikation sind nicht von Belang, da die Anzahl der Elemente für die Evaluation ausschlaggebend ist. Durch die Verwendung des Metamodells wird sichergestellt, dass die getätigten Anpassungen nicht nur für ein auf die Evaluation zugeschnittenes Beispiel geeignet sind.

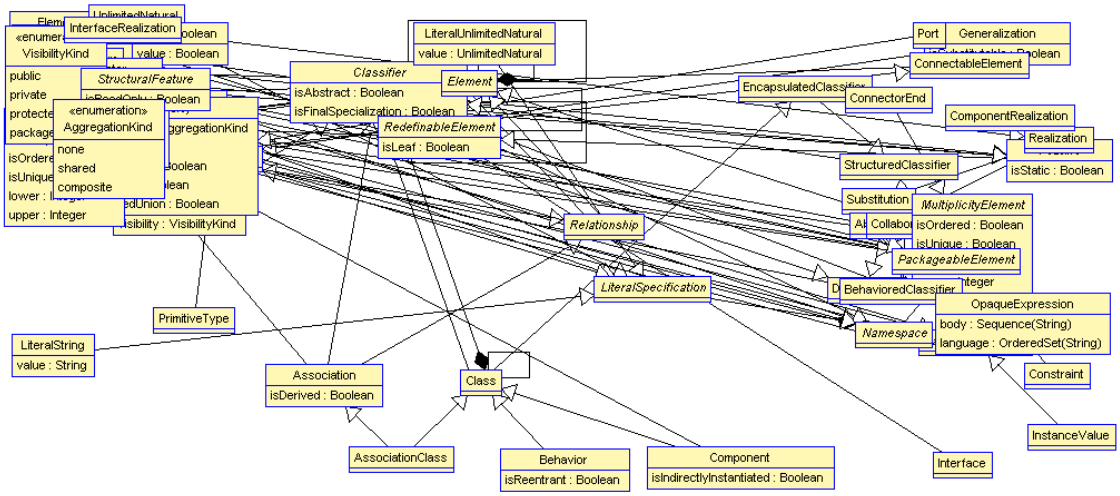


Abbildung 36: UML-Metamodell

Die Darstellung entstand durch das Einlesen einer USE-Spezifikation und das UML-Metamodell abbildet. Die teilweisen oder vollständigen Überdeckungen von Elementen können bei Darstellungen mit vielen Elementen in USE gehäuft auftreten. Die Überdeckung ist ein Problem, dessen Auswirkung durch die Anwendung von Auswahl- und Filteroptionen reduziert werden kann, sofern die Optionen nutzbar sind.

Zuerst wird die Anwendung der Basisoperationen auf Klassen, Assoziationen und Generalisierungen betrachtet. Im Kapitel *Implementierung und Anpassung der Benutzungsmöglichkeiten* wurde bereits die fehlerfreie Anwendung und die vollständige Umsetzung gezeigt, sodass der Fokus der Betrachtung auf die Nutzbarkeit der Benutzungsmöglichkeiten liegt. Es wird jede der drei Basisoperationen einmal betrachtet. Auf Grund der einheitlichen Anwendung ist einer Unterscheidung der Anwendung der Operationen auf Klassen, Assoziationen und Generalisierungen nicht notwendig.

Das Ergebnis einer Anwendung kann auf die anderen Elemente übertragen werden. Da drei Konzepte und drei Basisoperationen existieren, wird jede Operation auf ein anderes Konzept angewendet. Die Betrachtung der neu geschaffenen Funktionen erfolgt später in diesem Abschnitt.

Das Verstecken von Elementen ist durch die Basisoperation *hide* möglich und wird über zwei Optionen ermöglicht. Das Kontextmenü enthält für jedes Konzept ein Untermenü, über das die Operation *hide* auf Elemente des Konzepts initiiert werden kann. Zusätzlich wird das Kontextmenü um eine entsprechende Option erweitert, wenn ein Element selektiert wurde. Für die Evaluation soll die Klasse *Parameter* versteckt werden. Die Auswahl dieses Elements erfolgte willkürlich und ist mit keinen Bedingungen verbunden. Jede andere Klasse, Assoziation oder Generalisierung wäre ebenfalls als Testfall geeignet.

Zunächst wird versucht die Klasse zu selektieren und anschließend über die Option *Hide Parameter* zu verstecken. Die Voraussetzung für eine Selektion ist, dass die Klasse sichtbar ist und der Anwender die Position der Klasse in der Darstellung kennt. Nach dem Einlesen einer UML-Spezifikation sind alle Elemente sichtbar, sodass die gesuchte Klasse ebenfalls sichtbar sein muss. Damit ist die erste Voraussetzung erfüllt. Die Position der Klasse wird für die Durchführung der Selektion benötigt und ist nicht zu bestimmen. Die Klasse *Parameter* befindet sich oben links in der Darstellung, ist aber von anderen Elementen komplett verdeckt. Eine Selektion und das damit verbundene Verstecken der Klasse ist nicht möglich. Würde der Testfall das Verstecken eines Elements in dem unteren Teil der Abbildung anfordern, wäre eine Selektion möglich. Die Schwierigkeit die Position des gewünschten Elements zwischen den vielen Elementen zu finden, bleibt jedoch bestehen. Zusammengefasst stellt das notwendige Finden des gesuchten Elements eine Schwierigkeit dar, wenn eine große Anzahl von Elementen in der Darstellung enthalten sind. Das Verstecken der Klasse *Parameter* ist mit dieser Option nicht möglich. Das Ergebnis ist auf Assoziationen und Generalisierungen übertragbar, da diese sich ebenfalls in hohem Maß überschneiden und nicht selektiert werden können.

Das Verstecken über das Menü *Hide classes* bildet die zweite Möglichkeit zur Realisierung der gewünschten Anforderung. Das Menü enthält eine Vielzahl von Einträgen, die alphanumerisch sortiert aufgelistet werden. Die Sortierung erleichtert das Finden der gesuchten Option. Durch die Einführung des scrollbaren Menüs und der Beschränkung der gleichzeitig angezeigten Einträge sind die dargestellten Optionen übersichtlich. Der Anwender navigiert zu der gewünschten Option und kann die Aktion *Hide Parameter* initialisieren. Die Sortierung und die Scroll-Optionen erleichtern das Finden der gewünschten Interaktion und ermöglichen das Verstecken

der Klasse.

Insgesamt ist das Verstecken von Elementen mit den vorhandenen Optionen möglich. Allerdings wurde das Ausführen der Aktion durch mehrere Umstände erschwert. Die Option, die eine Selektion voraussetzt, konnte nicht verwendet werden. Das zu selektierende Element ist in der Darstellung vollkommen von anderen Elementen überdeckt. Auch ohne die anderen Klassen würden Enumerationen einen Großteil der Darstellung der Klasse *Parameter* und deren Auswahl erschweren. Auf Enumerationen haben die Basisoperationen keine Auswirkungen und können deren Sichtbarkeit nicht verändern. Diese Überdeckung kann nur durch manuelles Verschieben der Elemente aufgelöst werden. Das Ausführen von Operationen auf selektierte Elemente ist ein effizientes Konzept, wenn keine Überlagerungen vorhanden sind und die Position der gewünschten Elemente erkennbar ist. Über die Aktion *Hide Parameter* des Untermenüs *Hide classes* konnte die Anforderung dennoch umgesetzt werden. Die implementierten Verbesserungen erleichterten die Bedienung und Navigation zu der gesuchten Aktion. Allerdings kann über das Untermenü eine Aktion nur auf alle oder genau ein Element eines Konzepts ausgeführt werden. Um die vorhanden Überlagerungen durch eine Reduzierung der sichtbaren Klassen aufzuheben, wäre eine mehrmalige Wiederholung dieser Interaktion erforderlich. Ein Auswahl von mehreren Elementen ist über das Untermenü nicht möglich.

Als Nächstes wird die Nutzbarkeit der Basisoperation *show* am Beispiel einer Assoziation untersucht. Im Gegensatz zum Verstecken ist die Aktion *show* nur über ein Untermenü, in diesem Fall über das Menü *Show associations*, möglich. Das Menü wird über das Kontextmenü eingeblendet, wenn mindestens eine versteckte Assoziation vorhanden ist. Die bisher verwendete Abbildung *UML-Metamodell* enthält keine versteckten Assoziationen, sodass die Darstellung für die Evaluation angepasst wird. Auf das geladene UML-Metamodell wird die Aktion *Hide all associations* ausgeführt. Das Ergebnis wird in der Abbildung *UML-Metamodell ohne Assoziationen* dargestellt.

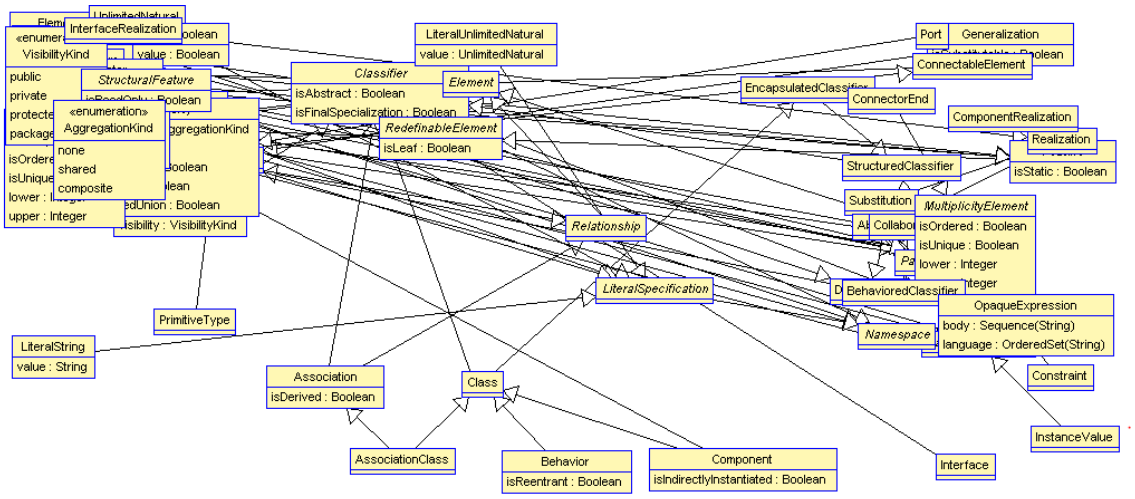


Abbildung 37: UML-Metamodell ohne Assoziationen

Die Assoziation *A_Element_AnnotatedElement_Comment_Comment* wurde für die Evaluation ausgewählt. Bei der genannten Beziehung handelt es sich um eine binäre Assoziation, die die Klassen *Element* und *Comment* miteinander verbindet. In der Abbildung *UML-Metamodell ohne Assoziationen* sind mehr als 20 Assoziationen ausgeblendet, sodass wieder Scrolling im Untermenü eingesetzt wird. Nachdem die gesuchte Interaktion gefunden und ausgeführt wurde, wird in die Darstellung die binäre Assoziation wieder angezeigt. Allerdings ist die Beziehung schwer erkennbar, da die Verbindungslinie größtenteils von der Klasse *Classifier* überdeckt wird. Erschwerend kommt hinzu, dass die Klasse *Comment* vollständig verdeckt ist und somit ein Endpunkt überdeckt wird. Das Ergebnis der Anwendung der Operation *show*, eine eingeblendete Kante, ist ebenfalls von Überdeckungen beeinflusst. Es liegen aber keine Einschränkungen beim Initiieren der Aktion vor.

Als letzte Basisoperation wird die Anwendung der Operation *crop* auf eine Generalisierung auf ihre Nutzbarkeit untersucht. Die Aktion wird nur im Kontextmenü angeboten, wenn zuvor ein Element selektiert wurde. Es wurde bereits gezeigt, dass die Selektion von Elementen auf Grund von Überdeckungen erschwert beziehungsweise nicht möglich ist. Das Ergebnis dieser Untersuchung ist daher abhängig von der Wahl der Generalisierung und ob diese identifizierbar ist. Die beiden Abbildungen *UML-Metamodell* und *UML-Metamodell ohne Assoziationen* enthalten einige Generalisierungen. Nachdem die Assoziationen ausgeblendet wurden, ist die Selektion vereinfacht. Überdeckungen sind jedoch weiterhin vorhanden. Bei einigen Generalisierungen ist zudem nicht erkennbar, an welchen Klassen die Beziehung endet. Die Identifikation einer selektierten Generalisierung ist auf Basis der Bezeichnung der Option im Kontextmenü möglich, solange nur ein einziges Element selektiert wurde. Werden mehrere Elemente selektiert, gibt die Beschriftung der Option nur noch Auskunft über die Anzahl der selektierten Elemente. Ohne eine eindeutige Identi-

fizierung auf welche Generalisierung die Operation *crop* angewandt wird, können ungewollte Darstellungen entstehen. Während die Operation *hide* und *show* sich gegenseitig aufheben, ist die Umkehrung eines *crop* nur durch die Ausführungen mehrerer Basisoperationen möglich. Daher ist eine ungewollte Ausführung mit einer versehentlich falsch selektierten Generalisierung zu vermeiden.

Zusätzlich zur Vereinheitlichung der Anwendung der Basisoperation auf Klassen, Assoziationen und Generalisierungen wurden zwei neue Benutzungsmöglichkeiten entwickelt und implementiert. Die beiden neuen Interaktionen ermöglichen die Anwendung der Basisoperationen auf Ober- und Unterklassen. Für die Nutzung dieser Optionen sind zwei Voraussetzungen zu erfüllen. Damit das Kontextmenü um die Optionen erweitert wird, muss eine Klasse selektiert werden und diese Klasse muss eine Generalisierung aufweisen.

Die Selektion einer Klasse stellt ein Problem dar, wenn die gesuchte Klasse von anderen Elementen überdeckt wird. Bei einer vollständigen Überdeckung ist eine Selektion nicht möglich. Die Aktionen wirken sich auf eine Teilmenge der vorhandenen Klassen aus. Im Gegensatz zu den bisher betrachteten Aktionen arbeiten die Optionen aus dem Menü *Sub/Superclasses* auf eine Menge von Klassen. Die neuen Funktionen ermöglichen mit einem Klick, und ohne vorherige mehrfache Selektion, eine Aktion auf eine Teilemenge von Klassen auszuführen. Je nach gewählter Option wird die Sichtbarkeit der Ober- und Unterklassen verändert, was bei Diagrammen mit vielen Elementen problematisch sein könnte. Wird beispielsweise die Klasse *Classifier*, die im oberen mittigen Teil der Abbildungen positioniert ist, selektiert, ist weder in der Abbildung *UML-Metamodell* noch in der Abbildung *UML-Metamodell ohne Assoziationen* erkennbar, über welche Ober- und Unterklassen die Klasse *Classifier* verfügt. Folglich ist das Ergebnis der Anpassung der Darstellung des Diagramms für den Anwender vorher nicht ersichtlich. Anders sieht es aus, wenn die Operationen auf Klassen angewendet werden, die nicht von einer Überdeckung betroffen sind. Im unteren mittigen Teil der Abbildungen ist die Klasse *Class* positioniert, die über Unterklassen verfügt. Eine Anwendung der Operationen auf deren Unterklassen und die damit verbundenen Anpassungen der Darstellung ist nachvollziehbarer als es bei der Klasse *Classifier* der Fall ist. Abhängig vom Szenario ist die Nutzbarkeit der Interaktionen durch die vorausgesetzte Selektion erschwert.

Während der Konzeption und Implementierung wurde angemerkt, dass die Zusammenlegung der Aktionen auf Ober- und Unterklassen in einem Menü unter Umständen unübersichtlich sein könnte. Bei der Nutzung des Untermenüs *Sub/Superclasses* im Zusammenhang mit dem UML-Metalmodell wurden keine Einschränkungen festgestellt, die gegen die verwendete Gestaltung des Menüs sprechen. Bei der Aus-

wahl einer Klasse, die über Ober- und Unterklassen verfügt, werden über das Menü maximal sechs Benutzungsmöglichkeiten angeboten. Die Anzahl kann je nach Sichtbarkeit der Klassen reduziert sein. Die maximale Anzahl von sechs Optionen ergibt sich aus den drei möglichen Basisoperationen und deren Anwendung auf die Ober- und Unterklassen. Bei der Selektion von mehreren Klassen, die über eine Generalisierung verfügen, ändert sich die maximale Anzahl der Einträge im Menü nicht. Es wird die Bezeichnung der Optionen angepasst. Beispielsweise wird die Bezeichnung *Hide subclasses of <Name einer Klasse>* durch *Hide subclasses of n elements* ersetzt. Der Platzhalter $n$ entspricht der Anzahl an ausgewählten Klassen, die über Unterklassen verfügen. Während der Evaluation konnten keine Argumente gefunden werden, die gegen die jetzige Gestaltung der Menüs *Sub/Superclasses* sprechen. Aus diesem Grund wird die aktuelle Gestaltung beibehalten.

Zusammengefasst ist die Nutzung der geschaffenen und angepassten Aktionen mit Einschränkungen möglich. Ob eine Aktion über die gewünschte Benutzungsoption verwendet werden kann, ist abhängig von der Positionierung der Elemente in der Darstellung und der damit verbundenen Wahrscheinlichkeit von Überdeckungen.

## 7.2. Ansicht Objektdiagramm

Auch in der Ansicht Objektdiagramm stand die Stärkung der drei Basisoperationen im Vordergrund, um eine einheitliche und vollständige Anwendung der Operationen auf die Konzepte des Objektdiagramms zu ermöglichen. Die Anwendung der Aktionen soll innerhalb, aber auch zu der Ansicht Klassendiagramm, vergleichbar sein. Die Vereinheitlichung und Vervollständigung verfolgen das Ziel, die Bedienbarkeit und Akzeptanz des Anwenders zu erhöhen. Das Ziel soll durch die Wiedererkennbarkeit von Strukturen und Verhaltensweisen erreicht werden. Zusätzlich wurde durch die Konzeption und Umsetzung von zwei neuen Benutzungsmöglichkeiten die vorhanden Auswahl- und Filteroptionen erweitert. Allerdings liegen noch keine Kenntnisse über die Nutzbarkeit der getätigten Änderungen vor, wenn das Objektdiagramm über eine Vielzahl von Elementen verfügt.

Das bisher verwendete Beispiel der *StudentWorld* genügt den Anforderungen für die Evaluation nicht, da es nicht ausreichend Elemente enthält. Für die Erzeugung eines geeigneten Szenarios wird die USE-Erweiterung *ModelValidator*[7] verwendet. Die Erweiterung generiert auf Basis eines Klassendiagramms ein Objektdiagramm. Der Prozess der Generierung kann mit Anforderungen versehen werden, die das Objektdiagramm erfüllen muss. Für die Evaluation werden Anforderungen an die

[7] `https://sourceforge.net/projects/useocl/files/Plugins/ModelValidator/` (zuletzt aufgerufen am 27.12.2018)

Anzahl der vorhandenen Instanzen gestellt.

Es wird eine *.properties*-Datei erstellt, die die Anforderungen an das Objektdiagramm enthält. Mit Hilfe der Datei können Unter- und Obergrenzen für die Anzahl einzelner Objekte und Links definiert werden. Des Weiteren kann der Wertebereich der Attribute beschränkt werden. In dem Beispiel *StudentWorld* verfügt nur die Assoziationsklasse *Tutorial* über ein Attribut, das für die Evaluation jedoch nicht von Bedeutung ist. Daher werden nur Einschränkungen in Bezug auf die Anzahl an Objekten und Links definiert. Es werden keine Beschränkungen vorgegeben, welche Objekte über einen Link miteinander in Beziehung stehen müssen. Folglich besteht die Möglichkeit, dass ein Student einmal einen Kurs besucht und in einer anderen generierten Variante den Kurs nicht belegt. Die *.properties*-Datei befindet sich im Anhang (vgl. [GHD17] S.9f.).

Zu beachten ist, dass der *ModelValidator* keine Instanzen von qualifizierenden Assoziationen unterstützt. Die *StudentWorld* verfügt über die Klassen *Server* und *Student*, die über die qualifizierende Assoziation *ServerStudent* verbunden sind. Da diese Art der Assoziation von dem *ModelValidator* nicht unterstützt wird, wird keine Instanz von *ServerStudent* angefordert beziehungsweise die Anzahl der Instanzen dieser Assoziation wird auf Null beschränkt. Alternativ könnte die Assoziation durch eine Assoziationsklasse ausgedrückt werden. Dadurch entsteht jedoch kein Mehrwert, da es mit *Tutorial* bereits eine Assoziation mit Assoziationsklasse gibt (vgl. [GHD17] S.26.).

Die Abbildung *Generierte StudentWorld* zeigt ein mögliches Objektdiagramm, das auf dem Klassendiagramm der *StudentWorld* basiert. Bei jeder Generierung auf Basis der *StudentWorld.properties* besteht die Möglichkeit, dass ein anderes Objektdiagramm entsteht. Für die Evaluation ist das nicht relevant, da die Existenz der Instanzen ausreichend ist. Jedes Diagramm enthält 64 Objekte und 100 Links. Die Darstellung entstand durch das Einlesen der Datei *StudentWorld.properties* über die USE-Erweiterung *ModelValidator* und anschließend wurde die *Auto-Layout*-Funktion angewendet, die die Positionierung der Instanzen verändert. In der ursprünglichen Darstellung waren nahezu alle Elementen übereinander positioniert.

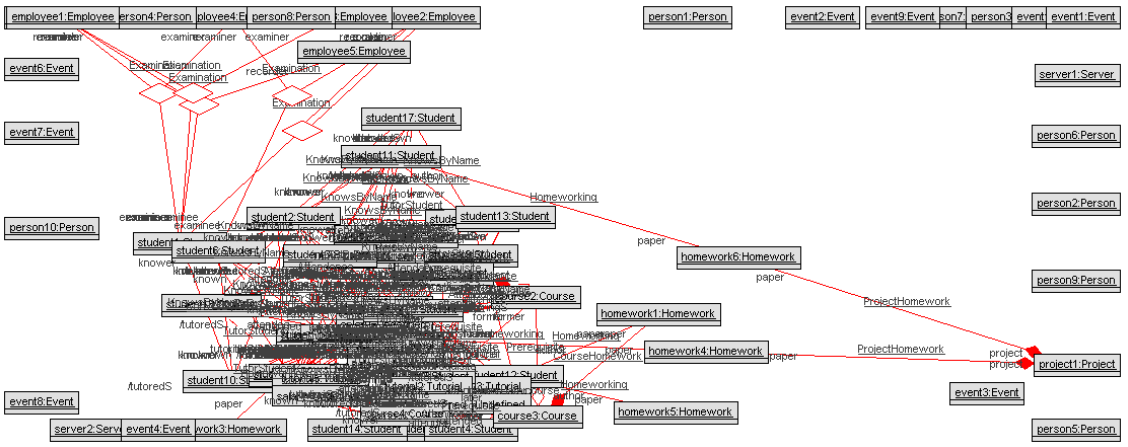


Abbildung 38: Generierte *StudentWorld*

In der Abbildung sind nicht alle 164 Instanzen klar und eindeutig erkennbar, da einige Instanzen teilweise oder vollständig von anderen Instanzen überdeckt werden. Diese Problematik wurde bereits im Abschnitt *Ansicht Klassendiagramm* betrachtet und das Ergebnis ist auf die Ansicht Objektdiagramm übertragbar. Die Selektion von einzelnen Elementen ist teilweise nicht möglich, sodass die Benutzungsmöglichkeiten nicht verwendet werden können. Die Basisoperationen *hide* und *show* können über die geschaffenen Untermenüs *Hide objects* und *Hide links* beziehungsweise *Show objects* und *Show links* initiiert werden. Da die vollständige und fehlerfreie Implementierung bereits im Kapitel *Implementierung und Anpassung der Benutzungsmöglichkeiten* gezeigt wurde und keine zusätzlichen Einschränkungen im Vergleich zu der Ansicht Klassendiagramm vorhanden sind, wird auf eine weitere Untersuchung dieser Benutzungsmöglichkeiten verzichtet.

Die neu geschaffenen Interaktionen, die ein Ausgrauen von Objekten und die Anwendung der Basisoperationen *hide* und *show* auf Assoziationsarten ermöglichen, wird nun in Bezug auf deren Nutzbarkeit untersucht. Ähnlich wie bei der Basisoperation *crop* ist eine Selektion eines Objektes notwendig, um die Aktion *Grey out* oder *Grey in* anzuwenden. In der generierten Darstellung ist es nicht möglich alle Objekte gezielt zu selektieren und die Aktion zu nutzen. Auch wenn die Aktion auf ein überdecktes Objekt anwendbar wäre, entsteht kein Mehrwert. Das Objekt wäre ausgegraut in der Darstellung vorhanden, wird aber weiterhin von anderen Elementen überdeckt. Die Veränderung des Kontrastes ist nicht sichtbar und generiert keinen Effekt.

Die Funktion des Ausgrauens wurde konzeptioniert, um den Fokus des Betrachters auf nicht ausgegraute Objekte zu richten. Dazu wurde der Kontrast zwischen der Darstellung eines Objekts und dem weißen Hintergrund verringert. Die Nutzung dieser Funktionalität kann bei großen Diagrammen jedoch das Gegenteil bewirken.

Die Abbildung *Fokussierung durchs Ausgrauen* zeigt eine solche Konstellation. In der Darstellung wurden zwei Objekte ausgegraut. Das Objekt *event5* ist am linken Rand der Darstellung positioniert und vollständig sichtbar, während das Objekt *student7* sich in der Mitte befindet und teilweise durch andere Elemente verdeckt wird. Die beiden Objekte heben sich in diesem Fall nicht stark von dem Hintergrund ab, allerdings von allen anderen Elementen. Diese ungewollte Abhebung der beiden Objekte kann je nach Betrachter zu einer ungewollten Fokussierung auf die beiden ausgegrauten Elemente führen. Statt einer Schwächung der Fokussierung rücken die Elemente in den Fokus. Eine mögliche Umkehrung könnte erreicht werden, wenn deutlich mehr Objekte ausgegraut werden. Sind mehr ausgegraute Objekte oder allgemein weniger Elemente in der Darstellung vorhanden, verschwimmen die ausgegrauten Objekte besser mit dem Hintergrund und es kommt nicht zu einer Abhebung. Um eine große Anzahl an Objekten auszugrauen, ist die Selektion von entsprechend vielen einzelnen Elementen notwendig, da keine Option existiert, die die Anwendung dieser Operation auf eine Teilmenge ermöglicht. Dieses Vorgehen ist jedoch mühsam und nicht sehr benutzerfreundlich, sodass der eigentliche Zweck dieser Funktion zumindest bei Darstellungen mit vielen Elementen nicht erfüllt wird.

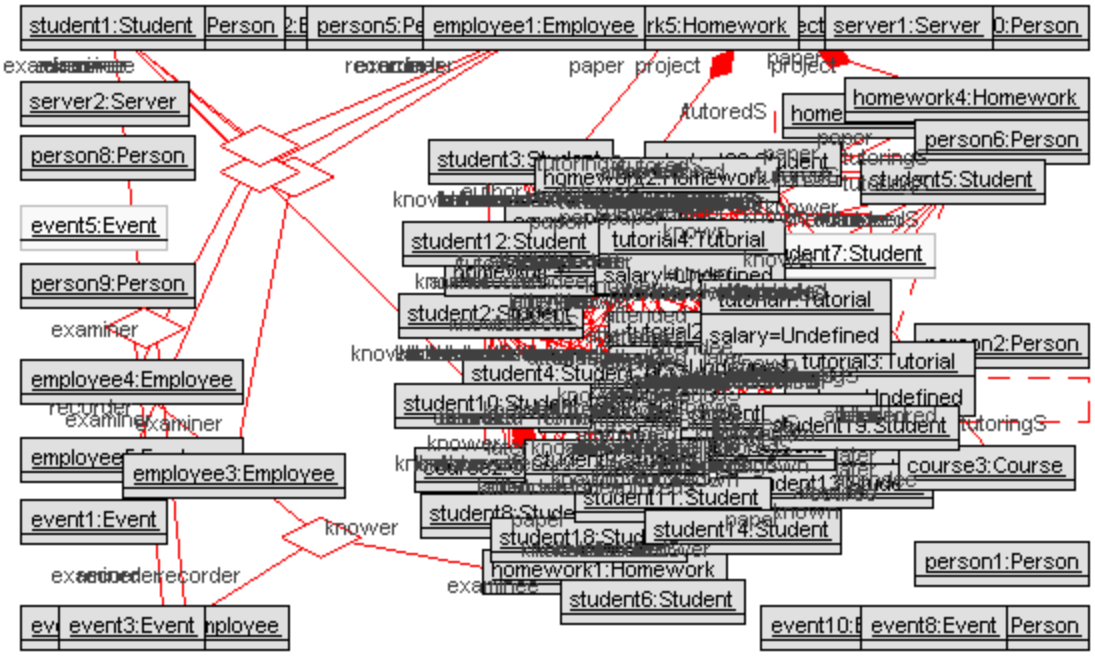


Abbildung 39: Fokussierung durchs Ausgrauen

Während die Untermenüs *Hide objects*, *Hide links*, *Show objects* und *Show links* eine Gruppierung nach Klassen oder Assoziationen durchführen, werden bei dem Untermenü *Hide/show links-by-kind* Links nach deren Assoziationsart gruppiert. Das Menü bietet die Operationen *hide* und *show* auf alle Links, alle Links einer Assoziationsart oder auf genau einem Link an. In der ersten Ebene des Menüs werden die

sieben Assoziationsarten gelistet, wenn mindestens eine Instanz der Assoziationsart vorhanden ist. Bei der Prüfung der Funktionalität dieser Benutzungsmöglichkeit in der Implementierungsphase wurden die Instanzen in zwei Kategorien gruppiert, da nicht für jede Assoziationsart eine Instanz erstellt wurde. Zur Sicherstellung, dass auch alle anderen Kategorien korrekt befüllt werden, wird die Gruppierung auf Basis der generierten *StudentWorld* erneut untersucht und die Ergebnisse in der Tabelle *Evaluation der Gruppierungen bei links-by-kind* festgehalten.

| **Testfall** | **Ergebnis** |
|---|---|
| Das Menü muss über sieben Assoziationsarten verfügen. | ✓ |
| Die Gruppierung *Aggregations* muss über zehn Einträge verfügen. | ✓ |
| Die Gruppierung *Binary links* muss über 31 Einträge verfügen. | ✓ |
| Die Gruppierung *Compositions* muss über vier Einträge verfügen. | ✓ |
| Die Anzahl der Links in der Gruppierung *Derrived links* variiert je nach generiertem Zustand. Eine Überprüfung ist nicht möglich. | entfällt |
| Die Gruppierung *Linkobjects* muss über fünf Einträge verfügen. | ✓ |
| Die Gruppierung *N-Ary links* muss über fünf Einträge verfügen. | ✓ |
| Die Gruppierung *Reflexiv links* muss über 40 Einträge verfügen. | ✓ |

Tabelle 13: Evaluation der Gruppierungen bei links-by-kind

Das neue Menü schafft eine gute Mischung aus Aktionen, die auf alle Links, eine Teilmenge von Links oder genau einem Link angewendet werden können. Das Ausführung von Aktionen auf einzelne Links oder auf alle Links ist auch über andere Optionen möglich. Neu hinzugekommen ist die Auflistung auf Basis der Gruppierung von Assoziationsarten und die Anwendung von Aktionen auf alle Links einer Assoziationsart. Durch das Verstecken oder Einblenden kann die Darstellung um eine Teilmenge von Elementen reduziert oder erweitert werden, ohne das eine vorherige Selektion erforderlich ist. Die Problematik der Selektion ist nicht gegeben. Die neuen Benutzungsmöglichkeiten verwenden zwei Basisoperationen, die Bestimmung der Teilmenge ist jedoch nicht ohne eine Reihung von einzelnen Aktionen durch andere Benutzungsmöglichkeiten realisierbar. Das Verstecken oder Einblenden von Instanzen einer Assoziationsart ermöglicht neue Filter- und Auswahlmöglichkeiten, die den Schwerpunkt auf Assoziationsarten legen und die Anzahl der Elemente in der Darstellung schnell verändern können.

Alle neu geschaffenen Funktionen werden durch eine Erweiterung des Kontextmenüs realisiert. Folglich können alle Optionen aus einem Menü ausgeführt werden, das leicht verständlich über einen Rechtsklick geöffnet werden kann. Es existiert ein zentraler Einstiegspunkt für alle in dieser Ausarbeitung relevanten Benutzungsmöglichkeiten. Im Kontext dieser Arbeit wurde durch die Stärkung des Konzepts der Basisoperationen und durch die Implementierung neuer Interaktionen neue Einträge in dem Kontextmenü geschaffen. Je nach Darstellung verändern sich die möglichen

Optionen, die über das Menü angeboten werden. Bei einem Objektdiagramm, dessen Darstellung folgende Merkmale aufweist, werden 28 Menüeinträge angezeigt.

- Mindestens ein sichtbares Objekt
- Mindestens ein verstecktes Objekt
- Mindestens einen sichtbaren Link
- Mindestens einen versteckten Link
- Ein selektiertes Objekt, mit dem ein reflexiver Link gebildet werden kann

Bei einer Auflösung von 1366 x 768 Pixeln nimmt die Länge des Kontextmenüs ungefähr 90% der Höhe des Bildschirms ein. Die Vielzahl der angebotenen Optionen kann bei einem neuen Anwender zu einer Überforderungen führen. Es ist zu überlegen, ob die Fülle an möglichen Interaktionen reduziert werden kann. Beispielsweise könnten die *Layout*-Funktionen (*Auto-Layout, Reset layout, Load layout, Save layout*) über ein Untermenü zusammengefasst werden. Ähnlich könnte mit den Optionen verfahren werden, die zusätzliche Merkmale ein- oder ausblenden. Gemeint sind die Optionen *Show states*, *Show attributes*, *Show association names* und *Show role names*.

Die Ergebnisse dieses Abschnitts werden im nachfolgenden Abschnitt mit den Erkenntnissen aus dem Abschnitt *Ansicht Klassendiagramm* in Zusammenhang gebracht. Zudem wird ein abschließendes Fazit zu der Verwendbarkeit der geschaffenen Benutzungsmöglichkeiten gezogen.

## 7.3. Ergebnisse der Evaluation

Zu Beginn dieses Kapitels wurde die Frage gestellt, ob die neu geschaffenen und angepassten Benutzungsmöglichkeiten geeignete Auswahl- und Filteroptionen sind, wenn die Darstellung eine Vielzahl von Elementen enthält. Die Ansichten Klassen- und Objektdiagramm wurden separat untersucht und die Teilergebnisse werden aufgegriffen und zu einem Gesamtergebnis zusammengefasst.

Bei der Evaluation wurden Darstellungen verwendet, die eine Vielzahl von Elementen im Vergleich zu den Beispielen aus den Kapiteln *Analyse der USE-Version*, *Konzeption neuer Benutzungsmöglichkeiten* und *Implementierung und Anpassung der Benutzungsmöglichkeiten* aufweisen. Die erhöhte Anzahl der Elemente resultierte in Problematiken, die in den vorherigen Kapiteln nicht in dieser Ausprägung auftraten. Die Teilevaluationen belegen aber auch die positiven Effekte, die durch

die angepasste Sortierung und die Einführung des Scrollings erzeugt werden konnten. Im Verlauf dieses Abschnitts wird primär auf die Problematiken eingegangen, die die Verwendung der Benutzungsmöglichkeiten erschweren.

In beiden Teilevaluationen ergaben sich Schwierigkeiten, wenn eine Interaktion eine Selektion eines Elements voraussetzt. Damit eine Selektion durchgeführt werden kann, muss die Position des zu selektierenden Elements bekannt sein und es darf nicht vollständig durch andere Elemente verdeckt werden. Die Bestimmung der Position kann durch eine teilweise Überdeckung erschwert oder durch eine vollständige Überdeckung verhindert werden. Sollte ein gesuchtes Element keine Überdeckungen aufweisen, wird durch die Vielzahl anderer Elemente das Auffinden der Position einige Zeit in Anspruch nehmen. In der Ansicht Klassendiagramm existiert mit den Enumerationen ein Konzept, dessen Sichtbarkeit nicht von den drei Basisoperationen beeinflusst wird. Eine Überdeckung durch eine Enumeration kann nur durch das Verschieben der Enumeration aufgelöst werden. Die grundlegende Problematik bei der Selektion von Elementen kann auf die Überdeckung und die damit verbundene Positionierung von Elementen zurückgeführt werden.

Die Nutzbarkeit der Interaktionen, die eine Selektion voraussetzen, ist bei Darstellungen mit Überdeckungen nur eingeschränkt gegeben. Der Nutzen dieser Interaktionen entfaltet sich erst vollständig, wenn die Positionsbestimmung schnell erfolgen kann. So zeigten die vorherigen Kapitel, dass die Anwendung der Basisoperationen mit vorheriger Selektion eine schnelle und komfortable Möglichkeit ist, um eine Darstellung anzupassen. Es wird die Auswahl mehrerer Elemente unterstützt und die beiden Basisoperationen *hide* und *crop* werden schnell auffindbar im oberen Teil des Kontextmenüs als Optionen angeboten.

Im Kontext dieser Ausarbeitung wurden neue Untermenüs geschaffen und vorhandene angepasst. Diese Menüs bieten neben den Basisoperationen *hide* und *crop*, die auch über eine Selektion initiiert werden könnten, auch die Operation *show* an. Die Möglichkeit Aktionen über eine Selektion oder ein Untermenü zu initiieren ergänzen sich und kompensieren ihre Nachteile. Es wurde dargelegt, dass die Selektion bei Überdeckungen teilweise nicht verwendet werden kann, aber ihre Vorteile bei Darstellungen ohne Überdeckungen hat. Die Untermenüs ermöglichen die Anwendung der Operation *hide* und *show* auch bei Überdeckungen. Die Auswahl, auf welches Element eine Aktion ausgeführt werden soll, erfolgt über eine Auflistung im Untermenü. Die Position des Elements in der Darstellungen ist dabei nicht relevant. Folglich ist eine Ausführung von Aktionen auf verdeckte Elemente möglich, auch wenn die Position nicht bestimmt werden kann. Sobald die Anzahl der sichtbaren Elemente abnimmt, reduziert sich auch die Eintrittswahrscheinlichkeit von Überde-

ckungen.

Alle Benutzungsmöglichkeiten werden in das Kontextmenü eingebunden. Je nach vorhandener Darstellung wird dem Benutzer eine Vielzahl unterschiedlicher Optionen und Untermenüs angeboten. Diese Auflistung möglicher Interaktionen wird durch die Länge der Liste unübersichtlich. Während der Evaluation der Ansicht Objektdiagramm wurden 28 Einträge im Kontextmenü angezeigt. Je nach verwendeter Auflösung nimmt die Liste einen großen Teil der Bildschirmhöhe ein und könnte einen möglichen Anwender durch die reine Anzahl der Optionen überfordern. Außerhalb dieser Ausarbeitung könnte eine Überarbeitung der vorhandenen Struktur des Kontextmenüs geprüft werden.

Des Weiteren wurde gezeigt, dass die Funktionalität des Ausgrauens von Objekten auch die entgegengesetzte Wirkung erzeugen kann. Je nach Anzahl der Elemente kann ein ausgegrautes Objekt auch fokussierend wirken. Sind nur wenige Objekte ausgegraut, heben sich diese Objekte von den anderen Objekten ab. Die Verschmelzung mit dem Hintergrund erfolgt nur teilweise, wenn hinter dem Objekt noch weitere Elemente sichtbar sind, also eine Überdeckung vorliegt. Der Hintergrund ist dann nicht weiß, sondern eher dunkel und der gewünschte Effekt durch die veränderte Darstellung tritt nicht ein. Aus diesem Grund wird die Funktionalität vorerst nicht auf andere Elemente erweitert und bleibt ein exklusives Feature für Objekte.

Zusammengefasst wurde gezeigt, dass sowohl die Ausführung der Basisoperationen über eine Selektion als auch über die Untermenüs benötigt werden. Die Selektion verfügt über Stärken, wenn eine Darstellung keine Überdeckungen aufweist und somit die Auswahl eines Elements möglich ist. Mit steigender Anzahl von Elementen in der Darstellung steigt auch die Wahrscheinlichkeit, dass es in der Darstellung zu Überdeckungen kommt. Sobald Überdeckungen vorliegen, ist die Selektion erschwert. Die Untermenüs ermöglichen die Initiierung von Aktionen auch auf verdeckte Elemente, unterstützen jedoch keine gezielte mehrfache Auswahl von Elementen. Die Selektion und die Untermenüs ergänzen sich und bieten je nach Situation einem Anwender die Möglichkeit die Darstellung zu verändern. Für eine sinnvolle Ergänzung ist die vollständige Implementierung der Basisoperationen notwendig, da ansonsten die benötigten Interaktionen nicht zur Verfügung stehen würden. Die beiden Beispiele in diesem Kapitel unterstreichen nochmals die Anforderung die Basisoperationen vollständig zu implementieren. Über die Untermenüs können Operationen initiiert werden, wenn keine Selektion von Elementen möglich ist. Es existieren auch bei vorhandenen Überdeckungen Möglichkeiten die Anzahl der sichtbaren Elemente zu reduzieren. Die Nutzbarkeit einiger Funktionalitäten ist von der Anzahl und Positionierung der Elemente abhängig. Insgesamt ergänzen die neu geschaffenen Be-

nutzungsmöglichkeiten die vorhandenen Interaktionen, obwohl einige Aktionen von der Problematik der Überdeckungen betroffen sind.

# 8. Zusammenfassung und Fazit

In vielen Gebieten der Forschung werden Modelle erstellt und verwendet. Einige Bereiche verwenden Modelle als Grundlage für die Vorhersage von zukünftigen Entwicklungen. Dabei werden beispielsweise Prognosen über den demografischen Wandel der Gesellschaft oder über den Anstieg der Meeresspiegel erzeugt. Auf diese Prognosen haben eine Vielzahl von Faktoren einen Einfluss, deren Bedeutungen und Auswirkungen modelliert sein müssen. In der Softwareentwicklung werden ebenfalls Modelle erstellt, um die Anforderungen an eine Software in Form einer strukturellen Beschreibung zu erfassen. Ähnlich wie bei den Prognosen werden Objekte und deren Verhaltensweisen erfasst und beschrieben. Je komplexer ein zu modellierendes Vorhaben ist, desto mehr Informationen werden erfasst. Das entstehende Modell wird detaillierter und ausgereifter. Die Anreicherung eines Modells mit Informationen kann in einer unübersichtlichen Darstellung resultieren. Soll ein Teilaspekt betrachtet oder fokussiert werden, ist die Reduzierung der Darstellung auf die wesentlichen Elemente erforderlich. Diese Reduzierung kann durch geeignete Auswahl- und Filteroptionen erfolgen.

Das Tool *USE - A UML-based specification environment* kann Modelle in Form von verschiedenen Diagrammen darstellen. In dieser Ausarbeitung wurde untersucht, welche Benutzungsmöglichkeiten das Tool für die Veränderung der Darstellung von Klassen- und Objektdiagrammen bietet. Dazu wurden die vorhanden Interaktionen der beiden Diagrammarten untersucht. Das Ergebnis der Analyse ist, dass die Aktionen auf ein übergeordnetes Schema abgebildet werden können, welches jedoch nicht vollständig und fehlerfrei umgesetzt worden war. Es existieren drei Basisoperationen, die für eine Veränderung der Darstellung im Sinne einer Auswahl- und Filterfunktionalität verwendet werden können. Die gefundenen Schwachstellen in den Benutzungsmöglichkeiten wurden korrigiert und die Verfügbarkeit der drei Basisoperationen wurde auf weitere Konzepte der Diagrammarten erweitert, um eine einheitliche und vollständige Umsetzung zu erreichen. Die einheitliche und vollständige Implementierung resultiert in der Wiederverwendung von Strukturen und Abläufen, die die Benutzung der einzelnen Optionen vereinfachen und durch die erhöhte Wiedererkennbarkeit verständlicher werden lassen.

Die Stärkung der Verwendung der Basisoperationen wird durch neue Benutzungsmöglichkeiten ergänzt. Die neuen Aktionen erweitern die Möglichkeiten zur Anpassung einer vorhandenen Darstellung. Bei einigen neuen Interaktionen werden wieder Basisoperationen eingesetzt. Die Operationen wirken dabei auf eine Menge von Elementen, die nicht durch andere Interaktionen realisierbar sind. Es liegt eine Er-

weiterung der Anwendungsmöglichkeiten vor.

Der Bedarf an Auswahl- und Filteroptionen wird vor allem bei Darstellungen mit vielen Elementen ersichtlich, wenn diese Darstellungen auf einen Teilaspekt reduziert werden sollen. Für die Betrachtung eines Teilaspekts sind oftmals nicht alle Elemente erforderlich und können aus Gründen der Übersichtlichkeit und des Verständnisses ausgeblendet werden. Eine Darstellung mit wenigen Elementen ist für Dritte schneller zu erfassen und zu verstehen. Bei der Anwendung der geschaffenen und vorhandenen Optionen in Verbindung mit vielen Elementen traten jedoch Schwierigkeiten auf. Die Nutzbarkeit einiger Benutzungsmöglichkeiten wurde durch teilweise oder vollständige Überdeckungen erschwert beziehungsweise verhindert. Bei einer großen Anzahl an Elementen kann es zu Überdeckungen kommen, wenn nicht ausreichend Platz für eine überschneidungsfreie Platzierung vorhanden ist. Oftmals überschneiden sich Beziehungen, da sie verschiedene Elemente miteinander verbinden und eine kollisionsfreie Positionierung nicht möglich ist. USE platziert Klassen, beziehungsweise Objekte, gehäuft in einem Teil der Darstellung und verwendet andere Teile der Darstellungsfläche weniger intensiv. Das hat zur Folge, dass nicht nur Überscheidungen von Beziehungen auftreten, sondern auch Klassen und Objekte sich gehäuft überdecken. Die Überdeckung dieser Elemente betrifft wieder die Beziehungen, wenn diese Elemente an einer Beziehung beteiligt sind. Es entsteht eine Art Cluster von Elementen und einzelne Elemente können nicht selektiert werden. Ohne die Möglichkeit einer Selektion sind einige Auswahl- und Filteroptionen nicht verwendbar. Die Problematik der Überdeckungen könnte durch die Einführung von Positionsregeln und der damit verbundenen veränderten Positionierung von Elementen reduziert werden. Würde USE Klassen und Objekte nicht gehäuft in einem Teil der Darstellungsfläche platzieren, sondern den vorhandenen Platz effizienter nutzen, hätte dieses Verhalten positive Auswirkungen auf die Nutzbarkeit der Interaktionen, die eine Selektion voraussetzen. Eine bessere Nutzbarkeit dieser Aktionen resultiert zudem in einer effizienteren Anwendung der Auswahl- und Filteroptionen und einer erleichterten Anpassung von Darstellungen.

Die eingeführte alphanumerischen Sortierung und die scrollbaren Menüs entfalten erst ab einer bestimmten Anzahl von Elementen ihren Nutzen. Diese Anpassungen führen bei der Handhabung von Benutzungsmöglichkeiten bei Darstellungen mit vielen Elementen zu Erleichterungen und einer übersichtlichen Anordnung der angebotenen Interaktionen. Neben diesen Änderungen sind auch die Aktionen auf Ober- und Unterklassen positiv zu erwähnen. Diese Funktionen kombinieren die Basisoperationen mit der Eigenschaft einer Generalisierung, Klassen in Ober- beziehungsweise Unterklassen zu unterteilen. Diese Interaktionen wurden nicht nur im Kontext dieser Ausarbeitung evaluiert, sondern kurz nach Fertigstellung der Imple-

mentierung einem Mitglied der Arbeitsgruppe Datenbanksysteme der Universität Bremen zur Verfügung gestellt, der die neu geschaffenen Funktionalitäten im Zuge einer weiteren Ausarbeitung im Umfeld von USE verwendet.

Der Fokus dieser Ausarbeitung lag auf der Untersuchung der Fragestellung, ob die in USE implementierten Auswahl- und Filteroptionen für die Anpassung von Darstellungen eines Klassen- oder Objektdiagramms geeignet sind. Die Optionen sollen es einem Anwender ermöglichen, eine vorliegende Darstellung schnell, effektiv und fehlerfrei an seine Wünsche anpassen zu können. Neben den beiden Auswahl- und Filteroptionen *hide* und *crop*, welche die Anzahl der sichtbaren Elemente reduzieren, wurde auch die Operation *show* in die Betrachtung miteinbezogen. Die Operation *show* kann die beiden anderen Operationen umkehren und ebenfalls die Sichtbarkeit von Elementen verändern. Diese drei Operationen bilden die Grundlage für viele Interaktionen in den beiden Diagrammarten. Die Untersuchung zeigte jedoch, dass die Umsetzung der drei Operationen nicht einheitlich, nicht vollständig und nicht fehlerfrei erfolgte.

Die wichtigste Funktion von Auswahl- und Filteroptionen ist die Veränderung der Anzahl von sichtbaren Elementen in einer Darstellung. Benötigt werden diese Funktionen vor allem bei Darstellungen mit vielen Elementen. In der Evaluation wurde gezeigt, dass die geschaffene vollständige und fehlerfreie Umsetzung der Basisoperationen für die Anpassung von Darstellungen essenziell ist. Die in der Evaluation verwendeten Darstellungen zeigten Problematiken bei der Verwendung von Auswahl- und Filteraktionen auf, die nur durch die durchgeführte Stärkung der Anwendung der Basisoperationen abgefangen werden konnten. Wie effizient und nutzbar die geschaffenen Optionen sind, hängt vom Einzelfall und den damit verbundenen Gegebenheiten ab. Nicht nur die Anzahl der Elemente und deren Beziehungen haben starke Auswirkungen auf die Nutzbarkeit der Interaktionen, sondern auch die Positionierung der Elemente in der Darstellung. Treten teilweise oder vollständige Überdeckungen auf, verschlechtert sich die Nutzbarkeit einiger Optionen.

Die neu geschaffenen Möglichkeiten bieten neue Arten, eine Darstellung anzupassen. Diese neuen Funktionen greifen dabei zum Teil auf die Selektion und Basisoperationen zurück, kombinieren deren Eigenschaften neu und schaffen dadurch neue Anwendungsszenarien. Die Gesamtheit der umgesetzten Maßnahmen führt zu einem mächtigen Set an Auswahl- und Filteroptionen, die auch Optionen zur Anpassung von Darstellungen mit vielen Elementen enthalten. Im Vergleich zu der USE-Version 5.0.1 wurde der Einsatz von Auswahl- und Filteroptionen in den Bereichen Nutzbarkeit, Einheitlichkeit, Vollständigkeit und Korrektheit verbessert.

# Referenzen

[BG11] Büttner, Fabian ; Gogolla, Martin: Modular Embedding of the Object Constraint Language into a Programming Language. In: *Formal Methods, Foundations and Applications*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2011. – ISBN 978–3–642–25032–3, S. 124–139

[DGH17] Desai, Nisha ; Gogolla, Martin ; Hilken, Frank: Executing Models by Filmstripping: Enhancing Validation by Filmstrip Templates and Transformation Alternatives. In: *Proceedings of MODELS 2017 Satellite Event: Workshops (ModComp, ME, EXE, COMMitMDE, MRT, MULTI, GEMOC, MoDeVVa, MDETools, FlexMDE, MDEbug), Posters, Doctoral Symposium, Educator Symposium, ACM Student Research Competition, and Tools and Demonstrations co-located with ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS 2017), Austin, TX, USA, September, 17, 2017.*, 2017, 88–94

[Fow04] Fowler, M.: *UML konzentriert: eine kompakte Einführung in die Standard-Objektmodellierungssprache ; [zu UML 2.0]*. Addison-Wesley, 2004 (Programmer's choice). `https://books.google.de/books?id=DRUCJ_Vrn9IC`. – ISBN 9783827321268

[GBR07] Gogolla, Martin ; Büttner, Fabian ; Richters, Mark: USE: A UML-based specification environment for validating UML and OCL. In: *Science of Computer Programming* 69 (2007), Nr. 1, 27 - 34. `http://dx.doi.org/https://doi.org/10.1016/j.scico.2007.01.013`. – DOI https://doi.org/10.1016/j.scico.2007.01.013. – ISSN 0167–6423. – Special issue on Experimental Software and Toolkits

[GD18] Gogolla, Martin ; Doan, Khanh-Hoang: Visualizing and Analyzing Discrete Sets with a UML and OCL Software Design Tool. In: Sato, Yuri (Hrsg.) ; Shams, Zohreh (Hrsg.): *Proc. 6th Workshop Set Visualization and Reasoning (SetVR 2018)*, CEUR Proceedings 2116, 2018, S. 76–83

[GHD17] Gogolla, Martin ; Hilken, Frank ; Doan, Khanh-Hoang: Achieving Model Quality through Model Validation, Verification and Exploration. In: *Journal on Computer Languages, Systems and Structures, Elsevier, NL* (2017)

[GHH+14] Gogolla, Martin ; Hamann, Lars ; Hilken, Frank ; Kuhlmann, Mirco ; France, Robert B.: From Application Models to Filmstrip Models: An Approach to Automatic Validation of Model Dynamics. In: *Modellierung 2014, 19.-21. März 2014, Wien, Österreich*, 2014, 273–288

[GHHS14] Gogolla, Martin ; Hamann, Lars ; Hilken, Frank ; Sedlmeier, Matthias: Modeling Behavior with Interaction Diagrams in a UML and OCL Tool. In: *Behavior Modeling - Foundations and Applications, International Workshops, BM-FA 2009-2014, Revised Selected Papers*, 2014, 31–58

[Gog18] Gogolla, Martin: Squeezer Graphs: Proposing a Simple Basis for Computing (Meta-)Model Features. In: *Proc. STAF 2018 Workshops, Workshop Graph Computation Models (GCM 2018)*, 2018

[HHG14] Hilken, Frank ; Hamann, Lars ; Gogolla, Martin: Transformation of UML and OCL Models into Filmstrip Models. In: *Theory and Practice of Model Transformations - 7th International Conference, ICMT 2014, Held as Part of STAF 2014, York, UK, July 21-22, 2014. Proceedings*, 2014, 170–185

[KB18] Kastens, U. ; Büning, H.K.: *Modellierung: Grundlagen und formale Methoden*. Carl Hanser Verlag GmbH & Company KG, 2018. – ISBN 9783446454644

[Koc15] Koch, Christian: *Objektorientierte Modellierung*. `https://doi.org/10.1007/978-3-658-05606-3_3`. Version: 2015

[KS15] Kecher, C. ; Salvanos, A.: *UML 2.5: das umfassende Handbuch*. Rheinwerk Verlag, 2015 (Galileo Computing). – ISBN 9783836229777

[OMG17] OMG: *OMG Unified Modeling Language (OMG UML), Version 2.5.1*. `http://www.omg.org/spec/UML/2.5.1`. Version: 2017

[Ric02] Richters, Mark: *A Precise Approach to Validating UML Models and OCL Constraints*, Universität Bremen, Diss., 2002

[RJB04] Rumbaugh, James ; Jacobson, Ivar ; Booch, Grady: *Unified Modeling Language Reference Manual, The (2Nd Edition)*. Pearson Higher Education, 2004. – ISBN 0321245628

[RQZ12] Rupp, Chris ; Queins, Stefan ; Zengler, Barbara: *UML 2 glasklar - Praxiswissen für die UML-Modellierung*. 2012

[Rum13] Rumpe, B.: *Modellierung mit UML: Sprache, Konzepte und Methodik*. Springer Berlin Heidelberg, 2013 (Xpert.press). `https://books.google.de/books?id=ajLyBQAAQBAJ`. – ISBN 9783642187339

[Sch17] Schlobohm: *Erweiterung und Anpassung von UML-Interaktionsdiagrammen im Tool USE für eine übergreifende Repräsentation und verbesserte Einstellungsmöglichkeiten*. Dec 2017

[SG06] Seemann, Jochen ; Gudenberg, Jürgen W.: *Software Entwurf mit UML2*. 2. Springer, 2006 (Xpert.press). – ISBN 3–540–30949–7

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Listingverzeichnis

# A. USE-Spezifikation StudentWorld

```
1  model StudentWorld
2
3  class Person
4  end
5
6  class Event
7  end
8
9  class Student < Person
10 end
11
12 class Course < Event
13 end
14
15 association Attendance between
16   Student[*] role attendee
17   Course[*] role attended
18 end
19
20 associationclass Tutorial between
21   Student[*] role tutor
22   Course[*] role tutored
23 attributes
24   salary:Integer
25 end
26
27 association KnowsByName between
28   Student[*] role knower
29   Student[*] role known
30 end
31
32 class Employee < Person
33 end
34
35 association Examination between
36   Student[*] role examinee
37   Employee[*] role examiner
38   Employee[*] role recorder
39 end
40
41 aggregation Prerequisite between
42   Course[*] role later
43   Course[*] role former
44 end
45
46 class Homework
```

```
47 end
48
49 association Homeworking between
50   Student[1] role author
51   Homework[*] role paper
52 end
53
54 composition CourseHomework between
55   Course[0..1] role course
56   Homework[*] role paper
57 end
58
59 class Project < Event
60 end
61
62 composition ProjectHomework between
63   Project[0..1] role project
64   Homework[*] role paper
65 end
66
67 class Server
68 end
69
70 association ServerStudent between
71   Server[*] role server qualifier(login:String)
72   Student[0..1] role student
73 end
74
75 association TutorStudent between
76   Student[*] role tutoringS
77   Student[*] role tutoredS derived = self.tutored.attendee->asSet()
78 end
```

# B. USE-Spezifikation MiniStudentWorld

```
model StudentWorld

class Person
end

class Student < Person
end

class Course
end

association Attendance between
  Student[*] role attendee
  Course[*] role attended
end

associationclass Tutorial between
  Student[*] role tutor
  Course[*] role tutored
attributes
  salary:Integer
end

aggregation Prerequisite between
  Course[*] role later
  Course[*] role former
end
```

# C. Instanziierung der MiniStudentWorld

```
1 !new Student('ada'); new Student('dan')
2 !new Course('PCS1'); new Course('TCS1')
3 !insert (dan,TCS1) into Attendance
4 !insert (ada,TCS1) into Attendance
5 !ada_PCS1 := new Tutorial('ada_PCS1') between (ada,PCS1)
6 !ada_PCS1.salary := 400
7 !new Course('TCS2')
8 !insert (TCS2,TCS1) into Prerequisite
```

# D. StudentWorld.properties

```
[default]

# Person
Person_min = 10
Person_max = 10

# Event
Event_min = 10
Event_max = 10

# Student
Student_min = 20
Student_max = 20

# Course
Course_min = 5
Course_max = 5

# Employee
Employee_min = 5
Employee_max = 5

# Homework
Homework_min = 6
Homework_max = 6

# Project
Project_min = 1
Project_max = 1

# Server
Server_min = 2
Server_max = 2

# Tutorial (tutor:Student, tutored:Course)
Tutorial_min = 5
Tutorial_max = 5

# Attendance (attendee:Student, attended:Course)
Attendance_min = 25
Attendance_max = 25

# KnowsByName (knower:Student, known:Student)
KnowsByName_min = 40
KnowsByName_max = 40

```

```
47 # Examination (examinee:Student, examiner:Employee, recorder:Employee)
48 Examination_min = 5
49 Examination_max = 5
50
51 # Homeworking (author:Student, paper:Homework)
52 Homeworking_min = 6
53 Homeworking_max = 6
54
55 # TutorStudent (tutoringS:Student, tutoredS:Student)
56 TutorStudent_min = 5
57 TutorStudent_max = 5
58
59 # Prerequisite (later:Course, former:Course)
60 Prerequisite_min = 10
61 Prerequisite_max = 10
62
63 # CourseHomework (course:Course, paper:Homework)
64 CourseHomework_min = 2
65 CourseHomework_max = 2
66
67
68 # ProjectHomework (project:Project, paper:Homework)
69 ProjectHomework_min = 2
70 ProjectHomework_max = 2
71
72 # ServerStudent (server:Server, student:Student)
73 ServerStudent_min = 0
74 ServerStudent_max = 0
75
76 # Tutorial
77 Tutorial_salary_min = 0
78 Tutorial_salary_max = 10
79 Tutorial_salary_minSize = 0
80 Tutorial_salary_maxSize = 10
81
82 # Options
83 aggregationcyclefreeness = off
84 forbiddensharing = on
```