Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Annotation von anforderungsbezogenen Videos auf Basis von Ontologien

Masterarbeit

im Studiengang Informatik

von

Matthias Herrmann

Prüfer: Prof. Dr. Kurt Schneider
Zweitprüfer: Prof. Dr. Joel Greenyer
Betreuer: M.Sc. Oliver Karras

Hannover, 23.10.2017

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 23.10.2017

_____

Matthias Herrmann

Zusammenfassung

Die Erstellung und Betrachtung von Vision Videos ist eine in den frühen Phasen des Requirements Engineering genutzte Methode. Diese Videos werden erzeugt um prototypisch und beispielhaft Nutzungsszenarien eines noch nicht existierenden Systems darzustellen. Kunden und Entwickler sehen sie sich gemeinsam an und erarbeiten anhand der konkreten Visualisierung der Vision ein gemeinsames Verständnis des zu entwickelnden Produktes. Diese Vorgehensweise hat sich als Ergänzung zu textuell erhobenen Anforderungen bewährt. Obwohl ihre Produktion zeitlich und monetär aufwendig ist, werden sie aktuell nur für diesen einen Zweck benutzt. In dieser Arbeit wird ein Ansatz präsentiert, mit dem die Inhalte von Vision Videos in die Dokumentation aufgenommen werden können. Um für Entwickler und Werkzeuge späterer Phasen der Softwareentwicklung nützlich zu sein ist es notwendig, die dargestellten Konzepte und Beziehungen in Videos eindeutig in die Begrifflichkeiten der Kundendomäne einordnen zu können. Dazu wurde eine prototypische Software entwickelt, die dargestellte Menschen und Systemkomponenten grafisch annotierbar und von Begriffsklassen einer Ontologie ableitbar macht. Kunde und Requirements Engineer können damit entsprechende Teile von Vision Videos markieren und miteinander verbinden, so dass sie direkt im Video eindeutig hervorgehoben und in eine Domänen-Ontologie eingeordnet werden können. Evaluiert werden dabei der Nutzen und die Zugänglichkeit von annotierten Videos für Entwickler, die nicht direkt an der Anforderungserhebung teilnehmen aber deren Ergebnisse umzusetzen haben. Das Ziel dabei war es, annotierte Vision Videos so als Artefakt für die Dokumentation maschinenlesbar nützlich zu machen. Die Eignung von Darstellungsarten innerhalb von Filmen wurde dabei ebenso betrachtet, wie die grundsätzliche Eignung von Videos als ergänzendes Dokument für die Beseitigung von Klärungsbedarf.

Abstract

Creating and watching vision videos is a method used in the early stages of requirements engineering. Those videos are created to show prototypical and exemplary usage scenarios of a not yet existing system. Customers and developers watch them together to work out a common understanding of the product to be developed based on the concrete visualization of the vision. This approach has proven to be a useful supplement to the textual requirements. Although their production is time-consuming and costly, they are currently only used for this single purpose. This thesis presents an approach to include the content of vision videos in the documentation. In order to be useful for developers and tools of later phases of software development, it is necessary to be able to clearly classify the concepts and relationships presented in videos into the terms of a customer domain. For this purpose, a prototype software was developed, which enables a user to graphically annotate represented people and system components and, in addition, derive them from term classes of an ontology. Customers and requirements engineers may use it to mark and link appropriate parts of vision videos so that they can be clearly highlighted directly in the video and classified into concepts of a domain ontology. The benefits and accessibility of annotated videos are evaluated for developers who do not participate directly in the requirements assessment but have to implement results of it. The aim was to make annotated vision videos machine-readable as artifact for documentation. The suitability of presentation types within films was considered as well as the basic suitability of videos as a supplementary document for the elimination of clarification needs.

Inhaltsverzeichnis

1. Einleitung

Dieses Kapitel stellt eine Einführung in das Themengebiet dieser Arbeit dar. Zunächst wird auf die Beweggründe eingegangen, daraufhin die konkrete Problemstellung definiert, die in den folgenden Ausführungen bearbeitet wird. Abschließend wird in diesem Kapitel der gewählte Lösungsansatz beschrieben.

1.1. Motivation

Videos werden mittlerweile in der Softwareentwicklung zu verschiedenen Zwecken eingesetzt, wobei sich deren Verwendung speziell während der frühen Phase der Anforderungserhebung im Requirements Engineering als wirksam erwiesen hat [11, 20, 31].

Karras et al. [20] nutzen Mitschnitte von Requirements Workshops, da darin sehr viele Anforderungen und weitere wichtige Informationen über das zu entwickelnde Produkt genannt werden. Bei nachträglicher Betrachtung mit Hilfe eines Software Werkzeugs versehen sie wichtige Stellen in den Videos mit Notizen und stellen damit unter anderem sicher, dass keine in dem Workshop genannte Information verloren geht. Creighton et al. [11] verwenden visionäre Videos, die Szenen mit Interaktionen zwischen dem Benutzer und einem System darstellen, um daraus UML-Sequenzdiagramme zu generieren und diese dann für weitere Projektphasen zu nutzen. Damit sich Kunden und Entwickler ein gemeinsames Verständnis über ein zukünftiges System erarbeiten können, stellten Schneider et al. [31] den Ansatz sogenannter Vision Videos zur bildhaften Darstellung bei der Anforderungserhebung und -verhandlung mit dem Auftraggeber vor. Diese kurzen Filme dreht man bereits sehr früh im Projekt und sie enthalten prototypische Darstellungen davon, wie sich die Seite des Entwicklerteams die Verwendung eines zu entwickelnden Systems auf Basis bis dahin erhobener Rohanforderungen vorstellt. Sie zeigen damit ein noch nicht existierendes System visuell um das gemeinsame Verständnis darüber, wie das zu entwickelnde System später aussehen und funktionieren soll nicht nur schriftlich festzuhalten.

Laut Ebert [14] ist die Verständlichkeit von schriftlichen Anforderungen ein wesentliches Qualitätsmerkmal. Bewegte Bilder haben textuellen Beschreibungen gegenüber den Vorteil, dass sie weniger breit gefächerte Interpretationsmöglichkeiten für die verschiedenen Betrachter mit unterschiedlichem Vorwissen bieten. Die

Kommunikation zwischen Software Entwicklern und deren Kunden ist wegen des fehlenden Wissens über die jeweils andere Domäne grundsätzlich ein sehr sensibles Thema. Fischer [16] identifizierte eine Asymmetrie bezüglich des Wissens zwischen Diskussionspartnern mit unterschiedlichem fachlichen Hintergrund. Die daraus resultierende Gefahr ist die Bennenung von zu inkonkreten und sehr vagen Vorstellungen der Diskussionspartner. Aus Voreingenommenheit, sowie vorausgesetztem und daher nicht geäußerten Vorwissen entsteht eine *Asymmetrie des Wissens* oder *Symmetrie der Ignoranz*, was eine Erhebung von konkreten Anforderungen erheblich erschwert. Um dieser Gefahr entgegen zu wirken, nutzt man Vision Videos als Diskussionsgrundlage. Sie basieren auf bisheriger Kommunikation zwischen Kunden und Entwickler, wobei es Personen ohne Vorwissen bezüglich Software Entwicklung trotzdem ermöglicht wird direkt zu sehen, ob sie ihren Wünschen entspricht. Die vermittelnde Funktion des Requirements Engineers ist es allerdings nicht nur zu verstehen, was die Kundenwünsche genau sind, sondern auch diese in einer geeigneten Form für die spätere Entwicklung festzuhalten und somit zugänglich zu machen. Hat man auf RE-Seite den Kunden verstanden, so bedeutet das nicht zwangsläufig, dass auch Entwickler, die nicht direkt am Requirements Engineering beteiligt sind ebenfalls genauso gut verstehen, was der Kunde wünscht. Diese Entwickler sind darauf angewiesen, dass das Requirements Engineering ihnen auf sehr konkrete und unmissverständliche Art dokumentiert, was genau zu entwickeln ist.

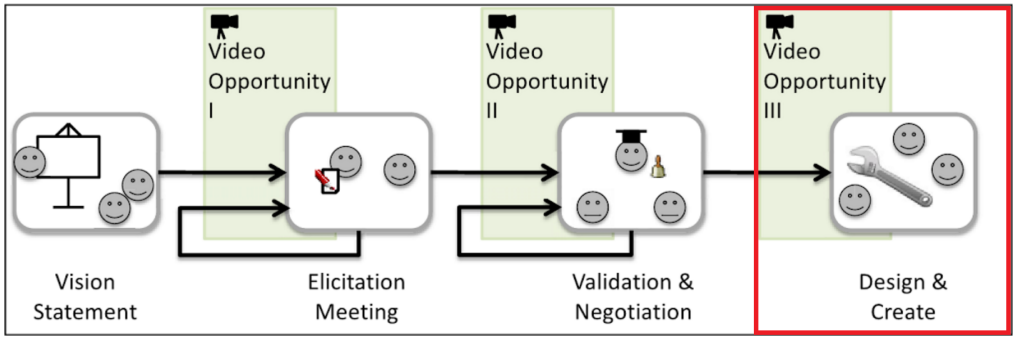


Abbildung 1.1.: Möglichkeiten zur Verwendung von Video Clips im Kontext des Requirement Engineerings von Brill [9], zusätzlich rot umrandet ist die für diese Arbeit gewählte Gelegenheit.

Brill et al. [9] beschreiben grundsätzliche Gelegenheiten der Videonutzung im Requirements Engineering, wobei Ad-hoc Videos erstellt werden um Anforderungen anhand einer visuellen Darstellung mit dem Kunden und weiteren Stakeholdern des Projektes besprechen zu können. Die Gelegenheiten befinden sich entweder innerhalb der Phasen des Requirements Engineerings oder grenzen daran, wie in Abbildung Abbildung 1.1 zu sehen. In den weiteren Phasen des Entwicklungsprozesses, welche in Abbildung Abbildung 1.1 als “Design and Create“ bezeichnet werden, könnten die bei

den Gelegenheiten zuvor erstellten Videos für Entwickler späterer Entwicklungsphasen geeignet sein. Artefakte dieser Art werden bisher hauptsächlich in textueller Form, beispielsweise als Stories oder Use Cases, oder aber in Diagrammform verwendet. Da Videos aber eine visuelle Darstellung mit direkt ersichtlichem Kontextbezug sind, ist es sinnvoll zu untersuchen, ob sie zur nachträglichen Beseitigung von Klärungsbedarf auf Entwicklerseite geeignet sind. Ebenfalls von Interesse ist die Art, wie sie den Entwicklern und vor deren Werkzeugen zugänglich gemacht werden können. Somit soll der Nutzen von Vision Videos für ganze Entwicklungsprojekte erhöht werden, denn deren Produktion kostet natürlich Zeit und Geld.

Definition 1.1 (Zentrale Problemstellung)

Das zentrale Problem dieser Arbeit bezieht sich auf das Verhältnis von Produktionskosten und Nutzwert von Vision Videos. Sie enthalten äußerst wertvollen Informationen, sind aber nicht ohne zeitlichen und monetären Aufwand zu produzieren. Da man sie bisher nur bei der Erhebung, Interpretation und Verhandlung von Anforderungen nutzt, wird ihr Wert für ein gesamtes Entwicklungsprojekt nicht voll ausgeschöpft.

1.2. Lösungsansatz

Da Vision Videos prinzipiell Instanzen von Konzepten und Beziehungen zwischen diesen enthalten, sollten diese Inhalte über einheitliche Domänenbegriffe in die Dokumentation einbezogen werden können. Um diese Zugänglichkeit formal für Maschinen und Menschen zu ermöglichen, bietet sich der Formalisierungsansatz über Ontologien an. Damit kann erarbeitetes Wissen über bestimmte Domänen festgehalten und geteilt werden. Nach Gruber [17] ist eine Ontologie “eine formale, explizite Spezifizierung einer gemeinsamen Konzeption“. Verwendete Begriffe und Relationen sind darin explizit definierbar, stellen einen zwischen Auftraggeber und Auftragnehmer geteilten Wissensraum dar und werden auf formale Art für Softwarewerkzeuge lesbar.

In dieser Arbeit wird der Versuch unternommen Inhalte von Vision Videos über eine Ontologie in die Dokumentation einzubeziehen. Die im Requirements Engineering erarbeiteten Videoinhalte sollen so auch weiteren Entwicklungsphasen schnell und zielgerichtet zugänglich gemacht werden. Dies soll mit einer Annotation von Bildbereichen über ihre Sichtbarkeitszeit geschehen, so dass ein Videoinhalt präzise einem Domänenbegriff zugeordnet und unabhängig von der Länge des ihn zeigenden Filmes schnell gefunden und gesichtet werden kann.

Um einen Lösungsansatz für das beschriebene Problem zu erarbeiten, musste zunächst die konkret zu behandelnden Aufgaben und Teilprobleme eingegrenzt werden. Die

Anforderungen an diese Arbeit enthielten neben klar formulierten Aufgaben auch gewisse Entscheidungsfreiheiten in Bezug auf Anforderungen an den zu entwickelnden Ansatz. Er sollte auf jeden Fall die grafische Annotation von Vision Videos ermöglichen und diesen auf Basis von Ontologien realisieren. Zusätzlich sollten die Ontologien in der *Web Ontology Language (OWL)* [2] spezifiziert werden. Wahlmöglichkeiten bestanden bezüglich der Verwendungsart einer Ontologie zusammen mit grafischer Videoannotation. Hier galt es zu entscheiden, ob man beim Annotieren eine Ontologie aufbauen oder eine bestehende verwenden können sollte.

Für die Entscheidung darüber wurden zu Beginn der Erarbeitung einer Konzeption 3 zunächst Vor- und Nachteile der jeweiligen Möglichkeiten gegeneinander abgewogen. Die Entscheidung fiel auf die Verwendung einer bestehenden Ontologie, da der Verfasser darin mehr Vorteile sowohl für das Software Engineering, als auch für die zeitliche Planung und Risikobehandlung seiner Arbeit sah. Zunächst wäre der Programmieraufwand, einen Ontologie-Editor zusätzlich zur geforderten visuellen Annotation von Videos zu entwickeln deutlich höher ausgefallen, womit das Risiko gar nicht oder nicht rechtzeitig fertig zu werden stark gestiegen wäre. Auch existieren bereits etablierte Ansätze mit Softwarelösungen für den Aufbau von Ontologien, wie beispielsweise *Protégé* [25] von der Stanford Universität in Kalifornien.

Im Hintergrund der Ausschreibung wurde ebenfalls erwähnt, dass die zu annotierenden Vision Videos als Instanzen von Konzepten, deren Eigenschaften und Beziehungen eines zu entwickelnden Systems angesehen werden. Konsequenter Weise wurde daraufhin entschieden, dass die Verwendung einer bestehenden Ontologie als Basis für die Instanzen in Videos einen sinnvolleren Ansatz darstellt. Aus Ontologien, die eine sprachlich formale Darstellung von Wissen sind können Instanzen erstellt werden, die dem Ontologieschema entsprechen. Nachdem zusätzlich noch einmal deutlich wurde, wozu Vision Videos erstellt und genutzt werden wurde ersichtlich, dass der Aufbau einer neuen Ontologie weniger sinnvoll für eine Annotation von Vision Videos ist. Man klärt nämlich gemeinsam mit dem Auftraggeber anhand der bewegten Bilder, ob die gezeigte Vision seinen Anforderungen entspricht. Dazu ist es wesentlich effizienter und fokussierter, die Begrifflichkeiten der Kundendomäne vorher geklärt zu haben und dann währenddessen aufzugreifen, anstatt sie bei der Sichtung von möglichen Verwendungen des zu entwickelnden Systems erst zu klären und damit ein Wissensschema aufzubauen. Man würde sonst entweder die eigentliche Aufgabe beim Ansehen der Videos aus den Augen verlieren oder müsste spezielle Videos zur Klärung von Domänenbegriffen einführen um einem Ontologieaufbau eine sinnvolle Bedeutung beizumessen. Domänenkonzepte sollten daher zuvor geklärt sein und deren Erläuterung nicht den Fokus auf Nutzungsszenarien ablenken. Einzig die Tatsache, dass Videos unter Umständen Inhalte haben, die nicht aus Konzepten einer bestehenden Ontologie abgeleitet werden können, würde für den Aufbau während der Annotation sprechen. Da dies aber ein sehr kleiner Aspekt ist, hat man sich dazu entschieden ihn auf andere Art zu behandeln. Speziell für diesen Fall wird eine Konzeptklasse eingeführt, der ein solcher Inhalt zugeordnet werden kann.

Das in Abschnitt 1.1 herausgestellte Ziel ist es, die bisherigen Verwendungsmöglichkeiten von Vision Videos zu erweitern und damit ihren Nutzen zu steigern. Vision Videos haben sich zur Erarbeitung einer geteilten Vorstellung von Menschen mit unterschiedlichem fachlichen Hintergrund bewährt. Die finalen Video-Versionen halten die geteilte Vision in bewegten Bildern fest und sind damit mindestens so wertvoll wie erhobene, interpretierte und verhandelte textuelle Anforderungen. Trotzdem fließen sie bisher gar nicht oder nur in wenig zugänglicher Form in die Dokumentation ein, obwohl sie potenziellen Klärungsbedarf auch in späteren Phasen der Softwareentwicklung beseitigen können. Es gilt als einer der schwierigsten Kommunikationsteile innerhalb eines Software-Entwicklungsprojektes, mit seinem Kunden und weiteren Stakeholdern genau zu klären, was zu entwickeln ist. Ergebnisse aus dieser Klärung sind also grundsätzlich auch wertvoll für Entwickler, die nicht mit dem Kunden persönlich kommunizieren aber Ergebnisse aus dem Requirements Engineering zu verarbeiten haben. Diese könnten von einer alternativen und visuell kontextreichen Darstellung von Anforderungen profitieren.

Man benötigt dazu allerdings ein Konzept, welches die Videos mit den Begrifflichkeiten der Domäne und möglicherweise auch weiterem Wissen über das Projekt verknüpft. Um den Wiederbetrachtungswert der Videos zu erhöhen, sollten diese allen Entwicklern des Projektes zur Verfügung stehen und der Bezug zur aktuellen Arbeit des Entwicklers hergestellt sein. Voraussetzung dafür ist allerdings auch eine verbesserte Zugänglichkeit für Maschinen beziehungsweise Programme, die bei der Gestaltung und der Implementierung des Produktes verwendet werden. Über einen Ontologie-basierten Ansatz kann die Zugänglichkeit für Maschinen durch formale Spezifikation erhöht werden. Für Menschen ist wiederum eine verbesserte Zugänglichkeit der Inhalte von Videos über ihre Software-Werkzeuge von Nutzen und kann den Wiederbetrachtungswert von Vision Videos steigern. Um dies zu erreichen wurde zunächst mit der Methode *Goal-Question-Metric* ein ZielbaumAbbildung 1.2 ermittelt. Dabei wird das oberste und am wenigsten konkrete Ziel G.1 formuliert und daraus weitere, konkreter und messbar werdende Ziele abgeleitet.

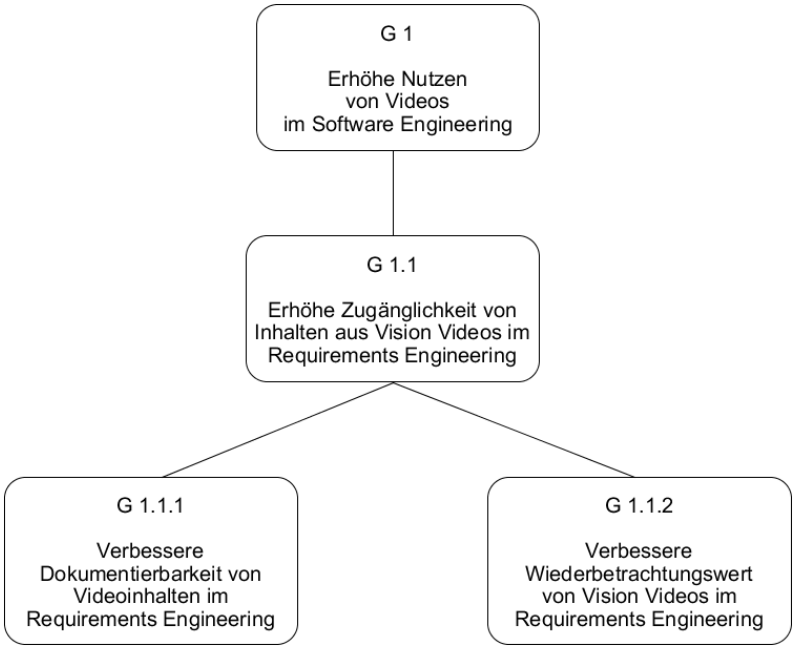


Abbildung 1.2.: Zielbaum für diese Arbeit

Zielsetzung der Arbeit

Die Arbeit hat zum Ziel, die Akzeptanz von Videos als Artefakt innerhalb der Dokumentation zu evaluieren und weiter festzustellen, inwieweit ein Ontologie-basierter Ansatz für die Verknüpfung von Videoinhalten mit erarbeitetem Wissen über das Entwicklungsprojekt dienen kann. Zu Wissen über das Projekt zählen sowohl Produkte des Requirements Engineerings, als auch Dokumente und Artefakte, die in späteren Entwicklungsphasen entstehen und die alle auf den Begrifflichkeiten der Kundendomäne aufbauen. Dargestellte Szenen in visionären Filmen werden dabei als Instanz einer Ontologie angesehen, die gesammeltes Wissen mindestens über jene Kundendomäne umfasst. Auf diese Weise soll auf Basis festgelegter Begriffe eine explizite Einordnung von Videoinhalten möglich werden. Einen Ansatz zur formalen Spezifikation einer Ontologie zu erarbeiten, welche möglichst viel Wissen über das Entwicklungsprojekt vereint ist somit wünschenswert und eine diesbezügliche Basis zu erarbeiten zwangsläufig ebenfalls Ziel dieser Arbeit.

Definition 1.2 (Ziel dieser Arbeit)

Das allgemeine Ziel ist es einen Weg zu finden, mit dem man den Wiederbetrachtungswert von mit Kunden besprochenen Videos erhöhen kann und deren Inhalte möglichst einfach und schnell für spätere Entwicklungsphasen verfügbar zu machen.
Das konkrete Ziel der Arbeit ist es, Inhalte von Vision Videos darin hervorzuheben und über eine formal spezifizierte Ontologie zu verknüpfen, so dass Entwickler und Software späterer Projektphasen einen möglichst direkten und einfachen Zugang dazu bekommen.

1.3. Struktur der Arbeit

Diese Arbeit ist in 6 Kapitel untergliedert und besitzt die folgende inhaltliche Struktur. Das Kapitel1 dient dem Einstieg in die Motivation und Zielsetzung der Arbeit. Welche Problematiken der Arbeit zugrunde liegen und warum diese sich zu untersuchen lohnen wird in darin erörtert. Die wissenschaftlichen Grundlagen bezüglich der Themen Videoeinsatz im Software Engineering, genauer gesagt im Requirements Engineering, und der Arten, sowie Anwendungsgebiete von Ontologien werden in Kapitel 2 dargelegt. Dabei werden Begriffe erläutert, sowie Beispiele genannt und die Einsatzgebiete näher erörtert. Das Kapitel 3 zeigt auf, wie die Anforderungen an diese Arbeit in eine Konzeption zur Entwicklung einer prototypischen Software umzusetzen sind. Dabei werden die Rahmenbedingungen für eine Annotation von Vision Videos auf Basis einer Ontologie erarbeitet. In Kapitel 4 beinhaltet die praktische Umsetzung der Aspekte des im vorangegangenen Kapitel erarbeiteten Konzepts, nämlich die prototypische Umsetzung einer Videoannotations-Software auf Basis von Ontologien. Dabei wird begründet auf die Designentscheidungen und Features des Prototypen eingegangen, die verwendeten Technologien näher erklärt und abschließend Problemstellungen betrachtet, die sich bei der Erstellung und dem Ausprobieren des Prototypen ergaben. Kapitel 5 widmet sich dem mit der entwickelten Software durchgeführten Experiment. Der Aufbau des Experiments und die verfolgten Ziele werden erklärt, sowie die Durchführung und Population der Teilnehmer erläutert. Im sechsten und letzten Kapitel dieser Arbeit wird ein Fazit zum erarbeiteten Ansatz, dessen Umsetzung und Evaluation gezogen. Zusätzlich beinhaltet Kapitel 6 einen Ausblick auf Ergebnisse dieser Arbeit, deren Weiterverfolgung sich der Meinung des Autors nach lohnen könnte.

2. Grundlagen

Dieses Kapitel enthält die wesentlichen Grundlagen für das Verständnis der weiteren Ausführungen dieser Arbeit. Es stellt die verwendeten Konzepte, Verfahren und Technologien dar. Da es um eine Annotation von Vision Videos auf Ontologie-Basis geht, muss zunächst erläutert werden, wo diese Videos her kommen, welche Eigenschaften sie besitzen und welchem Zweck sie dienen. Dazu wird zuerst eine Einführung in das Requirements Engineering und den darin produzierten und verwendeten Vision Videos gegeben. Folgend wird das Prinzip von Ontologien und deren Anwendung zur Formalisierung von geteiltem Domänenwissen vorgestellt, um den Sinn der Ontologie-Basiertheit der Annotation von Videos zu erklären.

2.1. Requirements Engineering

Der Prozess des Requirements Engineerings (RE) wird zu Beginn eines Entwicklungsprojektes durchgeführt und besteht aus Verfahren und Techniken zur Analyse und Verwaltung von Anforderungen an das zu entwickelnde Produkt [7]. Er ist grob unterteilt in zwei hauptsächliche Aufgabengebiete, die Requirements Analysis und das Requirements Management. Wie in Abbildung 2.1 zu sehen, sind Analysis und Management je weiter untergliedert in detailliertere Aufgabenbereiche beziehungsweise -phasen. Diese Phasen sind nicht unbedingt als strenge einzuhaltende Abfolge nacheinander zu sehen, viel mehr können sie auch durchaus parallel zueinander oder sich wiederholend ablaufen. Im folgenden wird für jede Phase des Requirements Engineerings kurz zusammengefasst, welche Ziele sie verfolgt und es werden einige Beispiele für Aktivitäten gegeben. Die Erläuterungen, genannten Begriffe und Beispielaktivitäten beruhen auf Börger [7], Schneider [32], Rupp [28, 29] und dem Standard ISO/IEC/IEE 29148:2011 [12].

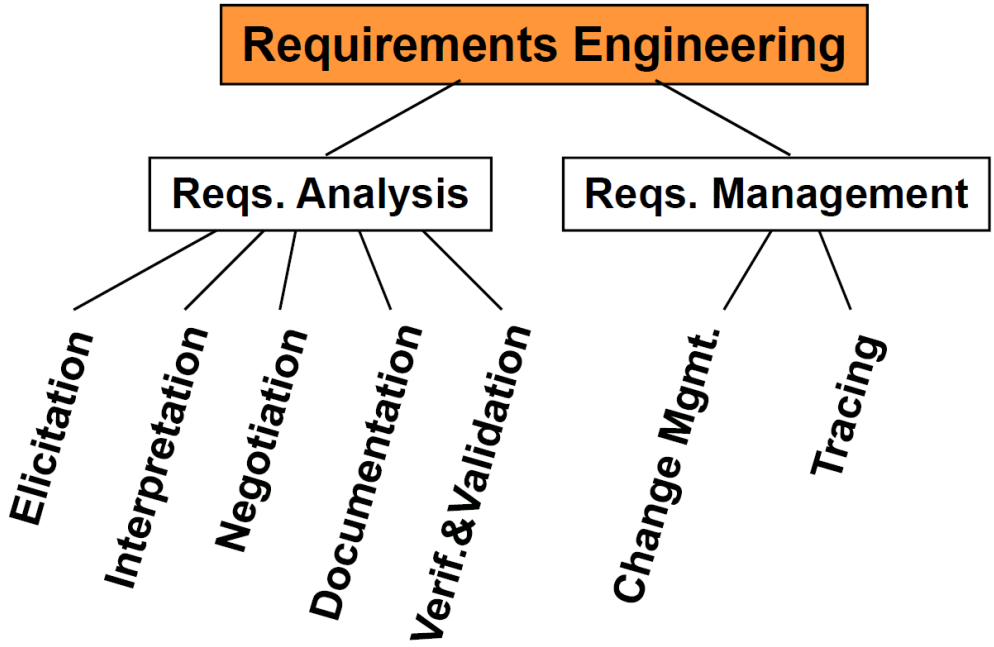


Abbildung 2.1.: Das Referenzmodell für das Requirements Engineering nach Börger et al. [7]

2.1.1. Requirements Analysis

Die *Requirements Analysis* oder Anforderungsanalyse verfolgt mehrere Ziele. Es gilt Anforderungen zu ermitteln, zu erfassen, zueinander in Beziehung zu setzen und dies alles zu dokumentieren. Dabei ist es sowohl für Entwickler, als auch Kunden, als auch jegliche weitere Stakeholder von großer Bedeutung, ein gemeinsame Basis für Verständnis und Wissen über die Anforderungen herzustellen. Die Anforderungsanalyse findet zu Beginn eines Entwicklungsprozesses statt und ist in die fünf im folgenden näher erklärten Phasen unterteilt (Abbildung 2.1).

Elicitation

Die erste Phase wird als *Elicitation* oder Anforderungserhebung bezeichnet und bildet den Beginn des Requirements Engineering. Zuallererst sucht man den Kunden auf und ermittelt diejenigen Personen oder Personengruppen, die direkt oder indirekt von dem zu entwickelnden Softwareprodukt betroffen sind. Diese werden zusammenfassend als *Stakeholder* bezeichnet und von besonderem Interesse deren genaue Beziehung zum System. Damit ist beispielsweise ein späterer Benutzer ein Stakeholder, aber auch schon der Auftraggeber selbst, da er oder sein Geschäft vom Erolg des Projektes abhängen. Die Ermittlung von betroffenen Personen dient dem allgemeinen Ziel

der Elicitation, welches das Sammeln möglichst umfangreichen Wissens über das System ist. Durch die vielen verschiedenen Sichtweisen der einzelnen Stakeholder entsteht ein breites Spektrum an *Rohanforderungen*. Diese sind noch nicht strukturiert oder miteinander abgeglichen, es sind lediglich festgehaltene Ergebnisse aus Befragungen oder Beobachtungen der Stakeholder, wie zum Beispiel aus Interviews oder Workshops [29, 32]. Festgehalten werden Rohanforderungen auf unterschiedliche Art, allerdings zumeist in schriftlicher Form, wie in Use Cases, Stories oder schlichten Prosa-Formulierungen [28, 29, 32].

Interpretation

In der zweiten Phase, der *Interpretation* geht es darum, die genaue Bedeutung der erhobenen Rohanforderungen aus der Elicitation herauszuarbeiten. Das Ziel der Interpretation ist es, konkrete Anforderungen zu erhalten, sie zu strukturieren, zu klassifizieren und miteinander in Beziehung zu setzen. Anforderungen können unverständlich oder abstrakt formuliert sein, voneinander abhängen oder sich sogar gegenseitig ausschließen. Abstraktes muss konkretisiert und Beziehungen müssen identifiziert werden. Eine klare Anforderungsstruktur ist notwendig, damit man eine übersichtliche und konkrete Grundlage für die Verhandlung mit dem Auftraggeber hat. Damit man das Ziel dieser Phase erreicht, werden alle vorliegenden Dokumente genau analysiert, was sowohl die Produkte der Elicitation, als auch von Kunden bereitgestellte Dokumente, wie zum Beispiel bestehende Spezifikationen von angrenzenden Systemen einschließt. Auf Grundlage der interpretierten Rohandforderungen können bereits erste Visionen des Systems in Videoform erstellt werden [32, 29], die man dann in der nächsten Phase mit dem Auftraggeber verhandelt.

Negotiation

Die dritte Phase ist die *Negotiation*, also die Verhandlung von interpretierten Rohanforderungen mit dem Kunden [32, 28]. Man möchte erreichen, dass die beteiligten Stakeholder und Entwickler eine gemeinsame und einheitliche Grundlage von Anforderungen bekommen und diese alle Betroffenen möglichst zufrieden stellt. Dazu präsentiert man den Stakeholdern zuvor identifizierte Anforderungskonflikte und versucht diese aufzulösen. Man setzt ähnliche Aktivitäten wie in der Elicitation ein, da an der Auflösung von Konflikten prinzipiell alle Stakeholder beteiligt sein müssen. Workshops mit verschiedenen Stakholdern sind hierbei von größter Wichtigkeit, da identifizierte Konflikte meist hauptsächlich auf unterschiedliche Aussagen zum selben System oder Teilen davon beruhen.

Documentation

In der vierten Phase, der *Documentation* geht es darum, die Ergebnisse der vorherigen Aktivitäten in geeigneter, also möglichst unmißverständlicher Weise festzuhalten. Dokumentiert wird grundsätzlich bereits während der Elicitation, Interprettation und Negotiation, allerdings sind die Arten der Dokumentation noch nicht vollkommen darauf ausgerichtet, Anforderungen für die weitere Entwicklung zu optimieren. Es existieren beispielsweise Techniken, wie das Neuro-Linguistische Programmieren [27], die Formulierungen klarer, verständlicher und weniger interpretierbar machen. Dies ist für schriftliche Dokumente von großer Wichtigkeit, denn man fördert so die Verbesserung und Vervollständigung der Gesamtspezifikation. Für andere Arten von Dokumenten, wie beispielsweise Vision Videos, gibt es noch keine Regeln für ihre Beschaffenheit, was auch ein Grund dafür ist sie noch nicht wirklich in die Dokumentation zu integrieren [32].

Verification & Validation

Bei der fünften Phase der Anforderungsanalyse handelt es sich um die *Verification & Validation*. Dabei gilt es zu prüfen, ob die erhobenen und mittlerweile dokumentierten Anforderungen korrekt sind. Dies beinhaltet zwei verschiedene zu prüfende Aspekte, nämlich einerseits den Kundenwunsch und andererseits die zu Beginn erhobenen Rohanforderungen. Man verfolgt mit der *Validation* einerseits das Ziel, den Kundenwunsch vollkommen zu erfüllen und andererseits mit der *Verification* entstandene Abweichungen zu anfänglich erhobenen Rohanforderungen zu klären. Es wird *validiert*, ob die dokumentierten Anforderungen dem Kundenwunsch tatsächlich entsprechen. Ebenfalls *verifiziert* man, ob die dokumentierten Anforderungen den in der Elicitation erhobenen Rohandforderungen entsprechen und muss eventuelle Unstimmigkeiten mit den betroffenen Stakeholdern gesondert besprechen [32].

2.1.2. Requirements Management

Das *Requirements Management* befasst sich mit der Verwaltung von dokumentierten Anforderungen und zugehörigen Artefakten. Das Ziel dabei ist das Verwahren von verfolgbaren Dokumenten zu Anforderungen und damit auch die Verwaltung und Dokumentation von Änderungen [32].

Change Management

Mit dem *Change Management* sind diejenigen Tätigkeiten nach der Anforderungsanalyse gemeint, die sich auf Änderungen von Anforderungen beziehen [32]. Da sich Anforderungen im Laufe eines Entwicklungsprozesses ständig ändern können und dies auch sehr häufig tun, kann man nicht einfach nach einer Anforderungsanalyse das Requirements Engineering für beendet erklären. Da sich auch für den Auftraggeber Änderungen selbst ohne sein Zutun ergeben können, muss man darauf vorbereitet

sein Änderungen auch in ein weit fortgeschrittenes Projekt zu integrieren. Dazu dient ein systematisches Vorgehen im *Change Management*. Dies beginnt mit dem formellen Einreichen eines *Change Request*, also eines Änderungswunsches. Dieser beschreibt die notwendig gewordenen Änderungen und stellt weitere Informationen dazu bereit, so zum Beispiel deren Gründe und Dringlichkeit, sowie betroffene Systemtteile und Stakeholder. Ein eigens zu diesem Zweck innerhalb des Projektes installiertes Gremium, das *Change Control Board* entscheidet nach Einreichung einer *Change Request* darüber, ob, wie, mit welcher Priorität und zu welchem Preis die nachträgliche Änderung durchgeführt wird. Das *Change Control Board* sollte nicht einseitig besetzt sein, sondern viel mehr sowohl aus Mitgliedern der Kundenseite, als auch des Entwicklerteams bestehen. Selten aber ist es möglich, alle Änderungen auch tatsächlich zu integrieren.

Tracing

Das *Tracing* unfasst die Tätigkeiten in einem Softwareprojekt, die notwendig sind um Beziehungen zwischen und Änderungen an Anforderungen jederzeit nachvollziehen zu können [32]. Man verfolgt das Ziel den Gesamtzusammenhang so zu dokumentieren, dass alle Beteiligten des Projektes immer in der Lage sind die Evolution von für sie wichtigen Informationen zu verfolgen. Dabei wird zwischen *Pre-Tracing* und *Post-Tracing* unterschieden. Das *Pre-Tracing* dient der Rückverfolgbarkeit einer Anforderung bishin zu ihrem Ursprung in der Anforderungserhebung. *Post-Tracing* bezieht sich hingegen auf die Auswirkungen, welche eine Anforderung auf das gesamte System hat und deren Nachverfolgbarkeit. Man verknüpft dokumentierte Anforderungen und zugehörige Artefakte mit sogenannten *Trace Links*, wodurch ein Graph entsteht, der mittels formaler Methoden aus der Graphentheorie analysiert werden kann. Man benötigt das Wissen aus beiden Arten des *Tracing* für eine Bewertung von Entwurfs- und Implementierungsentscheidungen, sowie zur begründeten Entscheidung über nachträglich eingereichte Änderungswünsche.

2.1.3. Vision Videos

Vision Videos können als ein Produkt des Requirements Engineerings(2.1) angesehen werden, da sie innerhalb der Requirements Analysis(2.1.1) erstellt und verwendet werden. Um der Kommunikation mit dem Kunden zusätzlich zur verbalen und schriftlichen auch eine anschauliche Dimension abseits von formalen Modellen zu geben, führte man die Verwendung von Video Clips im Requirements Engineering ein [11, 32]. Als Vision Videos bezeichnet man kurze Filme, die Funktionen oder Teilfunktionen des zu entwickelnden Systems darstellen um für Kunden und Entwickler eine gemeinsame Diskussionsgrundlage und ein gemeinsames Verständnis des Softwareproduktes festzulegen. Sie werden typischerweise in der Elicitation-Phase des Requirements Engineering erstellt und mit dem Kunden gemeinsam angesehen. Beide Seiten bekommen darüber eine gemeinsame Vorstellung von Problemstellungen, Lösungsvisionen und -ansätzen. Nimmt man die *Modes of Communication* von

Ambler [1] (Abbildung 2.2 und darin die Effectivität der Kommunikation und die Reichhaltigkeit des Kommunikationskanals Videotape, dann ist der Vorteil einer Diskussion anhand eines Videos zu erkennen. Man kann sehen, dass ein Videotape an sich bereits die reichhaltigste Dokumentationsption darstellt, aber in Bezug auf Modellierung noch nicht ergiebig genug ist. Eine Kombination mit einer Konversation von Angesicht zu Angesicht allerdings verhilft dem Videotape zu einer größeren Bedeutung. Anhand von selbst gedrehten Videos Anforderungen mit dem Auftraggeber zu klären wird dadurch zu einer lohnenswerten Tätigkeit. Gerade bei unterschiedlichen fachlichen Hintergründen der Teilnehmer sind textuelle Formulierungen eher weniger effektiv und können unterschiedliche Interpretationen zulassen.

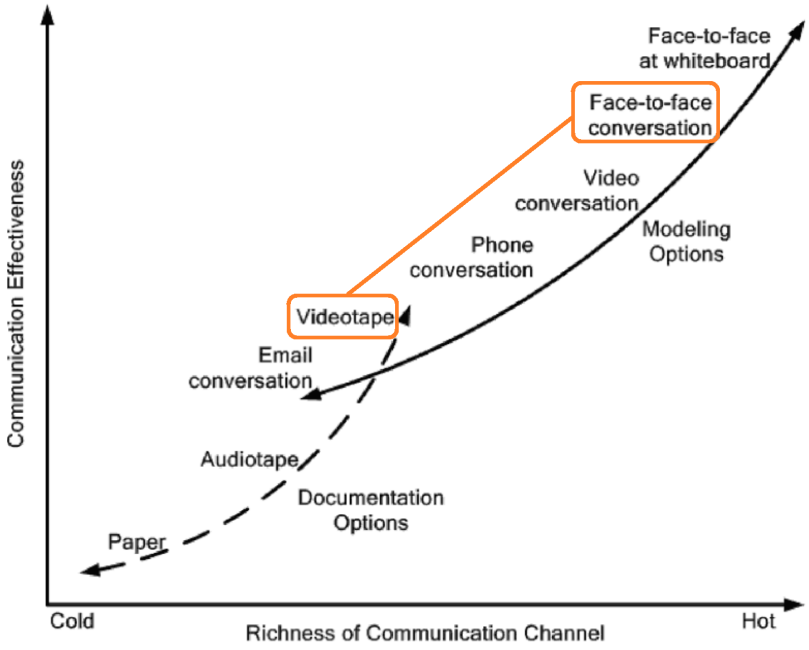


Abbildung 2.2.: Die für Vision Videos bedeutsamen Modellierungs- und Kommunikationsoptionen (orange markiert) aus den *Modes of Communication* nach Ambler [1]

Vision Videos zeigen prototypisch und beispielhaft wie man sich die Verwendung eines zukünftigen Systems in seiner Zielumgebung vorstellt, obwohl das System noch nicht oder nur zum Teil existiert. Das ist ihr grundsätzlicher Unterschied zu Prototypen, die ihrerseits auch zur Anforderungsklärung genutzt werden, aber das System in Teilen bereits implementieren. Die Verwendung von Vision Videos ist bislang auf die frühen Phase der Anforderungsanalyse beschränkt, ihre Inhalte sind allerdings potenziell ebenfalls sehr wertvoll für spätere Phasen des Entwicklungsprozesses. Das ist besonders dann der Fall, wenn ein Kunde die visionäre Darstellung als seinen Vorstellungen entsprechend bezeichnet. Ungeachtet des kommunikativen Wertes von

Vision Videos für den direkten Kundenkontakt enthalten finale Versionen nach der Verhandlung eine Darstellung von Szenarien, welche für die Dokumentation ebenfalls von Wert sein kann.

Genauso wie schriftliche Anforderungen gehört das Artefakt Video prinzipiell ebenfalls zur Dokumentation, hat aber den Nachteil der weniger ausgeprägten Zugänglichkeit bezüglich seiner Inhalte. Die Gründe dafür sind vielfältig, so haben Videos einen höheren zeitlichen Aufwand um komplett angesehen zu werden als der Aufwand, schriftlich ausformulierte Anforderungen zu lesen. Ein Entwickler, der Schwierigkeiten hat eine schriftlich dokumentierte Anforderung umzusetzen kann anhand einer visuellen Darstellung mittels Video Clip Klärungsbedarf beseitigen. Der Inhalt des Videos stellt nämlich auf alternative Art mindestens auch die schriftliche Anforderung dar und damit ein breiteres Spektrum an Informationen bereit. Zu einer Verwendung dieser Inhalte bedarf es allerdings einer sinnvollen Verknüpfung derer mit Begriffen der Kundendomäne und weiter auch mit den schriftlichen Anforderungsdokumenten. Nun wäre es sicher deutlich zu aufwendig, für jede einzelne ausformulierte Anforderung ein Video zu drehen oder sogar von einer professionellen Videoproduktionsfirma drehen zu lassen, wie Creighton et al. [11] empfehlen. Daher widmet sich diese Arbeit der Suche nach einem Ansatz, die erwähnte Verknüpfung von Videoinhalten und Domänenbegriffen im Kontext des Entwicklungsprojektes zu ermöglichen.

Hoher Produktionsaufwand und der nicht triviale Aufbau von Videos im Allgemeinen haben bisher die Integrationsmöglichkeiten von Videoartefakten in die Dokumentation und weiterführende Phasen der Softwareentwicklung eingeschränkt [15]. Die Art und Weise der Darstellung einer Systemvision für Vision Videos ist bisher auch noch keinen fest definierten Prinzipien und Regeln unterworfen. Da es nicht üblich ist Videos in wissenschaftliche Arbeiten direkt einzubetten, sind Beispiele dazu in dieser Arbeit als Einzelbilder dargestellt. Die Abbildungen 2.3, 2.4 und 2.5 zeigen Beispiele für die Verschiedenartigkeit, mit der Inhalte in Vision Videos dargestellt sein können.

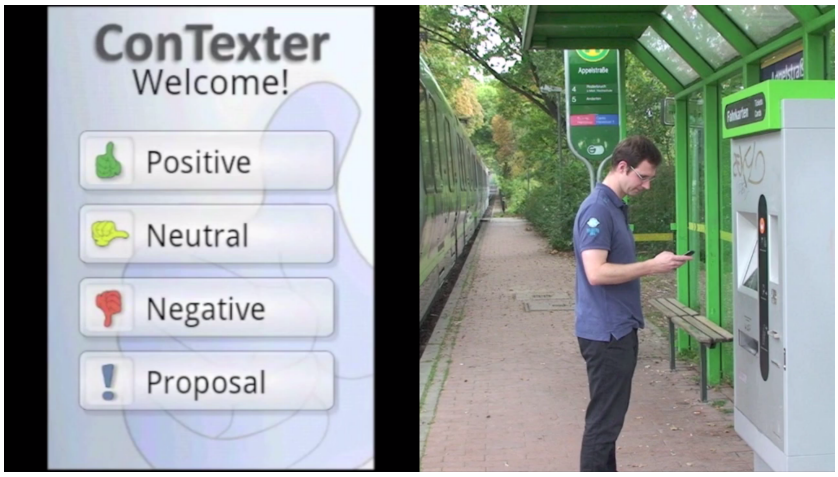


Abbildung 2.3.: Smartphone-App: Geteilter Bildschirm

Abbildung 2.4.: Tablet-App: Abgefilmter Papier-Prototyp

Abbildung 2.5.: Beispiel für ein Vision Video: Mit Schauspielern gedrehte Szene

Genauso, wie Regeln zur Formulierung möglichst unmißverständlicher textueller Anforderungen aufgestellt wurden [27], sollte es eigentlich auch Regeln für die Darstellung von möglichst unmißverständlichen Szenarien innerhalb von Videos geben. Diese Videos sind zum selben Zweck erstellt worden, nämlich der Erarbeitung von gemeinsamen Vorstellungen für Auftraggeber und alle Entwickler des Projektes. Da die Art und Weise, wie Szenen innerhalb eines Filmes aussehen sehr unterschiedlich sein kann und sich daher auch unterschiedlich gut für eine grafische Annotation auf Ontologie-Basis eignen, wird darauf in der Konzeption des Lösungsansatzes in Kapitel 3 noch genauer eingegangen.

2.2. Ontologien

Der Begriff Ontologie wurde zuerst in der Philosophie geprägt und leitet sich aus den griechischen Worten *on* und *logos* ab. Im Philosophischen Lexicon von Walch [35] ist er im 18. Jahrhundert als "(...) die Lehre vom Ende" bzw. "(...) die Lehre vom Sein" definiert. Damit bezeichnet die Ontologie die Lehre von den grundsätzlichsten Eigenschaften, den Prinzipien, den grundsätzlichsten Wesens-, Ordnungs- und Begriffsbestimmungen des Seins. Im Kontext der Informatik hat der Begriff Ontologie eine andere Bedeutung, obgleich diese Bedeutung von der philosophischen Definition beeinflusst wurde.

Menschen, Organisationen und Software Systeme verschiedener Bedürfnisse und Hintergründe müssen bei der Erarbeitung von Anforderungen miteineinander kommunizieren. Dabei existieren breit variierende Annahmen über und Blickwinkel auf ein und dasselbe Thema, wobei die teilnehmenden Menschen und Systeme verschiedende und sich teilweise überschneidende Begriffe, Methoden und Strukturen verwenden. Der daraus resultierende Mangel an geteiltem Verständnis führt zu folgenden Problemen bei der Entwicklung von Software Systemen [33]:

- schlechte Kommunikation zwischen den beteiligten Menschen und Organisationen mit unterschiedlichem fachlichen Hintergrund
- Schwierigkeiten bei der Identifikation und damit der Spezifikation von Anforderungen an ein zu erstellendes IT-System
- Uneinheitliche Methoden, Paradigmen und Sprachen, die zur Modellierung benutzt werden, was die Interoperabilität und das Potenzial zur Wiederverwendung und zum Teilen von Wissen einschränkt
- Hohem Aufwand, Wissen immer wieder neu zu erheben.

Eine Definition von Ontologien von Gruber und Borst [17] fasst treffend deren wichtigsten Eigenschaften und Ziele zusammen:

"*An ontology is a formal, explicit specification of a shared conceptualization.* ***Conceptualization*** *refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon.* ***Explicit*** *means that the type of concepts used, and the constraints on their use are explicitly defined.* ***Formal*** *refers to the fact that the ontology should be machine-readable.* ***Shared*** *reflects that notion that an ontology captures consensual knowledge that is, it is not private of some individual, but accepted by a group.*"

Ontologien zeichnen sich also dadurch aus, dass sie Begriffe für geteiltes Wissen festlegen, die von einer Gruppe Menschen akzeptiert werden und maschinenlesbar sind. Damit eignen sich Ontologien prinzipiell für eine vermittelnde Rolle in einem Software-Entwicklungsprojekt, bei dem generell Kunden, Entwickler und deren Werkzeuge Zugang zum erarbeiteten Wissen benötigen. Mit dem OWL Standard [2, 30, 23] und vorhandener Software zum Erstellen von Ontologien [25, 22, 24] erhalten Domänenexperten die Möglichkeit selbst solche zu entwickeln und direkte Kontrolle über deren Verhalten erlangen.

Das im Folgenden aufgebaute Beispiel für eine Ontologie befindet sich im Kontext der Mensch-System-Interaktion. Parallel zur Erläuterung der einzelnen Elemente einer Ontologie wird in den folgenden Unterabschnitten jeweils eine grafische und eine formale, maschinenlesbare Darstellung gezeigt. Da keine einheitlichen Vorgaben für die visualisierung von Elementen einer Ontologie existieren, hat man sich für dieses Kapitel auf eine konsistente grafische Notation festgelegt und diese in den jeweiligen Passagen kurz benannt. Eine formale und maschinenlesbare Darstellung wird später im Abschnitt 4.2 in der Ontologiesprache *Turtle* angegeben, die vom *World Wide Web Consortium* als textuelle Repräsentation eines RDF-Graphen 2.2.4 entwickelt wurde [4]. Diese greift dort die folgenden, grafischen Beispiele auf.

2.2.1. Bestandteile einer Ontologie

Eine Ontologie besteht grundsätzlich aus 5 wesentlichen Bestandteilen, welche in diesem Abschnitt erläutert werden. Diese sind Begriffe beziehungsweise Konzepte, Relationen, Attribute, Instanzen und Axiome.

Begriffe / Konzepte

Konzepte sind abstrakte Begriffe, ähnlich abstrakten Klassen oder Oberklassen und stellen Elemente beziehungsweise Entitäten in einer Ontologie dar. Die Worte *Konzept*, *Begriff* und *Klasse* werden in diesem Kontext häufig synonym verwendet, in dieser Arbeit ebenfalls. Im Englischen bezeichnet man sie als *concepts*, dabei werden sie im Deutschen oft mit dem Wort *Konzepte* übersetzt, wobei die Bezeichnungen *Begriff* und *Klasse* eindeutiger sind und im Kontext von Konzeptionen weniger Verwirrung hervorrufen. Daher wird im Folgenden hauptsächlich von *Begriffen* gesprochen, wenn die

Begriffsdefinitionen in einer formalen Ontologie gemeint sind und von *Klassen*, wenn die Ableitung von konkreten Instanzen im Fokus steht. Man meint damit schlicht einen abstrakten Begriff, von dem sich sprachliche Konkretisierungen ableiten lassen. Solche *Klassen* können hierarchisch strukturiert werden, so dass eine Vererbung möglich wird und das Wort *Klasse* einem Informatiker am direktesten die Bedeutung vermitteln kann. Im Gegensatz zu Klassen in einer objektorientierten Programmiersprache ist einem *Konzept* in einer Ontologie nicht klar zugeordnet, auf welche Weise eine Instanz davon konkretisiert werden kann. Statt festgelegter Klassenvariablen kann eine Instanz oder ein Individuum durch Relationen zu anderen Instanzen, Klassen oder Attributen näher beschrieben werden. Als Beispiel sind in Abbildung 2.6 für den Kontext der Softwareentwicklung die einfachen Begriffe *Mensch* und *GUI-Element* zunächst einmal ohne weitere Anreicherungen gegeben. Um Unterschiede in der Bedeutung klar hervorzuheben sind Begriffe, von denen Konkretisierungen abgeleitet werden können als Ellipsen mit durchgezogenem Rand dargestellt.

Abbildung 2.6.: Zwei Begriffe beziehungsweise Konzepte

Relationen

Beziehungen zwischen nicht hierarchischen Konzepten werden in einer Ontologie durch Relationen repräsentiert. Diese zeichnen sich dadurch aus, dass sie textuell gesehen einfache Tripel der Form Subjekt, Prädikat, Objekt sind. Ein Beispiel dafür ist in Abbildung 2.7 für den allgemeinen Fall der Betätigung einer Systemschnittstelle durch einen Menschen veranschaulicht. Allgemeine Beziehungen werden in diesem Abschnitt mittels durchgezogener Pfeile dargestellt.

Abbildung 2.7.: Zwei Begriffe, die über eine Relation in Beziehung gesetzt sind

Attribute

Mit Attributen können Ableitungen von Konzepten mit Informationen angereichert und dadurch weiter charakterisiert werden. Solche Charakterisierungen können Beziehungen zu Daten verschiedener Arte sein, beispielsweise Strings oder Zahlen. Diese werden ebenfalls als Subjekt, Prädikat, Objekt-Tripel formuliert. Man bindet die Möglichkeit der Attributierung an den jeweiligen Begriff. Das allgemeine Attribut beschreibt die Art seiner Ausprägung und kann dann bei einer Instanziierung mit einem konkreten Wert versehen werden, welcher beispielsweise ein Text oder eine Zahl sein kann.

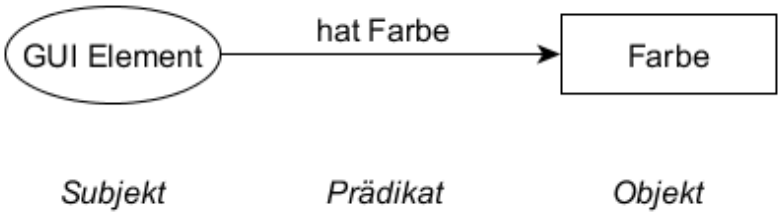


Abbildung 2.8.: Ein Begriff, dem über eine Relation ein Attribut zugeordnet ist

Instanzen

Instanzen einer Ontologie sind konkrete Ausprägungen von enthaltenen Konzepten. Um die bereits veranschaulichten Beispiele aufzugreifen, ist in Abbildung 2.10 dargestellt, wie aus den Begriffen und Relationen eine konkrete Instanz abgeleitet werden kann. Dabei ist die Konkretisierung des Begriffes *Mensch* der *Benutzer* und die von *GUI-Element* der *Button*. Die Relation dazwischen ist ebenfalls konkretisiert, aus "betätigt" wurde "drückt". Der Pfeil ist als Konsequenz daraus, dass eine Instanziierung einer allgemeinen Beziehung darzustellen ist auch gestrichelt.

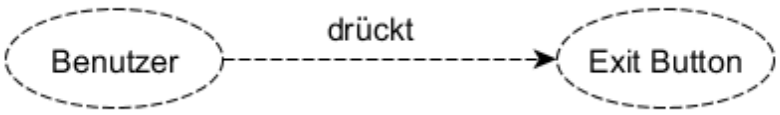


Abbildung 2.9.: Eine konkrete Instanz der Ontologie aus Abbildung 2.7

Um einen Zusammenhang zwischen Ontologiebegriffen und der Instanz sind hier für die Ableitungsbeziehung gestrichelte Pfeile mit nicht ausgefüllter Pfeilspitze gewählt worden. Abbildung 2.10 zeigt das komplette Beispiel. Auch die optische Hervorhebung durch zwei Kästen dient hier nur der Verdeutlichung des Unterschiedes zwischen Ontologie und deren Instanz.

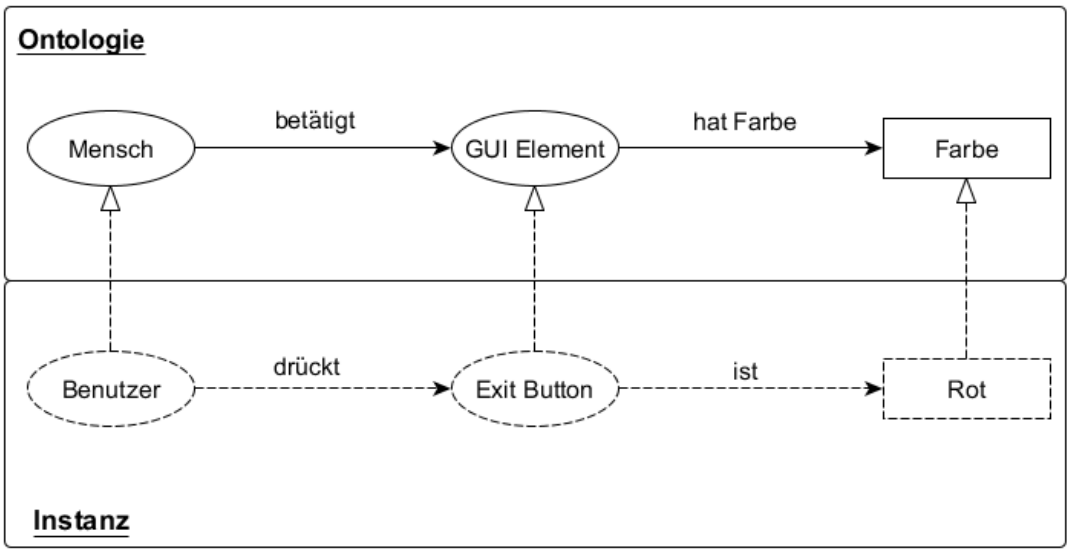


Abbildung 2.10.: Eine konkrete Instanz abgeleitet von einer Ontologie

Axiome

Axiome innerhalb einer Ontologie sind Regeln, die ohne Beweis im Kontext der modellierten Domäne immer gelten. Der oder die Ersteller der Ontologie legen also darin implizit fest, welche Ableitungen von Tripeln zulässig sind. Als Beispiel kann man im Kontext des Software Engineerings die Sätze “Ein Benutzer ist eine menschliche Person, die eine Systemkomponenten betätigt“und “Eine Systemkomponenten ist ein Teil des Systems, der von Menschen betätigt werden kann“ ansehen.

2.2.2. Arten von Ontologien

Ontologien können zu verschiedenen Zwecken erstellt und untereinander in Beziehung gesetzt werden [18]. Bevor man eine solche Ontologie-Landschaft implementiert sollte man also die unterschiedlichen Aufgaben definieren. Guarino [18] schlägt vier generelle Arten von Ontologien vor. Diese unterscheiden sich bezüglich der Generalität der Begriffe, die durch sie ausgedrückt wird:

Top-Level Ontologien beschreiben sehr generelle Begriffe, wie Raum, Zeit, Materie, Objekt, Ereignis, Aktion etc., welche unabhängig von einem bestimmten Problem oder Domäne sind.

Domänen-Ontologien (*Domain Ontologies*) beschreiben ein Vokabular bezüglich einer spezifischen Domäne, wie zum Beispiel Medizin oder dem Requirements

Engineering. Sie konkretisieren dabei Begriffe und Relationen aus einer Top-Level Ontologie.

Aufgaben-Ontologien (*Task Ontologies*) beschreiben ein Vokabular bezüglich einer spezifischen Aufgabe oder bestimmter Aktivitäten, wie beispielsweise das Diagnostizieren in der Medizin oder einen Workshop in der Elicitation des Requirements Engineerings durchzuführen. Auch Aufgaben-Ontologien spezifizieren Begriffe und Beziehungen aus der Top-Level Ontologie.

Anwendungs-Ontologien (*Application Ontologies*) beschreiben Begriffe, die jeweils von einer Domäne und einer Aufgabe abhängen und sehr oft eine Konkretisierung von je einer Domänen- und einer Aufgaben-Ontologie sind. Die Begriffe korrespondieren dabei häufig mit den Rollen der Entitäten einer Domäne, während sie eine bestimmte Aktivität ausüben. Ein Beispiel dafür ist die Rollenverteilung und die damit verbundenen Aufgaben von Beteiligten eines Workshops im Requirements Engineering.

Dies Art und Weise, wie die vier Typen von Ontologien hierarchisch strukturiert sind und voneinander abhängen ist in Abbildung 2.11 illustriert.

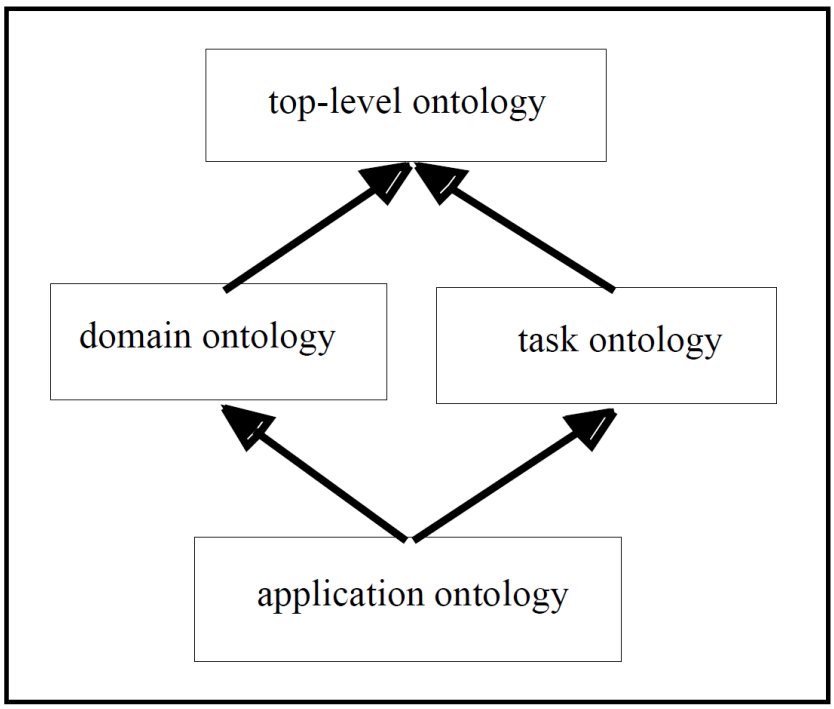


Abbildung 2.11.: Die Ebenen und Arten von Ontologien nach Guarino [18]

2.2.3. Anwendungsgebiete

Ontologien sind prinzipiell für nahezu alle Aufgaben geeignet, bei denen Wissen formal spezifiziert und gesammelt werden soll. Trotzdem sollten gewisse Gründe für die

Entwicklung einer Ontologie existieren, Uschold und Grüninger [33] identifizierten drei Hauptkategorien von Motivationen für die Anwendung und Entwicklung von Ontologien (Abb. 2.12).

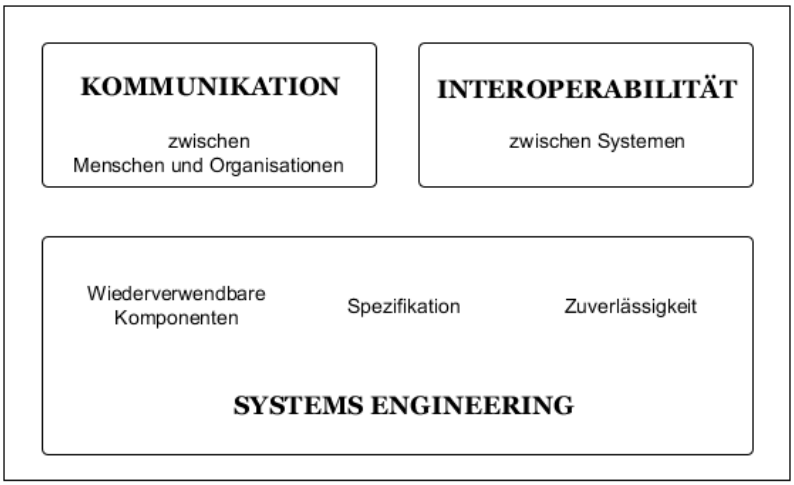


Abbildung 2.12.: Die Hauptkategorien für Anwendungen von Ontologien [33]

Kommunikation: Ontologien reduzieren Konfusion im Hinblick auf Begrifflichkeiten dadurch, dass sie eine Vereinheitlichung für deren Gebrauch innerhalb einer Organisation bieten. Daher eignen sie sich sehr gut für das Teilen von Wissen zwischen Menschen mit verschiedenen Blickwinkeln und Bedürfnissen. Mit Ontologien können normative, erweiter- und verfeinerbare Modelle entstehen, die eine semantische Transformation zwischen verschiedenen Kontexten zulassen. Ebenfalls sind Ontologien dann gebräuchlich, wenn eine Vernetzung zugrunde liegt, so dass man im Auge behalten kann, was miteinander verlinkt ist. Diese Eigenschaften sind für die Kommunikation grundsätzlich förderlich, genauso wie die Konsistenz und Eindeutigkeit von formalen Ontologien, sowie der Möglichkeit, verschiedene Perspektiven von Benutzern zu integrieren.

Interoperabilität: Wenn verschiedene Anwender Daten auszutauschen haben und dabei unterschiedliche Software-Tools verwenden ist dies ebenfalls ein Grund, eine Ontologie dazu zu benutzen. Gerade zwischen Werkzeugen, von denen es eine Vielfalt an Punktlösungen innerhalb von bestimmten Domänen gibt, sind wenig konsistent in ihrer Zusammenarbeit. Ontologien können als Zwischensprache implementiert werden und so zwischen formalen Sprachen als Übersetzungsschicht dienen. Dies ist vor allem zwischen verschiedenen Domänensprachen sinnvoll, da diese meist extrem spezialisiert sind. Beispiele für Dimensionen von Interoperabilität sind interne und externe Interoperabilität innerhalb von Organisationen und deren Abteilungen sein, oder aber eine formale Vermittlung zwischen verschiedenen Domänen oder Werkzeugen.

Systems Engineering: Beim Entwurf und der Entwicklung von Software Systemen können Ontologien ebenfalls eine wichtige Rolle spielen. Hier liegt der Fokus auf der Spezifikation von Elementen, der Zuverlässigkeit dieser Spezifikation und der Wiederverwendbarkeit von Komponenten. Als einen eher informellen Ansatz zur Spezifikation kann man Ontologien dazu verwenden, die Anforderungen an ein System zu identifizieren, sowie Beziehungen zwischen Systemkomponenten zu verstehen und diese dann als allgemeine Basis zu definieren. Dies ist besonders dann wichtig, wenn an der Systementwicklung mehrere verteilte Teams von Entwicklern mit verschiedenen Aufgabenbereichen beteiligt sind. Ein formaler Ansatz kann auch eine Ontologie zur deklarativen Spezifikation eines Systems sein, anhand derer man überlegen kann, wofür ein System entworfen ist und nicht, wie es die geforderte Funktionalität bereit stellt. Was Zuverlässigkeit angeht, so kann diese durch den Einsatz von Ontologien dahingehend helfen, dass eine Ontologie als Basis für die manuelle Prüfung von Entwurf gegen die Spezifikation dient. Um zusätzlich effektiv zu sein, muss eine Ontologie auch grundsätzlich Wiederverwendbarkeit unterstützen. Damit ist gemeint, dass man bei der Erstellung einer Ontologie bereits definieren kann, welche Teile daraus von verschiedenen Domänen, Aufgaben und Anwendungen wiederverwendet werden sollen und welche tatsächlich rein spezifisch sind.

Uschold und Grüninger [33] führten eine “skeletthafte Methode zur Erstellung einer Ontologie“ ein. Diese Empfehlung beinhaltet vier wesentliche Aktivitäten, die vor und bei einer Ontologieentwicklung ausgeübt werden sollten:

- Identifiziere Zweck und Tragweite der Ontologie
- Erstelle die Ontologie
- Evaluiere die Ontologie
- Dokumentiere die Ontologie

2.2.4. Resource Description Framework (RDF)

Das **Resource Description Framework (RDF)** ist ein Standardmodell für den Datenaustausch im Web [8]. Es verfügt über Funktionen, die das Zusammenführen von Daten erleichtern, selbst wenn sich die zugrunde liegenden Schemata voneinander unterscheiden. Es unterstützt insbesondere die Entwicklung von Schemata im Zeitablauf, ohne dass alle Datenkonsumenten geändert werden müssen. RDF erweitert die Verlinkungsstruktur des Webs, indem es URIs verwendet um die Beziehung zwischen den Dingen sowie die beiden Enden des Links zu benennen. Diese Form wird üblicherweise als “Triple“ bezeichnet. Mit diesem einfachen Modell können strukturierte und semi-strukturierte Daten gemischt, exponiert und über verschiedene Anwendungen verteilt werden. Die Verknüpfungsstruktur bildet einen gerichteten und beschrifteten Graphen, bei dem die Kanten die benannte Verbindung zwischen

zwei Ressourcen, also Graph-Knoten darstellen. Diese grafische Darstellung ist das einfachste Modell für RDF und wird oft in leicht verständlichen visuellen Erklärungen verwendet.

2.2.5. Web Ontology Language (OWL)

Das Ziel der *Semantic Web Initative* [5] ist es laut Knublauch [22] eine auf formalen Domänen-Modellen beruhende offene Infrastruktur für Web-Services und intelligente Agenten zur Verfügung zu stellen. Die **Web Ontology Language (OWL)** stellt in ihrer Spezifikation [2] drei unterschiedlich ausdrucksstarke Untersprachen zur Verfügung, die für den Einsatz in unterschiedlichen Anwendungsgebieten entwickelt wurden:

- OWL Lite
- OWL DL
- OWL Full

OWL Lite richtet sich an Anwender, die vornehmlich eine Hierarchie von Klassifikationen und einfache Einschränkungen benötigen. Dabei sind Einschränkungen bei Kardinalitäten möglich, allerdings nur Kardinalitätswerte von 0 oder 1. OWL Lite ist absichtlich einfacher gehalten als seine ausdrucksstärkeren Verwandten, bietet einen schnellen Migrationspfad für Thesauri und andere Taxonomien und hat eine deutlich geringere formale Komplexität als OWL DL und OWL Full.

OWL DL eignet sich bei Anwendungen, für die eine maximale Aussagekraft unter Beibehaltung der Vollständigkeit der Berechnungen und Entscheidbarkeit verlangt ist. Dabei ist garantiert, dass alle Schlussfolgerungen berechenbar sind und alle Berechnungen in endlicher Zeit enden. Die Untersprache enthält alle OWL-Sprachkonstrukte, kann aber nur unter bestimmten Einschränkungen verwendet werden. Beispielsweise kann eine Klasse eine Unterklasse vieler Klassen sein, eine Klasse darf aber keine Instanz einer anderen Klasse sein.

OWL Full ist für Anwendungsfälle gedacht, bei denen die maximale Ausdruckskraft und die syntaktische Freiheit von RDF ohne rechnerische Garantien benötigt wird. In OWL Full kann eine Klasse beispielsweise gleichzeitig als eine Ansammlung von Individuen und als eigenständige Person behandelt werden. OWL Full erlaubt es einer Ontologie, die Bedeutung des vordefinierten (RDF oder OWL) Wortschatzes zu erweitern. Es gilt als unwahrscheinlich, dass irgendeine Logiksoftware in naher Zukunft in der Lage sein wird, vollständige Logik für jede Funktion von OWL Full zu unterstützen.

Jede dieser Teilsprachen ist eine Erweiterung ihres einfacheren Vorgängers sowohl in dem, was legal ausgedrückt werden kann, als auch in dem, was valide

abgeschlossen werden kann. Es bestehen innerhalb der OWL Untersprachen die folgenden Beziehungen, ihre Umkehrung allerdings nicht. Eine Veranschaulichung dieser Hierarchie ist in Abbildung 2.13 zu sehen.

- Jede legale OWL Lite Ontologie ist eine legale OWL DL Ontologie
- Jede legale OWL DL Ontologie ist eine legale OWL Full Ontologie
- Jede gültige OWL Lite-Schlussfolgerung ist eine gültige OWL DL-Schlussfolgerung
- Jede gültige OWL DL-Schlussfolgerung ist eine gültige OWL Full-Schlussfolgerung

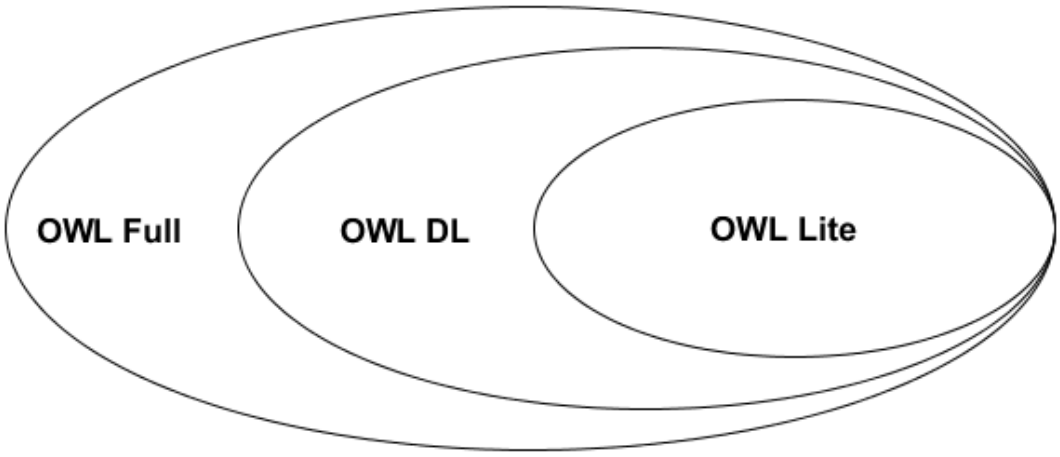


Abbildung 2.13.: Eine hierarchische Veranschaulichung der OWL-Untersprachen

Ontologieentwickler, die OWL einsetzen, sollten sich überlegen, welche Subsprache ihren Bedürfnissen am besten entspricht. Die Wahl zwischen OWL Lite und OWL DL hängt davon ab, inwieweit der Anwender die von OWL DL gebotenen ausdrucksstärkeren Konstrukte benötigt. Die Wahl zwischen OWL DL und OWL Full hängt hauptsächlich davon ab, inwieweit Benutzer die Metamodellierungsfunktionen des RDF-Schemas benötigen (z. B. das Definieren von Klassenklassen oder das Anhängen von Eigenschaften an Klassen). Bei der Verwendung von OWL Full im Vergleich zu OWL DL ist die Argumentations-Unterstützung weniger vorhersehbar, da vollständige OWL Full-Implementierungen derzeit nicht existieren.

OWL Full kann als Erweiterung von RDF angesehen werden, während OWL Lite und OWL DL als Erweiterungen einer eingeschränkten Ansicht von RDF angesehen werden können. Jedes OWL-Dokument (Lite, DL, Full) ist ein RDF-Dokument, und jedes RDF-Dokument ist ein OWL Full-Dokument, aber nur einige RDF-Dokumente sind ein legales OWL Lite oder OWL DL-Dokument. Aus diesem Grund muss bei

der Migration eines RDF-Dokuments auf OWL etwas Vorsicht walten. Wenn die Aussagekraft von OWL DL oder OWL Lite als angemessen erachtet wird, müssen einige Vorsichtsmaßnahmen getroffen werden, um sicherzustellen, dass das Original-RDF-Dokument den zusätzlichen Auflagen von OWL DL und OWL Lite entspricht. Unter anderem muss jeder URI, der als Klassenname verwendet wird, explizit als Typ Eule erklärt werden: Klasse (und ähnlich für Eigenschaften), jeder Einzelne muss mindestens einer Klasse angehören (auch wenn nur Eule: Sache), die für Klassen, Eigenschaften und Individuen verwendeten URIs müssen sich gegenseitig disjunkt sein.

Als allgemeinen Einstiegspunkt für Begrifflichkeiten leitet OWL jeglichen Konzeptbegriff zunächst einmal vom Grundbegriff *owl:Thing* ab, jeder nicht zuzuordnende Begriff erweitert automatisch vom Gegenpart *owl:Nothing* ab.

3. Konzept zur Umsetzung

In diesem Kapitel wird das erarbeitete theoretische Konzept zur Umsetzung einer Annotation von Videoinhalten beschrieben. Aus den zu Anfang vom Auftraggeber (*hier:* Betreuer und Erstprüfer) genannten Rohanforderungen waren zunächst konkrete Informationen zu extrahieren.

3.1. Anforderungen an die Videoannotation auf Basis von Ontologien

Die groben Anforderungen für die prototypische Umsetzung ergaben sich einerseits aus der Aufgabenstellung der Arbeit, andererseits mussten sie noch im Laufe der Entwicklung erarbeitet werden. Aus Gesprächen mit den Stakeholdern dieser Arbeit konnten die folgenden zwei grundlegenden Rohanforderungen entnommen werden.

[**R01**] In Vision Videos dargestellte Instanzen von Konzepten und deren Beziehungen sollen auf Basis einer Ontologie grafisch annotiert werden können.

[**R02**] Die Ontologie soll in der *Web Ontology Language (OWL)* spezifiziert werden.

Es galt nun ein Konzept für ein Software Werkzeug zu entwickeln, welches diese beiden Kernanforderungen sinnvoll erfüllt. Die Anforderung [R02] ist bereits eine unmissverständlich formulierte Bedingung an die Implementierung einer Ontologie, weshalb diesbezüglich keine Interpretationsmöglichkeiten oder Fragen entstanden. Um eine dem verbreitetsten Standard entsprechende Maschinenlesbarkeit zu garantieren, musste zwingend die Web Ontology Language (OWL) aus Abschnitt 2.2.5 verwendet werden.

Da die andere erhobene Anforderungen [**R01**] noch nicht konkret genug für eine Umsetzung war, musste diese zunächst in genauere Anforderungen aufgespalten werden. Um dies zu erreichen, wurde Fragen zu den verschiedenen Aspekten aus [**R01**] formuliert, die Formulierung von [**R01**] musste dazu aufgespalten werden:

In Vision Videos dargestellte (Frage 1.1)

Auf welche Art können Videoinhalte dargestellt werden?

Instanzen von Konzepten und deren Beziehungen (Frage 1.2)

Welche Konzepte und Beziehungen sind genau gemeint?

sollen

auf Basis einer Ontologie (Frage 1.3)

Wie genau hat der Bezug zu einer Ontologie auszusehen?

grafisch annotiert (Frage 1.4)

Wie soll die grafische Annotation genau aussehen und angebracht werden?

werden können.

Der Anforderungssatz enthält unterschiedliche inhaltliche Aspekte oder Eigenschaften, so dass man die Fragen in verschiedene Bezugsbereiche aufteilen konnte. Die folgenden Bereichseinteilungen gelten jeweils für Konzepte, die aus dargestellten Entitäten und deren Beziehungen untereinander bestehen.

Frage 1.1 betrifft die wahrnehmbaren Eigenschaften von Videoinhalten, die annotiert werden sollen.

Frage 1.2 betrifft die Eigenschaft der dargestellten Inhalte, einer übergeordneten Klassifzierung zugeordnet werden zu können

Frage 1.3 betrifft die Beschaffenheit einer übergeordneten Klassifizierung durch eine Ontologie

Frage 1.4 Betrifft die Eigenschaften, die eine grafische Hervorhebung im Video besitzen soll

Die identifizierten Eigenschaften der unterschiedlichen Aspekte musste nun weiter verfeinert werden. Zunächst hat man verschiedene beispielhafte Videos angesehen wurden und sich schlicht dabei vorgestellt, man würde mit der Maus Markierungen um Personen, Systemkomponenten und deren Interaktionen malen. Daraufhin wurden einzelne Bilder aus den Videos in einem einfachen Zeichenprogramm geladen und daran ausprobiert, welche Arten von grafischer Annotation infrage kamen. Dabei konnten einige detaillierte Antworten auf die Fragen 1.1 bis 1.4 gefunden werden. Die Antworten wurden so formuliert, dass sie wiederum Aussagen über möglichst atomare Aspekte ergeben. Sie sind im Folgenden aufgelistet.

Frage 1.1 – Wahrnehmbare Eigenschaften von Videoinhalten

(a) Videoinhalte haben einen Sichtbarkeitsbereich innerhalb eines Videobildes bzw. -frames, der visuell hervorgehoben werden kann.

(b) Videoinhalte haben eine zeitliche Dauer der Sichtbarkeit, d.h. über mehrere Einzelbilder oder -frames, während der sie visuell hervorgehoben werden können.

(c) Videoinhalte können hörbar sein.

(d) Videoinhalte können aus einer Kombination aus Sicht- und Hörbarkeit bestehen.

(e) Videoinhalte können zu anderen Videoinhalten in sichtbarer Beziehung stehen.

(f) Videoinhalte können zu anderen Videoinhalten in hörbarer Beziehung stehen.

(g) Videoinhalte können zueinander in Beziehung stehen, ohne dass man diese Beziehung sehen oder hören kann.

Frage 1.2 – Eigenschaften zur Klassifizierung

(a) Dargestellte Entitäten können Personen sein.

(b) Dargestellte Entitäten können Hardware-Systemkomponenten sein.

(c) Dargestellte Entitäten können Software-Systemkomponenten sein.

(d) Dargestellte Beziehungen zwischen Entitäten können Mensch-Maschine-Interaktionen sein.

(e) Dargestellte Beziehungen zwischen Entitäten können Mensch-Software-Interaktionen sein.

(f) Dargestellte Beziehungen zwischen Entitäten können Hardware-Software-Interaktionen sein.

Frage 1.3 – Ontologie-Basis

(a) Annotierte Entitäten sollen von Begriffsklassen einer Ontologie-Basis abgeleitet werden können.

(b) Annotierte Beziehungen zwischen annotierten Entitäten sollen von Beziehungsklassen einer Ontologie-Basis abgeleitet werden können.

Frage 1.4 Grafische Annotation

(a) Grafische Annotationen sollten den Entitäten von Konzepten visuell hervorheben können.

(b) Grafische Annotationen sollten den Bildbereich von Entitäten über den Zeitraum ihrer Sichtbarkeit visuell hervorheben können.

(c) Grafische Annotationen sollten Beziehungen zwischen Entitäten visuell hervorheben können.

(c) Grafische Annotationen sollten Beziehungen zwischen Entitäten über den Zeitraum ihrer Sichtbarkeit visuell hervorheben können.

Im Folgenden werden die erarbeiteten Antworten auf die Fragen **1.1**, **1.2** und **1.4** anhand von Beispielen aus Videos erklärt. Die Antworten zu Frage **1.3** werden textuell und mit Bezug zum Ontologie-Begriff aus den Grundlagen erläutert.

Die Antworten auf Frage **1.1** betreffen die Eigenschaften von Videoinhalten bezüglich ihrer Darstellung innerhalb des Videos. Hier gab es für eine grafische Annotation zu bedenken, welche Dimensionen der Videoinhalte genau von Interesse für die verschiedenen Arten der Benutzung einer Annotation sind. Zum einen soll ein Video annotiert, zum anderen ein bearbeitetes Video sinnvoll zur direkten Betrachtung von markierten Inhalten verwendet werden können. Um wirklich sinnvoll annotieren zu können, muss man sich überlegen, wozu man eigentlich Inhalte von Videos grafisch hervorheben und maschinenlesbar mit anderen Wissensbereichen verknüpfen möchte. Wenn man die Absicht hat, der Dokumentation ebenso behilflich zu sein, wie man es der Auflösung von Unklarheiten mit dem Kunden ist, dann ist folgendes zu bedenken.

Einzelne Entitäten zu annotieren, damit direkt einen Bezug zu einem Domänenbegriff herzustellen und eine konkrete Ausprägung zu markieren unterstützt primär eine Einordnung in eben jene Domäne. Für eine Dokumentation ist das zwar ebenfalls ein wenig hilfreich, allerdings ist es schwer vorstellbar, dass Entwickler nur für eine Beispielausprägung eines einzigen Begriffes extra ein Video anschauen und das womöglich noch mit einem extra Werkzeug. Bedenkt man aber die Form einer Interaktion, wie sie beispielsweise in den Schritten eines Use Case textuell beschrieben ist, so bringt eine Darstellung mit bewegten Bildern deutlich mehr Informationen. Bei tatsächlich unklarer Formulierung eines Schrittes innerhalb eines Use Cases kann eine beispielhafte Visualisierung mit Bezug zu den Domänenbegriffen wesentlich zum besseren Verständnis beitragen.

Eine Interaktion besteht im Wesentlichen aus zwei oder mehr Entitäten, die miteinander in aktionsartiger Beziehung stehen. Im Kontext der Softwareentwicklung und besonders im Bezug zu Use Cases wird eine Interaktion am häufigsten eine Beziehung zwischen einer menschlichen Entität – dem Benutzer eines Systems – und einer nicht

menschlichen – einer Systemkomponente – bestehen. Eine Systemkomponente kann dabei entweder eine Hardware- oder eine Softwarekomponente sein. Eigentlich ist eine direkte Interaktion von Mensch mit Software nur über zwischengeschaltete Hardware möglich, in einem Vision Video kann dies aber je nach Inhalt vernachlässigt werden. Als Beispiel sei ein Button einer Smartphone App gegeben. Das Betätigen des Buttons wird natürlich nur über eine Hardware Touch Screen möglich, diese Beziehungsbrücke betrifft allerdings unter Umständen weder die Kundendomäne, noch die Entwicklung der Software selbst.

Die Antworten auf Frage **1.3** betreffen Eigenschaften, die zur Klassifizierung von Entitäten und Relationen auf Ontologie-Basis gehören. Es müssen sowohl Entitäten, als auch Beziehungen von Oberklassen abgeleitet werden können. Beispielsweise kann die Betätigung einer Systemkomponente das Drücken eines Buttons sein oder sogar die bloße Bewegung vor einem Bewegungssensor. Daher galt es, für die Mensch-System-Komponenten eine geeignete Ontologie zu entwerfen, aus der diesbezügliche Inhalte weiter konkretisiert werden können und das möglichst Projekt-übergreifend und damit unabhängig von der Kundendomäne.

Die grafische Annotation betreffen konnte auf Frage **1.4** geantwortet werden, dass die wichtigen für eine Annotation zu berücksichtigen Eigenschaften Abbildungen der Eigenschaften von Inhalten sind. Annotationen sollten so lange und an der Stelle im Bild sichtbar sein, wie es der hervorzuhebende Inhalt ist. Dies gilt sowohl für Entitäten, als auch für Interaktion zwischen solchen.

3.2. Eigenschaften von Videoinhalten

Eine sehr entscheidende Eigenschaft von Videoinhalten ist ihre Sichtbarkeit. Diese besitzt im Hinblick auf eine grafische Annotation allerdings zwei Dimensionen, nämlich die Sichtbarkeit innerhalb eines einzelnen Videobildes oder -frames und die zeitliche Länge, in der sie zu sehen sind. Ebenfalls zu beachten ist es, ob eine Entität von ihrem ersten Auftreten im Video bis hin zu ihrem Verschwinden durchgehend zu sehen ist oder ob zwischendurch Bilder gezeigt werden, auf denen nur andere oder gar keine Entitäten abgebildet sind. Ein Video enthält prinzipiell immer auch eine Tonspur. So werden Szenen in Spielfilmen oft mit stimmungsvoller Musik untermalt oder ein nicht sichtbarer Erzähler spricht zum Zuschauer indem er die gezeigten Bilder näher erläutert oder kommentiert. Den Einsatz von Musik für ein Vision Video zuzulassen erhöht unter Umständen den Präsentationswert vor dem Kunden, kann aber eigentlich nur vom Inhalt ablenken, es sei denn, die Musik gehört klar zu einer sichtbaren Interaktion oder diese besteht sogar aus Musik. Gesprochene Erklärungen und Kommentare sind dagegen deutlich verständnisfördernder. Im Hinblick auf eine grafische Annotation aber sind sie problematisch, weil man ihre Bedeutung nicht einfach visuell hervorheben kann ohne zusätzliche Annotationsformen einzuführen und deren Bedeutung wiederum gesondert zu erklären.

3.3. Grafische Annotation im Bild

Mit grafischer Annotation ist laut Anforderungen durch den Betreuer eine sichtbare Markierung von Videoinhalten gemeint. Dazu muss der Bildbereich hervor gehoben werden, der Domänenbegriffe und -relationen enthält. Um ganz genau die Umrisse einzelner Entitäten zu markieren, wäre grundsätzlich eine automatische Erkennung von Bewegungen und Formen im Sinne der OpenCV-Technologie[1] wünschenswert. Da diese Genauigkeit aber nicht im hier behandelten Fokus liegt und zusätzlich den zeitlichen Rahmen für eine Masterbeit gesprengt hätte, entschied man sich für eine manuelle grafische Annotation. Diese läuft so ab, dass der Annoteur zunächst einmal das Video in das Werkzeug lädt und es anschaut. Hat er einen Inhalt identifiziert, der sich für eine Annotation eignet, so soll direkt auf dem Videobild "gemalt" werden. Um keine verwirrenden Markierungen verschiedenster Form zu erlauben, fiel die Wahl auf das Zeichnen von farbigen und semi-transparenten Rechtecken, da zwar eine grafische Hervorhebung gewünscht ist, diese aber natürlich nicht das zu annotierende Sub- oder Objekt verdecken soll.

3.4. Auflistung von Annotationen

Man benötigt für ein Annotationsprojekt eine Übersicht über alle enthaltenen und darin annotierten Entitäten und Interaktionen. Dies ist bedingt durch die Tatsache, dass ein Video nicht ein einzelnes Bild zeigt, sondern viele verschiedene Szenen mit annotationswürdigen Inhalten enthalten kann. Würde man ohne eine Übersicht Markierungen anbringen, so wäre der Mehrwert dieser im Sinne zielgerichteter Dokumentation der Inhalte und klar zuzuordnender Begriffe für den menschlichen Betrachter nicht mehr gegeben. Dieser müsste sich das Video wieder komplett ansehen und könnte ebenfalls nicht erkennen, welche Annotation eine konkrete Ausprägung eines Domänenbegriffes ist.

3.5. Sichtbarkeitszeit im Video

Für die Visualisierung von Entitäten und Interaktionen über die Zeit ihrer Sichtbarkeit ist grundsätzlich eine zweidimensionale Darstellung notwendig. Man muss die Entität abbilden und den Zeitbezug sichtbar machen. Um das zu gewährleisten, kamen einige Diagrammtypen infrage, Mit der Darstellung der zeitlichen Sichtbarkeit im Video als Linienediagramm ist es durchaus möglich zu bewerten, wie klar eine Interaktion im Video dargestellt ist. Aus verschiedenen Mustern, die sich aus der Annotation im Liniendiagramm ergeben können Schlüsse auf die Eignung der Darstellung für die Dokumentation und die Interpretierbarkeit des im Video Sichtbaren gezogen werden.

[1] `https://opencv.org`

3.6. Eignung von Videos für eine grafische Annotation

Da das erarbeitete Konzept noch relativ allgemein gehalten ist und sich auf die Annotation von Bildmaterial bezieht, gibt es nach wie vor Eigenschaften von Videoinhalten, die auf diese Weise noch problematisch zu annotieren sind. Welche Beschaffenheit Videos haben sollten um Interaktionsszenarien annotierbar darzustellen ist auch ein Themengebiet, welches in dieser Arbeit angesprochen werden muss. Es ist also eine prototypische Software zu entwickeln, mit der man Videoinhalte grafisch annotieren und diese als Instanzen einer Ontologie speichern kann. Die den Instanzen zugrunde liegende Ontologie ist ebenfalls zu spezifizieren und sie sollte auf die grafische Annotation von Mensch-System-Interaktionen zugeschnitten sein. Konsequenter Weise sollte sie Konzepte mit Bezug zur Kundendomäne beinhalten, von denen annotierte Entitäten und Beziehungen dann abgeleitet werden können.

Zu den Eigenschaften, die ein gut annotierbares Vision Video besitzen sollte zählen vor allem Inhalte, die man nicht sehen kann, zuallererst jeglicher Ton auf der Audiospur. Wird Essenzielles zum Verständnis der sichtbaren Inhalte per Audiokommentar erklärt, so bekommt der Betrachter ein Verständnisproblem, sobald Teile der hörbaren Erläuterung außerhalb der Sichtbarkeitszeit von annotierten Inhalten stattfindet, also davor oder danach.

Eine Interaktion ist nur dann sinnvoll annotierbar, wenn alle daran beteiligten Entitäten gleichzeitig im Bild sichtbar sind. Ein Beispiel für ein interessantes Video mit Visionscharakter, bei dem das nahezu nie der Fall ist, ist der *Knowledge Navigator* [13] von Apple aus dem Jahre 1988. Dort werden – mittlerweile Alltag gewordene – Interaktionen eines Menschen mit einer Art Tablet vorgeführt, bei der erstens die Interaktion über eine Spracheingabe funktioniert und zweitens die beteiligten Entitäten Mensch und Tablet praktisch nie gleichzeitig zu sehen sind. Obwohl dieses Video die Vision des zukünftigen System sehr anschaulich und verständlich darstellt, ist es für eine grafische Annotation von Interaktionen eher ungeeignet. Die Interaktionen sind nur zu hören und man kann diese wegen des Bildschnittes auch nicht klar zwei Entitäten grafisch zuordnen. Ein für die Verständnisklärung mit dem Kunden geeignetes Vision Video ist also nicht zwangsläufig auch für eine grafische Annotation im Sinne der maschinenlesbaren Dokumentation geeignet.

Zu implementieren war demnach eine Software, die Videos zunächst wie ein herkömmlicher Player abspielen kann. Zusätzlich sollte die Möglichkeit zur einfachen Annotation von Inhalten gegeben sein, wobei man bei Entitäten und Interaktionen grafisch hervorheben können sollte. Die Markierungen hatten benannt werden zu können und ebenfalls von einem Begriff aus einer zugrunde liegenden Ontologie abgeleitet. Jede grafische Hervorhebung sollte genauso lange zu sehen sein, wie der hervorgehobene Inhalt es im Videobild ist. Dazu war eine Möglichkeit zur Bestimmung der Sichtbarkeitszeit notwendig. Die Erscheinung der Annotation selbst sollte ebenfalls editierbar in Größe und Farbe sein um unabhängig von der Farbgebung des Videos

dessen Inhalte wirklich hervorheben zu können. Bevor im nächsten Kapitel auf die praktische Umsetzung der dieser Ergebnisse eingegangen wird, werden im letzten Abschnitt dieses Kapitels kurz einige verwandte Arbeiten vorgestellt und Unterschiede, sowie Gemeinsamkeiten mit dieser Arbeit erklärt.

3.7. Verwandte Arbeiten

Es existieren einige Arbeiten, die eine Hervorhebung von Videoinhalten auf Ontologie-Basis fokussieren. Diese adressieren allerdings im Gegensatz zu dieser Arbeit hauptsächlich eine automatische Annotation von Inhalten aus Videos zur späteren Analyse und Ergänzung einer Ontologie um ein bestehendes Domänenwissen zu erweitern. Bloehdorn et al. [6] nutzen ein automatisches Annotationswerkzeug um sehr detaillierte Umrisse von Entitäten oder Formen und weitere Charakteristika von sichtbaren Inhalten zu annotieren und damit eine bestehende Ontologie weiter mit Wissen anzureichern. Das Ziel ist es dabei nicht, die Softwareentwicklung zu unterstützen, sondern mehr Wissen über Domänen maschinenlesbar und -analysierbar zu machen. Als Beispiel nennen sie das Wissen über den Tennissport, wobei Details wie Schläger, Bewegungen der Spieler und ähnlich spezifische Aspekte der Domäne zu annotieren sind.

Bai et al. [3] behandeln in ihrer Arbeit eine Ontologie-basierte, automatische Videoanalyse. Dabei analysiert eine automatische Objekt-, Sequenz und Ereigniserkennung auf Ontologie-basis mit gegebenem Domänenwissen eingegebene Videos. Sie charakterisieren dabei als Beispiel Aufnahmen von Fussballspielen anhand von automatisch erkannten Farben, Formen und Bewegungsmustern.

Jeong et al [19] hingegen nutzen Ontologien zur semantischen Annotation von Videos im Bereich von SmartTVs. Die Annotation ist dabei ebenfalls automatisch und hat die Absicht, in Videos enthaltene Sub- und Objekte anhand ihrer Umrisse in Einzelbildern oder Szenen zu markieren. Aufgrund von erkannten, annotierten und über Ontologien geteilten Ähnlichkeiten von Inhalten unterschiedlicher Filme ist beabsichtigt SmartTVs Vorschläge von Filmen für Nutzer zu präsentieren.

Die genannten, verwandten Arbeiten verwenden Ontologien als Wissensspeicher und annotieren Inhalte von Videos. Letzteres tun sie allerdings automatisch und geben einem Anwender ihrer Software nicht die manuelle Möglichkeit zur Annotation. Die Zielsetzung der Annotationen auf Ontologie-Basis ist ebenfalls eine andere, da es nicht um den Zugriff auf ganz spezielle markierte Inhalte geht, sondern um ein aggregiertes Wissen und darauf basierende Erweiterung von bestehenden Daten. Die Softwareentwicklung selbst ist zwar Teil der erwähnten Arbeiten, allerdings nur in der Anwendung zum Aufbau von Implementierungen. Ihre Verbesserung ist in keiner Arbeit das Ziel. Aus diesen Gründen sind die Arbeiten verwandt, behandeln aber nicht dasselbe Thema wie diese.

4. Prototypische Umsetzung

Die Erarbeitung der prototypischen Umsetzung greift die Aspekte, die im vorangegangenen Kapitel 3 entwickelt wurden auf. Anhand der zu implementierenden Konzeptteile folgt in diesem Kapitel die Beschreibung der Implementierung. Die Anforderungen an das zu entwickelnde Werkzeug war es, eine Möglichkeit zur grafischen Annotation für Videoinhalte bieten. Ebenfalls sollten annotierte Inhalte auf Basis einer Ontologie in eine darüber gegebene Domäne eingeordnet werden können.

Für die Unterstützung der grafischen Annotation in Videos auf Basis von Ontologien wurde für diese Arbeit das System *ObViViAn*[1] entwickelt. Mit diesem Werkzeug ist man zunächst einmal in der Lage Videos einfach abzuspielen. Darüber hinaus aber können sichtbare Inhalte direkt auf dem Bild annotiert werden. Für markierte Entitäten und Interaktionen zwischen solchen kann daraufhin festgelegt werden, von welchen Konzepten der zugrunde liegenden Ontologie sie eine Instanziierung darstellen. In Abbildung 4.1 ist zu sehen, wie die Gesamtansicht für das Annotieren von Videos und das abspielen von annotierten Videos umgesetzt wurde. Die einzelnen Teilansichten und ihre Funktionsweisen werden in diesem Kapitel begründet erläutert.

Für einen Lösungsansatz wird dazu grundsätzlich angenommen, dass die Videos allen Entwicklern eines Projektes als Dateien zur Verfügung stehen, zum Beispiel auf einem zentralen Server. In der Praxis würde sich auch eine zentrale Ontologie-Datenbank eignen, auf eine solche wurde hier aber aus Gründen der Komplexität ihrer Erzeugung und Bereitstellung verzichtet. Stattdessen benutzt das Annotationswerkzeug für jedes Projekt ein lokales Repository. Dies dient nur der Verringerung von Komplexität und stellt für die Maschinenlesbarkeit und Möglichkeit der eindeutigen Einordnung von Videoinhalten keine Einschränkung dar.

[1] Der Name steht für **O**ntology-**b**ased **Vi**sion **Vi**deo **An**notation tool

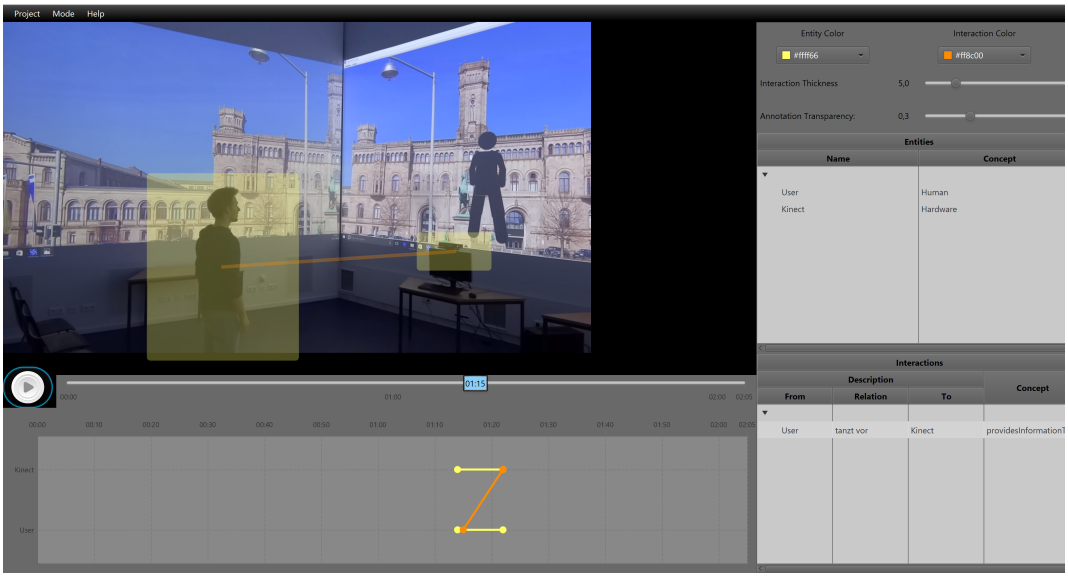


Abbildung 4.1.: Die Hauptansicht zum Editieren und Abspielen von Videos

Um die grafischen Annotation dem unterliegenden Video anpassen zu können wurden Einstellungsmöglichkeiten für die visuelle Darstellung der Annotationen implementiert. Es erwies sich als sinnvoll, die Transparenz und Farbe der Annotationen verändern zu können. In sehr dunklen Videos ist eine hellere Farbe wesentlich besser zu erkennen und in Videos mit einem eher hellen Bild eine dunkle Farbe. Des Weiteren ist es sinnvoll auch die Transparenz verändern zu können, da mit einer festgelegten Transparenz die Wahrscheinlichkeit steigt, dass die Farbabmischung Entitäten zu sehr verdeckt und diese nur noch schemenhaft zu erkennen wären. Jeweils unterschiedliche Farben für die Annotation von Entitäten und Interaktionen zu verwenden ist für die visuelle Hervorhebung ebenfalls sehr zweckdienlich, da so direkt zu erkennen ist, zwischen welchen Entitäten eine Interaktion stattfindet. Zusätzlich ist die Dicke der Linie, mit der Interaktionen dargestellt werden variierbar um abhängig vom Video eine deutliche Sichtbarkeit zu garantieren.

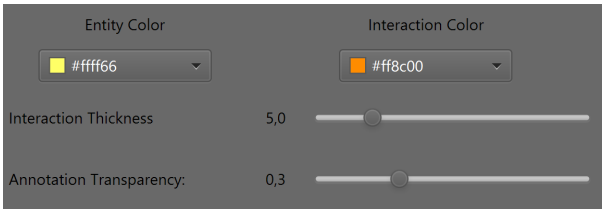


Abbildung 4.2.: Einstellungsmöglichkeiten zur Manipulation von Annotationen

4.1. Designentscheidungen und Features

Um ein nicht zu komplexes Werkzeug für die Annotation von Videos auf Basis von Ontologien zu erstellen, wurde von vorn herein darauf verzichtet, einen Editor für eine Ontologie einzubauen. Da die Bearbeitung ohnehin eine bestehende Wissensbasis über das Projekt voraussetzt, ist das Fehlen eines Ontologie-Editors nicht als Mangel für diese Arbeit zu sehen. Die Kernanforderungen an das Werkzeug bestanden lediglich aus der grafischen Annotation von Sub- und Objekten innerhalb des Videos. Die Interaktion ebenfalls grafisch hervorzuheben und mit der zugrundeliegenden Ontologie verknüpfen zu können ergab sich aus den Möglichkeiten der verwendeten Technologien und aus der Tatsache, dass Ontologien nicht nur bloße Aufzählungen von Sub- und Objekten sind, sondern deutlich mehr zu bieten haben. Eine rudimentäre Basis-Ontologie wurde erstellt um grundsätzlich das Prinzip der Mensch-Maschine-Interaktion grafisch annotieren zu können. Dazu wurde die beispielhafte Ontologie aus Kapitel 2 um die zu speichernden x- und y-Werte der Position von Annotationen erweitert. Jede annotierte Entität hat somit eine speicherbare Position innerhalb des Videos und eine Dauer ihrer Sichtbarkeit.

Es wurde ebenfalls darauf verzichtet, das Video in einzelne Bilder zu unterteilen und diese zu annotieren. Der Vorgang der Annotation wäre dadurch zwar wesentlich genauer bezüglich des Vorkommens von Entitäten und Interaktionen geworden, allerdings auch deutlich komplizierter und somit weniger geeignet für ein Annotieren zusammen mit dem Kunden. Um eine Video-Annotation so einfach wie möglich zu gestalten, setzte man sich das Ziel mit der Maus direkt auf dem Videobild zu annotieren. Für die zeitliche Einstellung der Annotationen musste daher eine gesonderte Konfigurations- und Darstellungsmöglichkeit geschaffen werden. Dabei wurde schnell klar, dass man eine Diagrammart dazu verwenden kann um zeitliche Begrenzungen auf der x-Achse von Objekten auf der y-Achse zu visualisieren.

4.1.1. Grafische Annotation im Bild

Die farbliche Hervorhebung von Videoinhalten ist so implementiert, dass der Benutzer von ObViViAn während des laufenden oder pausierten Videos auf dem zu sehenden Bild eine Annotation anbringen kann. Um eine Entität hervorzuheben ist der linke Mausknopf gedrückt zu halten und dabei die Maus in eine Richtung zu bewegen. So entsteht ein Rechteck, dessen Größe über die Mausbewegung bestimmt werden kann. Da eine Interaktion prinzipiell aus zwei Entitäten besteht, welche in einer inhaltlichen Beziehung zueinander stehen, kann man in diesem Tool nur dann eine Interaktion einzeichnen, wenn die beiden Entitäten gleichzeitig im Bild zu sehen und bereits annotiert sind. Man erstellt sie, indem man auf der Ausgangs-Entität den rechten Mausknopf gedrückt hält, die Maus bis zur Ziel-Entität bewegt und innerhalb deren Markierung den Button wieder loslässt. Dadurch entsteht eine farbige Linie vom Rechteck um die Ausgangs-Entität zur Markierung der Ziel-Entität. Die Abbildung 4.3 zeigt die Schritte der Annotation einer Interaktion. Diese teilt sich auf in die Annotation

der ersten sichtbaren Entität, dann der zweiten und dem Ziehen der Beziehungslinie von Entität 1 zum Ziel.

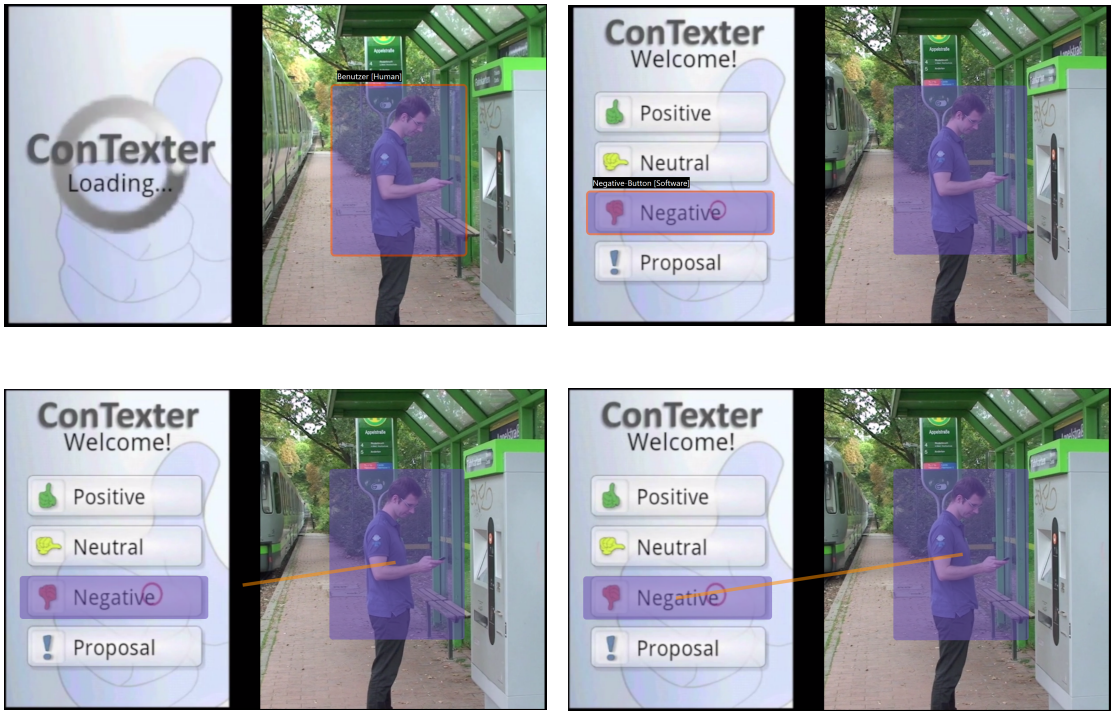


Abbildung 4.3.: Die Annotations-Ansicht von ObViViAn

4.1.2. Auflistung von Annotationen

Die Auflistung von annotierten Entitäten und Interaktionen zur Übersicht und Klassifizierung aus Ontologie-Begriffen ist in mit separaten Tabellen umgesetzt. In einer sind die Entitäten enthalten, in der unteren die Interaktionen. Sobald eine Annoation angebracht ist, kann sie in der Tabelle über das Feld in der Spalte *Name* benannt und über ein Dropdown-Menü in Spalte *Concept* einem Begriff aus der Ontologie zugeordnet werden. Ähnlich der tabellarischen Ansicht für Entitäten besteht die der Interaktionen ebenfalls aus einem Namen, der eingegeben werden kann und dem Dropdown-Menü für eine Ableitung einer Beziehungsklasse. Allerdings werden dort auch die beteiligten Entitäten aufgelistet, da sie essenzieller Bestandteil einer Interaktion sind. Die Unterteilung in zwei Tabellen dient der Übersichtlichkeit, in einer frühen Version von ObViViAn wurde zunächst nur eine Tabelle für alle Annotationen verwendet, was aber schnell als ungünstig zu benutzen erkannt und daher geändert wurde. Die Tabellen zum Contexter-Beispiel sind in Abbildung 4.4 zu sehen.

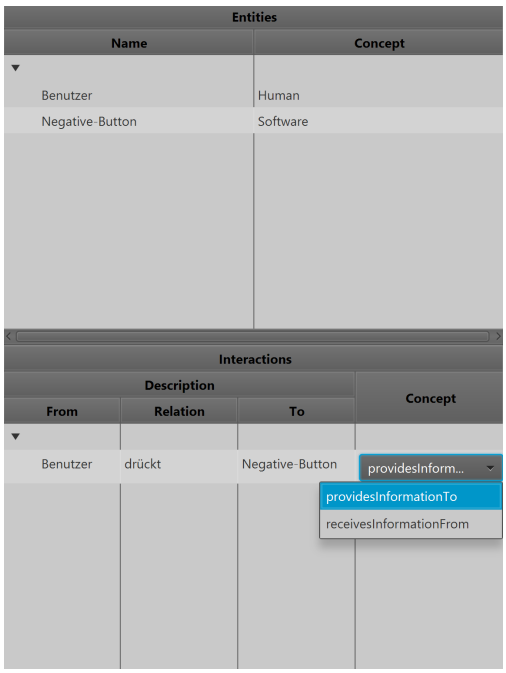


Abbildung 4.4.: Die tabellarische Ansicht von ObViViAn

4.1.3. Sichtbarkeitszeit im Video

Damit man die Zeit der Sichtbarkeit von Annotationen der Inhalte sehen und einstellen kann, wurde eine LineChart-Implementierung von JavaFX so manipuliert, dass sie eine Kategorienachse und eine Zeitachse zeigt. Darauf werden die Annotationszeiten der Entitäten mit einer horizontalen Linie auf Höhe ihres Namens dargestellt. Dabei können die Start- und Endpunkte mit gedrücktem linken Mausknopf sekundenweise angepasst werden. Eine solche Linie startet immer genau bei der aktuellen Stelle im Video und es wird zunächst angenommen, dass sie bis zum Ende des Videos sichtbar sein soll. Interaktionen hingegen werden auch durch Linien dargstellt. Allerdings verbinden diese diagonal die an der Interaktion beteiligten Entitäten. Dadurch entsteht eine eindeutige Leserichtung und die zeitliche Darstellung ist zusätzlich möglich. Abbildung 4.5 zeigt die Annotation von Benutzer und Software-Button aus dem aktuellen Beispiel.

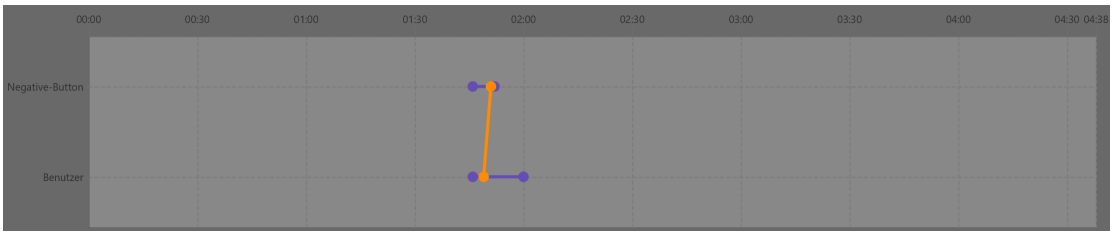


Abbildung 4.5.: Die Zeitansicht von ObViViAn

4.2. Verwendete Technologien

Dieser Abschnitt befasst sich mit den verwendeten Technologien zur Umsetzung des Prototypen. Dabei wird explizit auf die Programmiersprache der Programmlogik, die technische Gestaltung der grafischen Benutzeroberfläche (GUI) und die Integration von maschinenlesbaren Ontologien eingegangen.

Programmlogik und grafische Benutzeroberfläche

Für die Implementierung der Programmlogik wurde die objektorientierte Programmiersprache *Java* verwendet, da diese sowohl in der universitären Lehre gelehrt wird, als auch sehr ausführlich im World Wide Web dokumentiert ist. Zusätzlich ermöglicht sie eine plattformunabhängige Implementierung, so dass ein in Java geschriebenes Programm auf verschiedenen Rechnern laufen kann. Um eine zeitgemäße Technologie für die grafische Benutzerschnittstelle zu verwenden, entschied man sich für JavaFX[2]. Dadurch war nur wenig zusätzliche Technologie zu erlernen um die grafische Benutzerschnittstelle für ObViViAn zu erstellen.

Turtle – Terse RDF Triple Language

Zur Erzeugung und textuellen Darstellung einer Ontologie wurde die *Terse RDF Triple Language (Turtle)* verwendet, die eine konkrete Syntax für RDF ist. Ein in Turtle verfasstes Dokument stellt eine textuelle Repräsentation eines RDF-Graphen dar. Für verschiedene Kontexte beziehungsweise Vokabulare können Präfixe definiert werden. Diese bestehen aus einer sogenannten *IRI* (*Internationalized Resource Identifier*), welche dem definierten Präfix eine Eindeutigkeit beziehungsweise Einzigartigkeit innerhalb der Ontologie verleiht.

Graphdatenbanken

Im Gegensatz zu relationalen Datenbanken speichern Graphdatenbanken die Inhalte nicht in Tabellen, die nach Spalten orientiert sind. Sie realisieren Beziehungen über Fremdschlüssel aus anderen Tabellen und benötigen keine JOIN-Operationen oder sogar -Konstrukte in den Abfragen. Somit benötigen sie auch keine JOIN-Tabellen, also zusätzliche Tabellen für Beziehungen mit höheren Kardinalitäten. RDF-Triple-Datenbanken speichern Tripels, das heißt Subjekt-Prädikat-Objekt-Fakten, die die Grundlage für das RDF-Modell bilden. Ergänzt werden diese von Komponenten und Schichten in einigen Systemen, die dazu dienen, Schemas zu behandeln und die OWL-Teile von RDF zu unterstützen. Diese Datenbanken konzentrieren sich auf das Bearbeiten von Triples und Mengenoperationen zwischen diesen. Komplexe, tiefgehende Anfragen stoßen hier auf die gleichen Grenzen wie bei relationalen Datenbanken.

[2] `https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm`

```
PREFIX : <http://bedrock/>
INSERT DATA {
  :fred      :hasSpouse  :wilma .
  :fred      :hasChild   :pebbles .
  :wilma     :hasChild   :pebbles .
  :pebbles   :hasSpouse  :bamm-bamm ;
             :hasChild   :roxy, :chip .
}
```

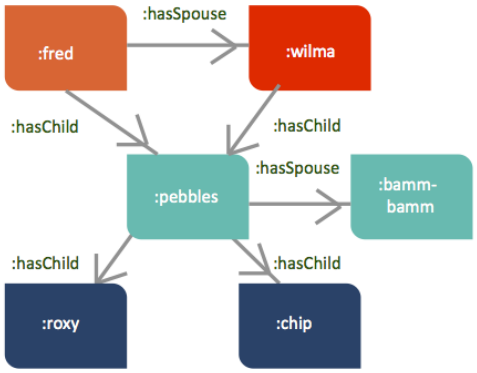


Abbildung 4.6.: Eine beispielhafte INSERT-Query für die Instanziierung von Familienverhältnissen anhand der Familie Feuerstein[4]

Auch erfordert die Bildung von Ontologien ein hohes Maß an Zusammenarbeit, um gemeinsame OWL-Strukturen zu bilden. Diese Vorgehensweise erschwert einerseits eine schnelle Entwicklung von RDF-Systemen, zeigt aber eine gute Langzeitwirkung. Das RDF-Modell wird inzwischen oft bei der Langzeitorganisation von Daten angewendet, da sich OWL-Ontologien gut zur Dokumentation von Datenstrukturen eignen. In der Systementwicklung finden OWL und RDF jedoch begrenzte praktische Verwendung als Datenmodell. Tripel-Datenbanken nutzen seit längerem die Mustersuche mit der SPARQL-Abfragesprache *SPARQL*[3], um vorzugeben, welchen strukturellen Kriterien die gewünschten Daten zu genügen haben. Die Tripel werden dort direkt als Paar-Muster vorgegeben. Die Abfragen aus der für jedes Annotationsprojekt erzeugten Datenbank werden daher ebenfalls mit *SPARQL* durchgeführt. Diese ähnelt der von relationalen Datenbanken bekannten Sprache *SQL* in der Syntax. Abbildung 4.6 veranschaulicht eine *INSERT DATA*-Query in *SPARQL*, wobei allgemeine familiäre Beziehungen mit den Familienmitgliedern der Famile Feuerstein konkretisiert werden.

4.3. Grundlegende Ontologie

Für eine Ableitung von Instanzen aus einer Ontologie ist zuerst einmal diese Basis zu erschaffen gewesen. Ohne eine konkrete Vorlage bezüglich einer Kundendomäne konnte die Ontologie nur in Bezug auf die geforderte, grafische Annotation von Videoinhalten gestaltet werden. Sie wurde daher mit den aus den Rohanforderungen abgeleiteten Fragen und verfeinerten Anforderungen gestaltet und enthält lediglich die grundlegenden Konzepte, welche in einer gefilmten Sequenz mit Bezug zu Interaktionen aus dem Software Engineering benötigt werden, nämlich menschliche und maschinelle Entitäten.

Um eine Konzepthierarchie bilden zu können, mussten logischer Weise verschiedenartige Entitäten von einem Oberkonzept “Entität“ ableiten. Dazu wurde

[3] https://www.w3.org/TR/rdf-sparql-query/

die OWL-Klasse *Entity* als Basis eingeführt und die Unterkonzepte *Mensch (Human)*, *Hardware* und *Software* daon abgeleitet. Um während einer Annotation nicht zuzuordnenden Sub- und Objekte speziell als noch nicht zuzuordnen markieren zu können, wurde zusätzlich die Klasse *Unassignable* eingeführt und ebenfalls von *Entity* abgeleitet. Zwei grundlegende Interaktionen, also mögliche Beziehungen zwischen Entitäten waren mindestens notwendig um daraus Interaktionen spezifiziert instanziieren zu können. Diese sind die absolut fundamentalen Beziehungen “Eine Entität sendet Informationen an eine andere Entität “ und “Eine Entität erhält Informationen von einer anderen Entität “. Diese muten im Englischen deutlich sinnvoller für den Kontext der Mensch-System-Interaktion an, welcher für die hier verwendeten Videos grundlegend ist. Sie sind in der formalen Ontologie als Objekteigenschaften *providesInformationTo* und *recceivesInformationFrom* realisiert.

Um die Ontologie tatsächlich für eine Video-Annotation zweckdienlich zu machen, wurden dem der Basis-Entität die notwendigsten Dateneigenschaften zugeordnet. Sie bestehen aus vier double- und zwei integer-Werten, von denen die double-Werte für die Position und Größe einer annotierten Entität innerhalb des Videobildes dienen und die Integer-Werte je einmal die Start- und die Endsekunde der Sichtbarkeit innerhalb der Laufzeit des Videos abbilden. In der Abbildung Abbildung 4.7 ist die erstellte Basis-Ontologie abgebildet, wobei gestrichelte Linien mit nicht ausgefülltem Pfeil eine Ableitung von einem Oberkonzept und Linien mit einem ausgefüllten Pfeil Objekt- oder Dateneigenschaften eines Konzeptes darstellen. Die Visualisierung wurde mit dem Web Service WebVOWL [5] erzeugt, in dem OWL- und RDFS-konforme Dateien hochgeladen und dann visualisiert werden können. Die hochgeladene Datei in der RDFS- und OWL-Sprache Turtle ist in Abbildung 4.8 zu sehen. Es bleibt zu erwähnen, dass die verwendete grundlegende Ontologie nur einen Bezug zum Software Engineering und zur allgemeinen Video-Annotation enthält. Eine reale oder fiktive Kundendomäne wurde nicht einbezogen, weil dieser Einbezug nicht zwangsläufig Gegenstand der Arbeit war, keine reale Implementierung vorlag und eine fiktive Kundendomäne keinen Mehrwert bot.

[5] `http://visualdataweb.de/webvowl/`

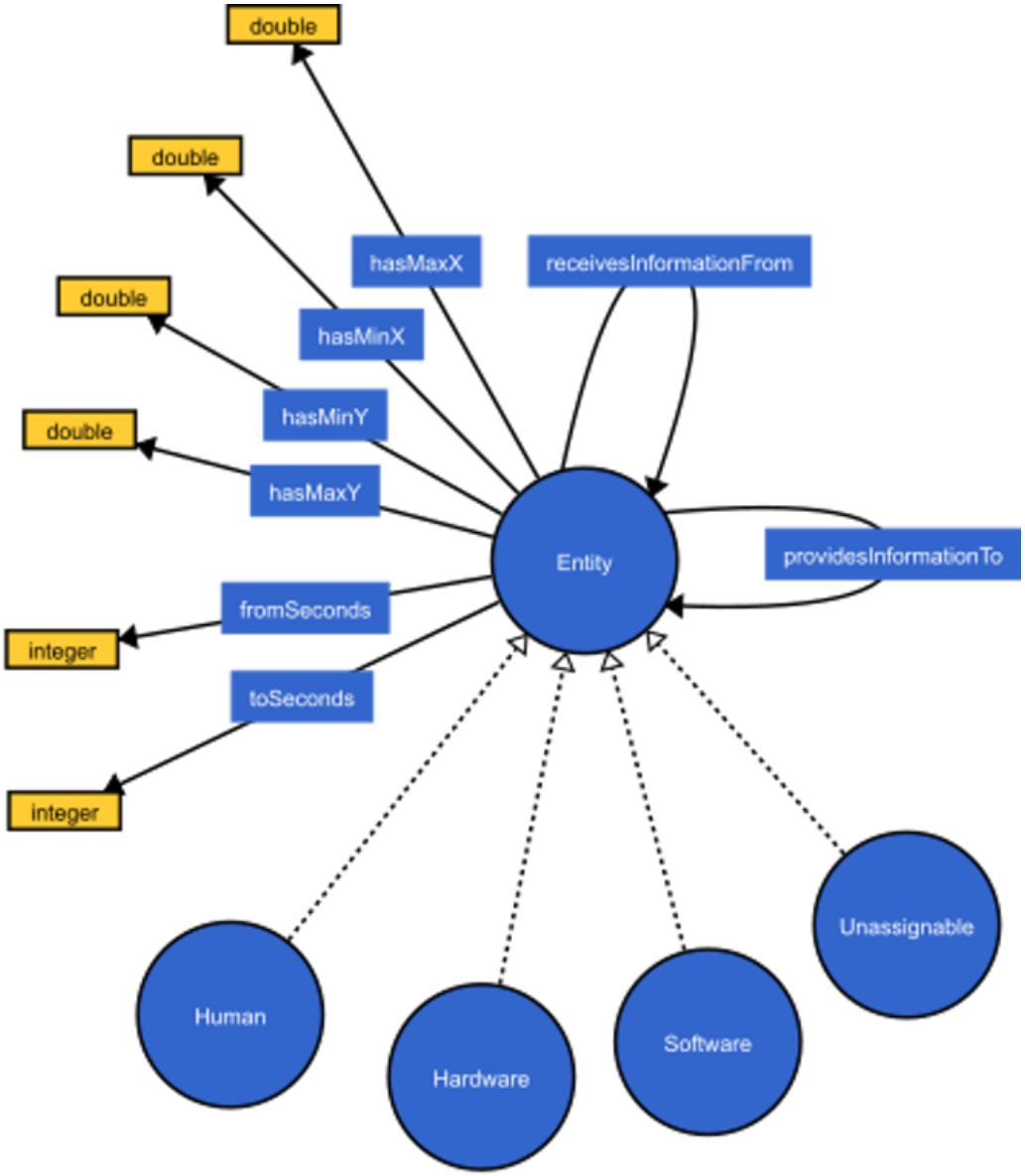


Abbildung 4.7.: Die Basis-Ontologie für eine grafische Annotation von Inhalten aus Videos als Grafik, visualisiert durch den Web Service WebVOWL

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix anno: <http://www.se.uni-hannover.de/annotation#> .
@prefix viz: <http://www.se.uni-hannover.de/annotation/viz#> .
@prefix data: <http://www.se.uni-hannover.de/annotation/data#> .

anno:Entity a owl:Class .

anno:Human a owl:Class ;
        rdfs:subClassOf anno:Entity .
anno:Hardware a owl:Class ;
        rdfs:subClassOf anno:Entity .
anno:Software a owl:Class ;
        rdfs:subClassOf anno:Entity .
anno:Unassignable a owl:Class ;
        rdfs:subClassOf anno:Entity .

anno:providesInformationTo a owl:ObjectProperty ;
        rdfs:domain anno:Entity ;
        rdfs:range anno:Entity .
anno:receivesInformationFrom a owl:ObjectProperty ;
        rdfs:domain anno:Entity ;
        rdfs:range anno:Entity .

viz:hasMinX a owl:DataTypeProperty ;
        rdfs:domain anno:Entity;
        rdfs:range xsd:double .
viz:hasMinY a owl:DataTypeProperty ;
        rdfs:domain anno:Entity;
        rdfs:range xsd:double .
viz:hasMaxX a owl:DataTypeProperty ;
        rdfs:domain anno:Entity;
        rdfs:range xsd:double .
viz:hasMaxY a owl:DataTypeProperty ;
        rdfs:domain anno:Entity;
        rdfs:range xsd:double .
viz:fromSeconds a owl:DataTypeProperty ;
        rdfs:domain anno:Entity;
        rdfs:range xsd:integer .
viz:toSeconds a owl:DataTypeProperty ;
        rdfs:domain anno:Entity;
        rdfs:range xsd:integer .
```

Abbildung 4.8.: Die Basis-Ontologie für eine grafische Annotation von Inhalten aus Videos als formale Spezifikation in der OWL-Sprache Turtle

4.4. Problematiken bei Darstellung und Bearbeitung von Videos

Das Konzept, Interaktionen zwischen Entitäten anhand ihrer Sichtbarkeit in einem Liniendiagramm darzustellen hat zwar bereits beschriebene Vorteile, allerdings auch Nachteile unter gewissen Umständen. Hat man nämlich ein sehr langes Video zu annotieren, so wird die x-Achse für die Zeitdarstellung mit zunehmender Anzahl an Entitäten zu klein um sekundenweise die Auftretenszeit der Objekte einzustellen.

Ebenso führt das Überschreiten einer gewissen Anzahl von Annotationen zwangsläufig zu einem sehr unübersichtlichen Liniendiagramm, dessen inhaltliche Elemente zu erkennen dann wohl nicht mehr als komfortabel bezeichnet werden kann.

Grundsätzlich sind Änderungen und Erweiterungen von ObViViAn denkbar, die diesen Problemen entgegenwirken. So könnte man die zeitliche Darstellung horizontal und vertikal scrollbar machen, der Nutzen davon bezüglich der Übersichtlichkeit ist allerdings anzuzweifeln, da so mehr Nutzeraktionen nötig wären um bestimmte Interaktionsmuster im Diagramm zu finden.

Der Wert einer grafischen Annnotation auf Basis von Ontologien hängt also sowohl von der verwendeten Softwarelösung, als auch von der Beschaffenheit der zu annotierenden Videos ab. Sind diese zu lang oder enthalten zuviele Entitäten, so wäre der Wiederanschauungswert des Videos mangels Übersichtlichkeit nicht beträchtlich höher, der Aufwand um Interaktionen zu finden würde steigen und den des Lesens textueller Zeitangaben in einem Use Case oder einem anderen Textdokument des Software Engineerings übersteigen.

Zur Zeit ist ObViViAn noch nicht in der Lage, mehrere Videos in einem Projekt zu behandeln. Es muss für jedes Video ein gesondertes Projekt erstellt werden, für das dann auch ein eigenes GrapdhDB-Repository angelegt wird. Sollte das Werkzeug erweitert werden, wäre es sinnvoll, dass man über ein zentrales Repository für die Ontologie und die damit verbundene Annotation von verschiedenen Videos aus dem Entwicklungsprojekt verfügt. Die Technologie erlaubt es, auf jegliche URLs zuzugreifen und dort Zugriff auf Repositories zu erhalten.

Zuletzt ist zu erwähnen, dass sich nicht alle Vision Videos gleich gut für eine Annotation mit ObViViAn eignen. Die Unterschiede bezüglich der Darstellung von Inhalten besitzen je mehrere Eigenschaften, die ihre Annotierbarkeit beeinflussen. Die Eignung dieser Eigenschaften ist in den folgenden Tabellen 4.1, 4.2 und 4.3 je für die Annotierbarkeit von menschlichen und nicht menschlichen entitäten, sowie für Interaktionen aufgelistet.

menschliche Entitäten		
Darstellung durch	**grafisch annotierbar**	**Eindeutigkeit**
Schauspieler	ja	hoch
Texteinblendung	ja	gering
akkustische Erwähnung	nein	—

Tabelle 4.1.: Grafische Annotierbarkeit der Darstellung von menschlichen Entitäten

nicht menschliche Entitäten		
Darstellung durch	**grafisch annotierbar**	**Eindeutigkeit**
reale Objekte	ja	hoch
Texteinblendung	ja	gering
akkustische Erwähnung	nein	—

Tabelle 4.2.: Grafische Annotierbarkeit der Darstellung von nicht menschlichen Entitäten

Interaktionen		
Darstellung durch	**grafisch annotierbar**	**Eindeutigkeit**
alle beteiligten Entitäten	ja	hoch
nur eine Entität	nein	—

Tabelle 4.3.: Grafische Annotierbarkeit der Darstellung von Interaktionen

Eine Einteilung dafür, wie gut sich Darstellungsarten aus Annotationssicht eignen zeigt Abbildung 4.9. Dabei ist berücksichtigt, wie gut sich die Darstellungsart an sich eignet, allerdings auch in Bezug auf die Menge, die innerhalb einer Szene davon auftritt. Im Idealfall besitzt eine annotierte Szene zwei Entitäten und eine Interaktion, die alle klar sichtbar sind. Sind mehr zu annotierende Entitäten enthalten, so wird die Annotation automatisch unübersichlicher. Dies gilt ebenfalls für mehr als eine Interaktion. Texteinblendungen als Substitut für eine Entität sind noch annotierbar, gesprochener Text (Audiokommentar) zur Zeit gar nicht.

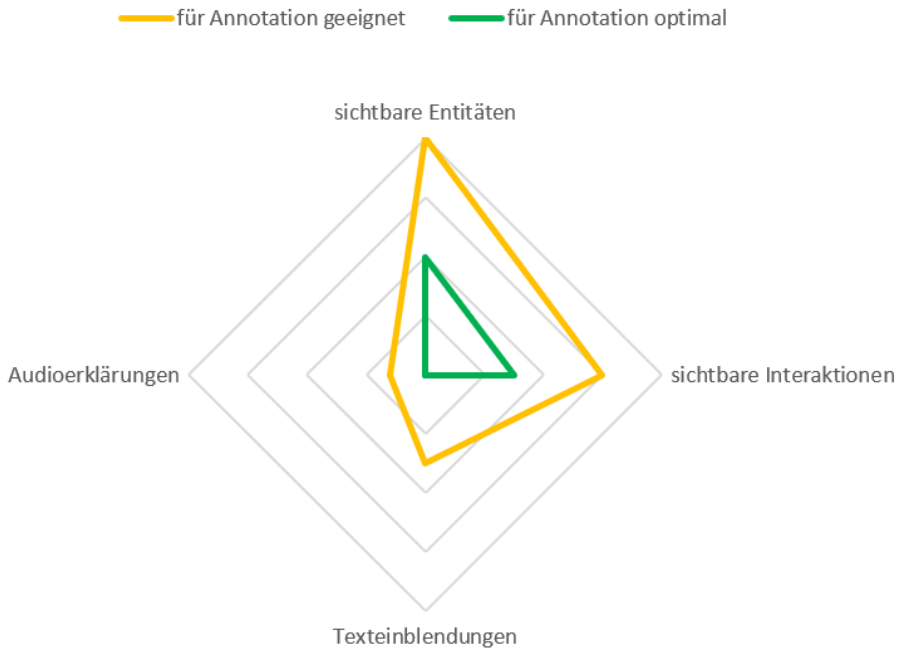


Abbildung 4.9.: Die Eignung bestimmter Darstellungsarten und zu wahrnehmbarer Mengen davon in einem zu annotierenden Vision Video

5. Evaluation

Im Rahmen dieser Arbeit wurde ein Ontologie-basierter Ansatz zur eindeutigen Zuordnung von Videoinhalten zu Domänenbegriffen entwickelt. Dieses Kapitel befasst sich mit der Evaluation zur Anwendbarkeit von Videos als Dokumentation mit Hilfe des implementierten Prototyps. Die Vorgehensweise beruht auf den von Wohlin et al. [36] vorgestellten Methoden zum Experimentieren im Software Engineering.

5.1. Vorbereitung mit GQM

Als Vorbereitung und wie in Kapitel 1 bereits erwähnt, wurde zunächst die Goal-Question-Metric nach Van Solingen et al. [34] angewandt um Ziele für eine Verbesserung des Nutzens von Videos im Software Engineering ableiten zu können. Mit Hilfe des GQM-Ansatzes werden erforderlichen Daten in Bezug auf die betrachteten Ziele und die damit verbundenen Metriken erfasst. Dadurch lässt sich der Aufwand in der Datenerfassung deutlich reduzieren und der Gesamtprozess der Interpretation von erhobenen Daten leichter bewerkstelligen. Da durch den GQM-Ansatz explizite Beziehungen zwischen den erhobenen Daten und den angestrebten Messzielen bestehen, werden Fehlinterpretationen der Daten unterbunden [34].

Als Ausgangspunkt diente der in Kapitel 1.2, Abbildung 1.2 vorgestellte Zielbaum, bei dem die Ziele **G 1.1.1 Verbessere Dokumentierbarkeit von Videoinhalten im Requirements Engineering** und **G 1.1.2 Verbessere Wiederbetrachtungswert von Vision Videos im Requirements Engineering** näher betrachtet. Der GQM-Methode folgend wurden dazu Fragen, Metriken und Indikatoren erarbeitet, anhand derer eine Beurteilung der zu erreichenden Ziele möglich war. Grundsätzlich war zu beurteilen, inwieweit die Ziele in der Praxis anwendbar sind, so dass eine Verbesserung der Zugänglichkeit von Vision Videos über eine Steigerung ihrer Dokumentierbarkeit und ihres Wiederbetrachtungswertes zu messen ist. Eines der entstandenen Abstraction Sheets ist in Abbildung 5.3 zu sehen. Die daraus weiter verfeinerten Ziele zeigen die Abbildung 5.1.

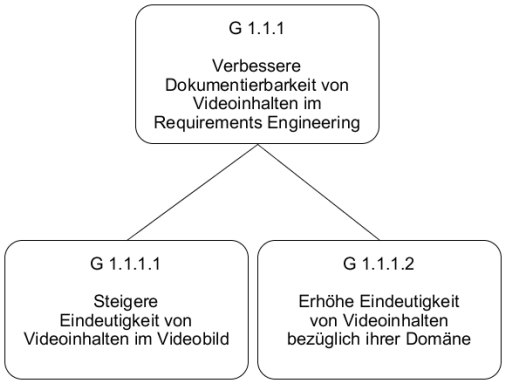


Abbildung 5.1.: Verfeinerung von G 1.1.1 bezüglich Dokumentierbarkeit von Videoinhalten

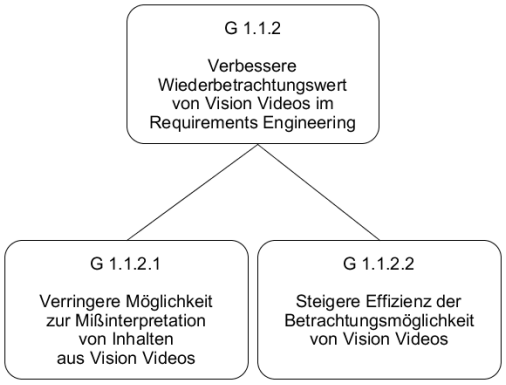


Abbildung 5.2.: Verfeinerung von G 1.1.2 bezüglich des Wiederbetrachtungswertes von Vision Videos

Zweck	**Qualitätsaspekt**	**Betrachtungsgegenstand**	**Perspektive**
Steigere	*Dokumentierbarkeit*	*Videoinhalt*	*Requirements Engineer*
Qualitätsfaktoren:		Einflussfaktoren:	
Möglichkeit der maschinenlesbaren Klassifikation von relevanten Inhalten aus Vision Videos in Domänenbegriffe direkt im Video.		Markierte Inhalte in Videos mit konkreter Zuordnung zu Domänenbegriffen, verfügbar über ein Werkzeug.	
Ausgangshypothese:		Einflusshypothese:	
Aktuell existiert keine direkte Zuordnungsmöglichkeit von dargestellten Inhalten aus Vision Videos zu Domänenbegriffen. Inhaltsangaben können lediglich in Form von Metadaten angebracht werden (Metadaten). Es gibt keinen Nutzen von Vision Videos außerhalb der Elicitation, Interpretation und Negotiation im RE.		Mit einer Zuordnung von in Vision Videos konkret dargestellten Entitäten und deren Beziehungen untereinander kann ein Video besser in die Dokumentation einbezogen und für Entwickler zugänglich gemacht werden.	

Abbildung 5.3.: Abstraction Sheet zur Dokumentierbarkeit von Videoinhalten

5.2. Planung des Experiments

Das Ziel des Experiments und dessen Evaluation ist die Prüfung der Eignung von Videos als Dokumentationsartefakt mit Hilfe von genauer Zuordnung der zu sehenden Inhalte zu übergeordneten Begriffen.

5.2.1. Design

Das Experiment folgt dem *Between-Subjects-Design*, bei dem jeder Teilnehmer nur eine der beiden zur Verfügung stehenden Techniken verwendet. Entweder der Proband hatte für das Betrachten der Videos nur ein herkömmlichen Player für Videodateien, oder den Prototypen mit annotierten Videos zur Verfügung.

Auswahl der unabhängigen Variablen des Experiments

Zu Durchführung einer Evaluation ist es zunächst notwendig, die zu prüfenden unabhängigen und abhängigen Variablen zu bestimmen. Mit abhängigen Variablen misst man die Auswirkung des betrachteten Verfahrens auf die Erledigung der Aufgabe durch Probanden. Dabei ist es selten möglich, diese direkt zu messen, stattdessen misst man meist mithilfe von indirekten Maßen. Unabhängige Variablen sind steuerbar, haben allerdings auch Auswirkungen auf die abhängigen Variablen. Für die Versuche standen der jeweiligen Gruppe eine bestimmte Technik zur Verfügung. Diese repräsentiert die unabhängige Variable des Experiments:

ObViVian Verwendung des für diese Arbeit entwickelten Prototypen einer Ontologie-basierten Annotations-Software

VLC Player Ein reiner Video-Player für die Betrachtung der Videos

Die Auswahl der zu messenden abhängigen Variablen wird in Abschnitt 5.2.1 über die Forschungsfragen in Abschnitt 5.2.1 hergeleitet.

Verwendete Videos und Begriffe

Für das Experiment wurde ein visionsartiges Video[1] benutzt, welches die Funktionsweise eines Systems zur Sicherung der Cockpittür eines Passagierflugzeugs zeigt. Obwohl die Struktur eines Videos für die Darstellung seiner Inhalte relevant ist und eine Einführung vor den primären Inhalten und ein klares Ende besitzen sollte [26, 21, 15], wurde für dieses Experiment entschieden, das ursprüngliche Video in drei Teile zu schneiden und die anfängliche Einführung zu entfernen. Diese Einführung hätte jeglichen Klärungsbedarf von vorn herein minimiert. Im Original-Video sind drei unterschiedliche Use Cases veranschaulicht, wobei jeder einzelne ebenfalls eine klare Einleitung, einen dargestellten Inhalt und ein klares Ende zeigt. Um einen

[1] `https://www.youtube.com/watch?v=ixEHV7c3VXs`

Ontologie-Bezug für den Benutzer herzustellen, hat man sich dazu entschlossen die Videos in drei separate Teil-Videos zu schneiden und die Einführung in die Szenerie als Ontologie-Instanz in Diagrammform zur Verfügung zu stellen. Der Erklärungsteil entfiel daher. Die Videos hatten eine unterschiedliche zeitliche Länge, die in Tabelle 5.1 angegeben ist.

Spieldauer	Video		
	Use Case 01	Use Case 02	Use Case 03
Minuten	1:14	1:15	1:27
Sekunden	74	75	87

Tabelle 5.1.: Die Spieldauer der Videos zu den einzelnen Use Cases

Zu den drei gezeigten Szenarien wurde jeweils ein Use Case in Tabellenform nach Cockburn [10] verfasst und um für eine einheitliche Begriffsnutzung zu sorgen eine Ontologie-Instanz aus vorliegenden *.ecore- und *collaboration- Dateien erstellt. Diese waren zuvor unabhängig von dieser Arbeit aus dem Ausgansvideo abgeleitet worden, ursprünglich um mit dem Werkzeug ScenarioTools [2] ein Zustand-basiertes Szenario zu prüfen. Mehrere Mitarbeiter des Teams von *scenariotools.org* waren an der Erstellung des zugrunde liegenden formalen Szenarios beteiligt. Daher sind die Begrifflichkeiten, auf der die im Versuch verwendete Ontologie-Instanz beruht nicht willkürlich für den Versuch konstruiert worden, sondern basieren auf einer validen Festlegung durch eine externe Personengruppe. Die verwendeten Begriffe wurden konsistent über die Use Case Tabellen und die Ontologie-Instanz verwendet. Für die Versuchsgruppe waren wichtige Begrifflichkeiten daraus in ObViViAn annotiert und damit die entsprechenden Stellen im Video per Maus innerhalb des Werkzeugs direkt abspielbar. Somit war der für die Evaluation wichtige Klärungsbedarf war somit nur über die Nutzungsszenarien in schriftlicher Form (Use Case Tabelle), Diagrammform (Ontologie-Instanz) und Video zu beseitigen.

Forschungsfragen

Beim Experiment ging darum zu evaluieren, ob Benutzer einen Mehrwert von der Nutzung von Videos als Dokumentationsoption haben und deren Inhalte für sie eindeutiger zu verstehen sind, wenn sie über eine Ontologie zusätzlich in Oberbegriffe eingeordnet werden. Wichtig war dabei zu ermitteln, ob Entwickler mit einem Ontologie-Bezug der Inhalte mehr Klärungsbedarf beseitigen können als ohne und ob dies schneller geht oder kein zeitlicher Unterschied existiert. Mit der Durchführung des Experiments sollten daher die folgenden Forschungsfragen evaluiert werden:

[2] `http://scenariotools.org`

I.) Sind Entwickler mit einer Ontologie-basierten Annotation von Videoinhalten schneller dazu in der Lage, anhand annotierter Videos Fragen richtig zu beantworten als mit nicht annotierten Videos?

II.) Sind Entwickler mit einer Ontologie-basierten Annotation von Videoinhalten dazu in der Lage mehr Fragen richtig zu beantworten als mit nicht annotierten Videos?

Zu messende abhängige Variablen

Basierend auf den im vorherigen Unterabschnitt formulierten Forschungsfragen konnten zu messende abhängige Variablen identifiziert werden. Diese wurden um das subjektive Empfinden der Probanden in Bezug auf Videos als Ontologie-basierte Möglichkeit zur Dokumentation erweitert:

I. Die gemessene Zeit zur Beantwortung der Fragen zu jedem Use Case

II. Die Anzahl korrekt beantworteter Fragen zu jedem Use Case

III. Die Höhe des subjektiv empfundenen Unterstützungswertes der Videos zur Beantwortung der Fragen

IV. Der subjektiv empfundene Aufwand, die Videos für die Beantwortung der Fragen anzusehen

V. Die Höhe des subjektiv empfundenen Unterstützungswertes der Ontologie-Instanz in Diagrammform

VI. Der subjektiv empfundene Mehrwert einer Verknüpfung von Videoinhalten über eine Ontologie

Zu testende Hypothesen

Um mit Hilfe eines Hypothesentests die Forschungsfragen zu klären, wurden die folgenden, zu testenden Nullhypothesen aufgestellt, wobei sich die jeweilige Alternativhypothese aus dem Gegenteil ergibt. Das bedeutet, wenn eine Hypothese von *keinem Unterschied* zwischen verglichenen Variablen ausgeht, ihre Alternative besagt, dass *ein Unterschied existiert*.

H_{0_1} Es gibt keinen zeitlichen Unterschied zwischen der Entwickler-seitigen Klärung von Fragen mit und ohne Werkzeugunterstützung.

H_{0_2} Es gibt keinen Unterschied bezüglich der Anzahl korrekt beantworteter Fragen zu Use Cases mit einer Ontologie-basierten Video-Annotation und ohne eine solche.

5.3. Durchführung

Die zur Durchführung angetretene Population und der Ablauf einer Sitzung des Experimentes werden in diesem Abschnitt erklärt.

5.3.1. Population

An dem Experiment haben insgesamt 12 Probanden teilgenommen, die alle eine Mindestgrundlage an Vorwissen bezüglich Software Engineering besitzen. 11 Teilnehmer besaßen zum Zeitpunkt der Durchführung einen Universitätsabschluss im Fach Informatik. 6 davon haben den Bildungsgrad eines Bachelor of Science Informatik erreicht und befanden sich zur Zeit der Versuchsdurchführung im fortgeschrittenen Master-Studiengang des Faches. 5 Probanden besaßen bereits einen Master of Science im Fach Informatik, einer hatte bereits einen Doktorgrad erreicht. Ein einzelner Teilnehmer hatte den Master of Science im Fach Mathematik mit Schwerpunkt Informatik absolviert. Alle Teilnehmer hatten mindestens die Vorlesung Grundlagen der Softwaretechnik an der Leibniz Universität Hannover oder eine äquivalente Veranstaltung an einer anderen Hochschule besucht.

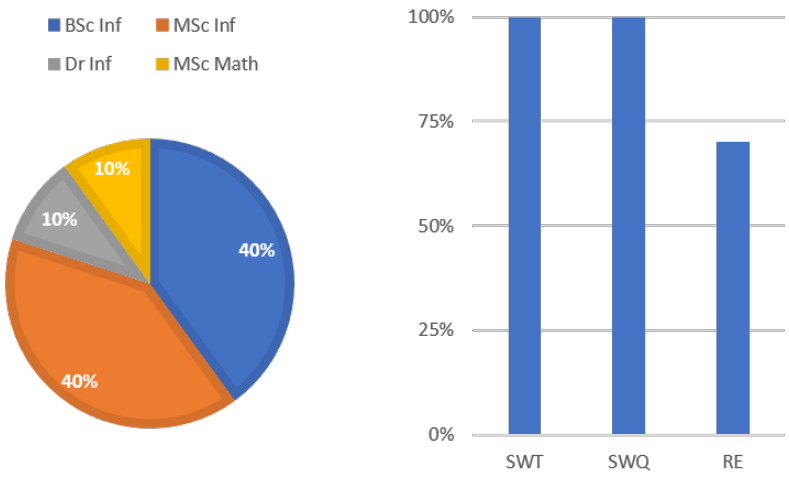


Abbildung 5.4.: Population nach Bildungsgrad und Fachrichtung und dessen Vorwissen bezüglich Veranstaltungen an der Leibniz Universität Hannover[3] oder äquivalenten an anderen Hochschulen

Eine Versuchsperson aus der Kontrollgruppe hatte deutlich weniger Vorwissen bezüglich besuchter Veranstaltungen über Software Engineering und wurde deswegen nicht gewertet. Eine andere Person aus der Versuchsgruppe hat während des

[3] *SWT* steht für die Veranstaltung *Softwaretechnik*, *SWQ* für *Softwarequalität* und *RE* für *Requirements Engineering*

Experimentes nicht konzentriert gearbeitet, was bereits während der Durchführung offensichtlich wurde. Daher wurde auch bei dieser Person entschieden, ihre Ergebnisse bei der Auswertung nicht zu berücksichtigen. Über die Gründe dafür kann nur spekuliert werden. Da es sich hierbei um jeweils einen Teilnehmer jeder Gruppe, wurden diese beiden aus der Wertung gestrichen. Die gewertete Population bestand daher aus 10 Probanden mit dem in Abbildung 5.4 dargestellten fachlichen Hintergrund und Vorwissen.

Zusätzlich zu den Kontrollfragen, der Zeitmessung und den Präferenzen wurden ebenfalls Beobachtungen während der Versuchsdurchführung notiert. Diese bezogen sich hauptsächlich auf Unterschiede im Vorgehen bei den verschiedenen Probanden.

5.3.2. Ablauf einer Sitzung

In jeder Sitzung ging es darum Fragen zu je drei Use Cases zu beantworten. Als zusätzliche Hilfsmittel waren ein ausgedrucktes Diagramm einer Ontologie-Instanz und ein Vision Video, das unter anderem die Schritte des Use Cases mit Darstellern und Kulisse zeigt. Bei jeder einzelnen Sitzung bekam jeder Teilnehmer zunächst eine Seite Einführung zu lesen, die Probanden der Versuchsgruppe erhielten zusätzlich eine Einführung in den Betrachtungsmodus des Prototypen. Anschließend erhielt ein jeder die in Diagrammform ausgedruckte Ontologie-Instanz mit Wissen über Entitäten und Beziehungen aus dem Szenario. Die erste Zeitmessung startete nach Signalisierung der Startbereitschaft durch den Teilnehmer mit der Aushändigung des ersten Use Cases in Tabellenform, der Fragen dazu und des geöffneten, aber noch nicht gestarteten Videos. Danach konnten die Probanden selbst entscheiden, welche Dokumente sie benutzen um die Fragen zu beantworten. Da es in der Arbeit um die Eignung von Videos als Dokumentationsoption über eine Ontologie ging, war es für eine korrekte Beantwortung der Fragen zwingend notwendig bestimmte Videoszenen anzusehen. Diese waren bei der Versuchsgruppe mit Unterstützung durch ObViViAn annotiert und konnten so über zwei zuvor erklärte Optionen direkt angesehen werden. Sobald ein Proband signalisierte, dass er alle drei Fragen zu einem Use Case beantwortet hatte wurde die Zeit gestoppt. Während der Durchführung wurde zusätzlich beobachtet, wie der jeweilige Teilnehmer vorgeht und welche Hilfsmittel er bevorzugt verwendet. Im letzten Teil der Durchführung bekamen die Probanden noch zwei Seiten, auf denen sie unterschiedliche Felder auszufüllen hatten. Zunächst gab es eine Seite mit vier Aussagen, denen sie ihrer Zustimmung nach einen von 5 Werten zwischen einschließlich "*stimme gar nicht zu*" und "*stimme voll zu*" anzukreuzen hatten. Auf der letzten auszufüllenden Seite hatten die Teilnehmer dann noch Informationen zu ihrer Person anzugeben, und zwar ihren aktuell höchsten Bildungsabschluss, dessen Fachrichtung und ihr Vorwissen bezüglich Software Engineering in Form von besuchten Veranstaltungen an der Leibniz Universität oder äquivalenten Veranstaltungen an einer anderen Hochschule. Zum Abschluss erhielt jeder Proband eine kleine Aufwandsentschädigung in Form von ungesunder Ernährung und konnte sich dann wieder seiner eigenen Bildung widmen.

5.4. Auswertung

In diesem Abschnitt werden zunächst die Ergebnisse des Experiments präsentiert. Dazu wird die Menge der Probanden mit Werkzeugunterstützung im Folgenden als *Versuchsgruppe* und die Menge ohne Unterstützung als *Kontrollgruppe* bezeichnet. Es werden alle statistisch verwertbaren Resultate vorgestellt, dazu zählen gemessene Zeiten, Anzahl beziehungsweise Prozentsatz korrekt beantworteter Fragen und Zustimmungswerte zu bestimmten Aussagen.

5.4.1. Ergebnisse der Evaluation

Die statistischen Ergebnisse aller gewerteten Sitzungen des Experiments beinhalten die Zeiten zur Beantwortung der Fragen und die Anzahl korrekt beantworteter Fragen pro Use Case mit zugehörigem Video. Die Resultate der Kontroll- und Versuchsgruppe wurden gegenüber gestellt, um daraus in der Interpretation der Ergebnisse 5.4.2 Schlüsse hinsichtlich der verfolgten Ziele ziehen zu können.

Da die Zeit für die Beantwortung der Fragen zu den einzelnen Use Cases separat gemessen wurde, ergaben sich auch drei separate Vergleichsmöglichkeiten, die statistisch auszuwerten waren. Für jede wurde zunächst getestet, ob die Ergebnisse normal verteilt sind. Da dies für alle der Fall war, konnte der *T-Test* auf statistische Signifikanz für jede Messreihe durchgeführt werden. Dazu wurde für jede Messreihe ein Signifikanzniveau von 0,05 festgelegt.

Gemessene Zeiten Use Case 1:

Bei den gemessenen Zeiten für Use Case 1 lag der Durchschnittliche Wert über alle Probanden hinweg bei 4:37 Minuten beziehungsweise 277,3 Sekunden. Die Kontrollgruppe beantwortete die Fragen dabei in durchschnittlich 3:11 Minuten, die Versuchsgruppe hingegen in 6:03 Minuten. Der T-Test ergab hierbei einen statistisch signifikanten Unterschied zwischen den beiden Teilnehmergruppen mit eine t-Wert von $3,3753$ und einem p-Wert von 0,009707. Da der p-Wert deutlich unter dem Signifikanzniveau liegt ist das Ergebnis der zeitlichen Messungen zu Use Case 1 als statistisch signifikant anzusehen.

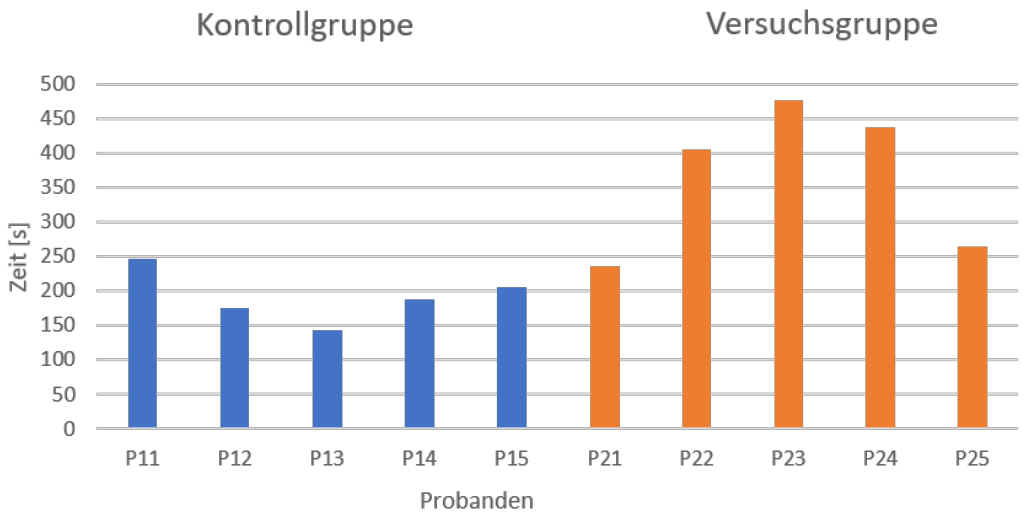
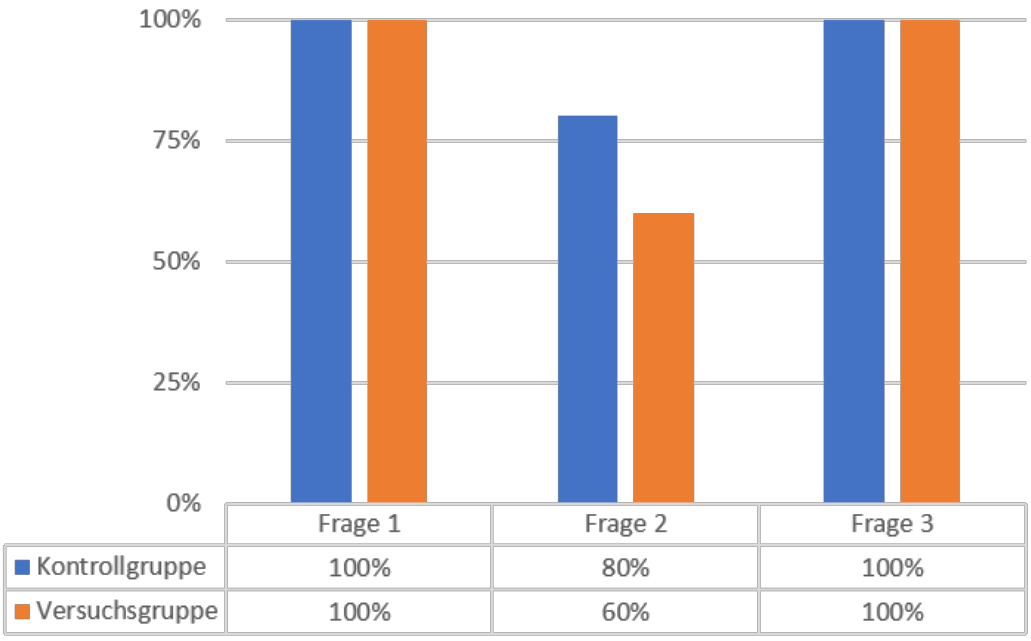


Abbildung 5.5.: Benötigte Zeiten für die Beantwortung der Fragen zu Use Case 1

Beantwortung der Fragen zu Use Case 1:

Die Fragen zu Use Case 1 wurden von allen Teilnehmern insgesamt zu insgesamt 90% richtig beantwortet, wobei die die Versuchsgruppe 87% korrekte Antworten gab und die Kontrollgruppe 93%. Insgesamt gab es 3 falschen Antworten, welche alle auf Frage 2 gegeben wurden. Zweimal lagen Probanden der Versuchsgruppe daneben und einmal eine Person aus der Kontrollgruppe. Damit wurde Frage 2 nur zu 70% insgesamt korrekt beantwortet. Zu den Fragen 1 und 3 kreuzten alle Probanden die richtige Antwort an.



	Frage 1	Frage 2	Frage 3
Kontrollgruppe	100%	80%	100%
Versuchsgruppe	100%	60%	100%

Abbildung 5.6.: Korrekte Antworten zu Use Case 1

Gemessene Zeiten Use Case 2:

Die gemessene durchschnittlich Zeit aller Teilnehmer für die Beantwortung der Fragen zu Use Case 2 betrug 3:25 Minuten. Dabei brauchte die Versuchsgruppe im Schnitt 4:14 Minuten und die Kontrollgruppe 2:36 Minuten. Mit der Überprüfung durch den T-Test ergab sich ein t-Wert von $2,5908$ und ein p-Wert von $0,032073$, wodurch die Messreihe mit $p < 0,05$ noch eine statistische Signifikanz aufweisen konnte.

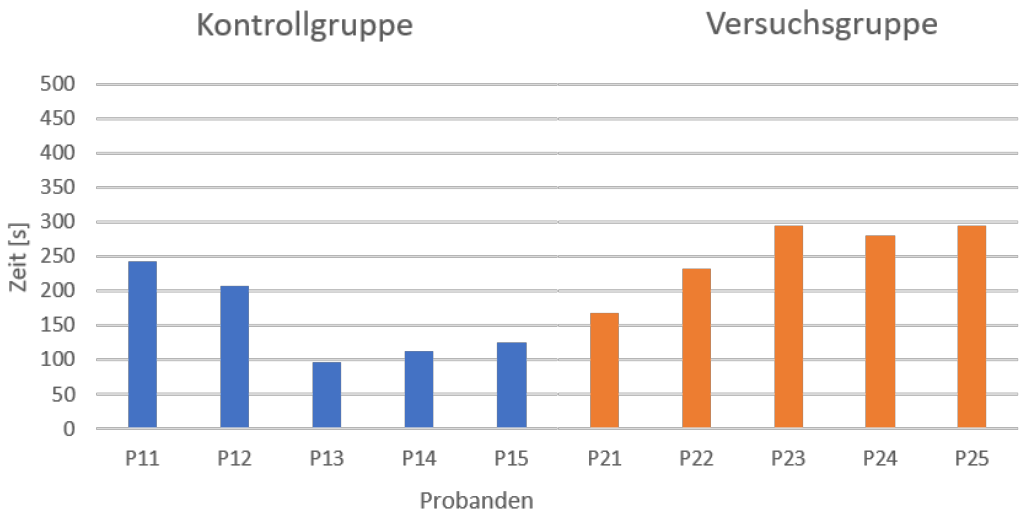


Abbildung 5.7.: Benötigte Zeiten für die Beantwortung der Fragen zu Use Case 2

Beantwortung der Fragen zu Use Case 2:

Die gegebenen Antworten auf die Fragen zu Use Case 3 waren über alle Probanden beider Gruppen zu 100% korrekt. Somit wurde jede einzelne Frage von allen Teilnehmern richtig beantwortet.

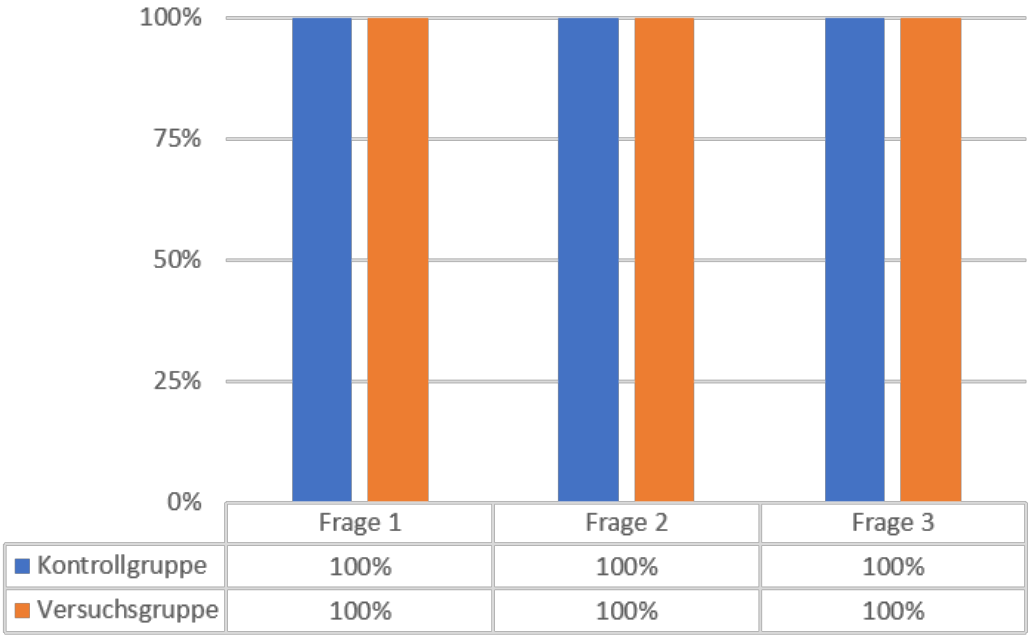


Abbildung 5.8.: Korrekte Antworten zu Use Case 2

Gemessene Zeiten Use Case 3:

Die letzten zu messenden Zeiten bezüglich der Beantwortung der drei Fragen zu Use Case 3 hatten einen Durchschnitt von 2:28 Minuten für alle Teilnehmer des Experiments. Die Kontrollgruppe benötigte im Mittel 2:21 Minuten, die Probanden der Versuchsgruppe ließen sich durchschnittlich 3:16 Minuten Zeit. Eine statistische Signifikanz konnte bei dieser Messreihe mit $t = 1,38812$ und $p = 0,202532$ nicht mehr untermauert werden, weil sich dabei $p > 0,05$ ergab.

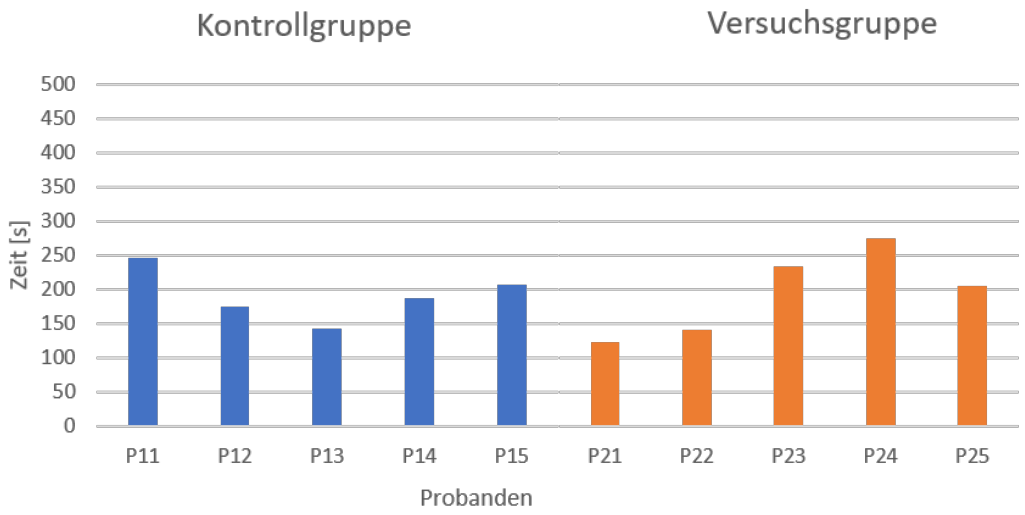
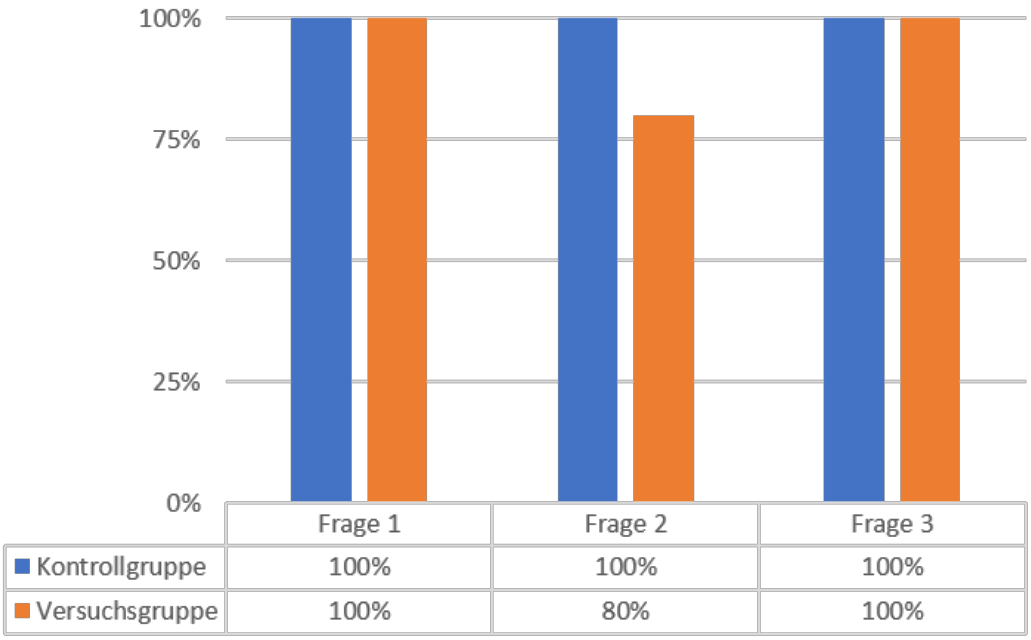


Abbildung 5.9.: Benötigte Zeiten für die Beantwortung der Fragen zu Use Case 3

Beantwortung der Fragen zu Use Case 3:

Ein einziger Proband gab bei den Fragen zu Use Case 3 eine falsche Antwort. Damit wurden die Fragen gruppenübergreifend zu 97% richtig beantwortet. Die einzig falsche Antwort gab ein Teilnehmer der Versuchsgruppe, so dass der Prozentsatz korrekter Antworten dieser Gruppe bei 93% lag und die Kontrollgruppe 100% erreichte. Alle Probanden beantworteten die Fragen 1 und 3 richtig, die eine falsche Antwort fand sich auf Frage 2. Damit hatten die erste und dritte Frage je eine Quote von 100% richtiger Antworten in beiden Gruppen und daraus folgend auch über alle Probanden hinweg. Frage 2 erreichte 100% bei der Kontrollgruppe und 80% bei der Versuchsgruppe, was insgesamt einen prozentualen Anteil von 90% ergab.



	Frage 1	Frage 2	Frage 3
Kontrollgruppe	100%	100%	100%
Versuchsgruppe	100%	80%	100%

Abbildung 5.10.: Korrekte Antworten zu Use Case 3

Messergebnisse der Zustimmung zu den gegebenen Aussagen

Aussage 1:
Die Videos haben geholfen, die Inhalte der Use Cases besser zu verstehen.

Die Aussage 1 erhielt über alle Teilnehmer hinweg eine Zustimmung von 94%, was bedeutet, dass 8 von 10 Probanden voll zustimmten. Alle Personen der Kontrollgruppe erteilten ihre volle Zustimmung, in der Versuchsgruppe wurde 3 mal voll zugestimmt und je einmal überwiegend und teilweise. Ein statistisch signifikanter Unterschied konnte bei einem t-Wert von 1,5 und einem p-Wert von $p = 0,172003 > 0,05$ nicht nachgewiesen werden. Der Median beider Gruppen lag bei einem Zustimmungswert von 5, also voller Zustimmung.

Zustimmung		Kontroll-gruppe	Versuchs-gruppe
Option	Wert		
voll	5	5	3
überwiegend	4	0	1
teilweise	3	0	1
kaum	2	0	0
gar nicht	1	0	0
Median		5	5

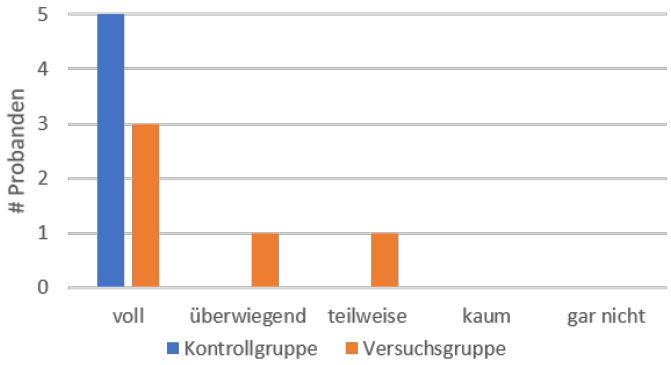


Abbildung 5.11.: Zustimmung zu Aussage 1 in Tabellen- und Diagrammform

Aussage 2:
Die Videos waren über die Ontologie zugänglicher.

Bei Aussage 2 gaben die Probanden beider Gruppen zusammen eine Zustimmung von 52%. Dabei viermal gar nicht zugestimmt, je zweimal pro Gruppe. Teilweise stimmten insgesamt 3 Personen zu, davon zwei aus der Kontrollgruppe und eine aus der Versuchsgruppe. Überwiegende Zustimmung wurde je einmal pro Gruppe angegeben, die volle Zustimmung erhielt diese Aussage nur einmal aus der Versuchsgruppe. Damit stimmte die Kontrollgruppe zu 48% zu, der prozentuale Zustimmungsanteil der Versuchsgruppe lag bei 56%. Der T-Test ergab hierzu ebenfalls keine statistische Signifikanz bei $t = 0,4$ und einem p-Wert von $0,699625 > 0,05$. Der Median für Kontroll- und Versuchsgruppe lag jeweils bei teilweiser Zustimmung beziehungsweise dem Wert 3.

Zustimmung		Kontroll-gruppe	Versuchs-gruppe
Option	Wert		
voll	5	1	0
überwiegend	4	1	1
teilweise	3	1	2
kaum	2	1	0
gar nicht	1	1	2
Median		3	3

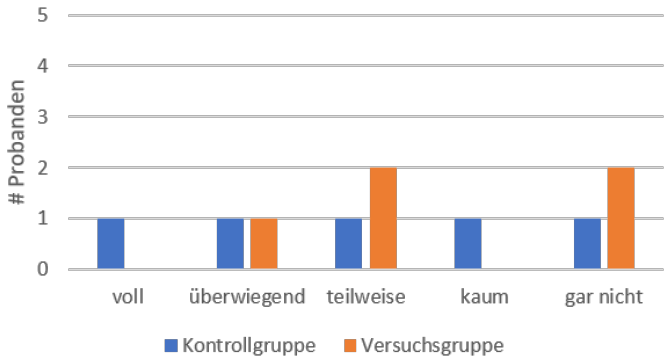


Abbildung 5.12.: Zustimmung zu Aussage 2 in Tabellen- und Diagrammform

Aussage 3:
Der Aufwand, die Videos anzuschauen war zu groß.

Bei Aussage 3 fiel die Zustimmung sehr verhalten aus. Kein Teilnehmer stimmte ihr voll oder überwiegend zu. Nur ein Proband aus der Versuchsgruppe stimmte teilweise zu, je einer aus jeder Gruppe kaum. Die meisten Teilnehmer befanden die Aussage für gar nicht zutreffend. Damit befand sich das statistische Mittel aller Präferenzangaben zu Aussage 3 ebenfalls bei gar keiner Zustimmung beziehungsweise beim Zahlenwert 1. Damit ergab sich ein prozentualer Gesamtanteil von 28% bezüglich der Zustimmung zu dieser Aussage. Die Kontrollgruppe stimmte viermal gar nicht und einmal kaum zu, die Versuchsgruppe dreimal gar nicht und je einmal kaum und teilweise. Prozentual gesehen erhielt Aussage 3 von der Kontrollgruppe eine 24%-ige und von der Versuchsgruppe eine 32%-ige Zustimmung. Auch hier ergab sich kein statistisch signifikanter Unterschied, da bei $t = 0,89443$ ein p-Wert von 0,397204 errechnet wurde, der eine statistische Signifikanz durch seine Eigenschaft $p > 0,05$ widerlegte. Im Median stimmten beide Gruppen gar nicht zu, er lag sowohl bei der Versuchsgruppe, als auch der Kontrollgruppe jeweils beim niedrigsten Wert 1.

Zustimmung		Kontroll-gruppe	Versuchs-gruppe
Option	Wert		
voll	5	0	0
überwiegend	4	0	0
teilweise	3	0	1
kaum	2	1	1
gar nicht	1	4	3
Median		1	1

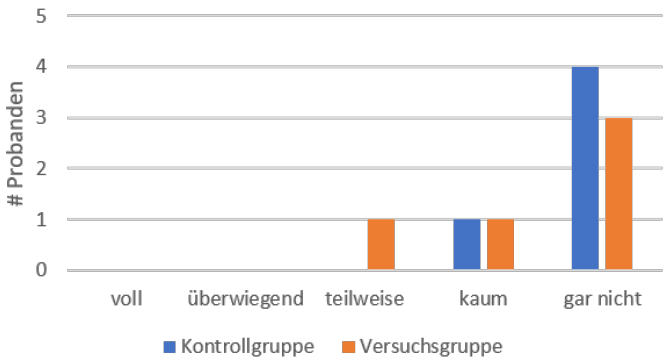


Abbildung 5.13.: Zustimmung zu Aussage 3 in Tabellen- und Diagrammform

Aussage 4:
Eine Verknüpfung von Use Cases und Videos auf Basis eine Ontologie ist sinnvoll.
Die vierte und letzte Aussage erreichte bei der Kontrollgruppe mit dreimal teilweise und zweimal überwiegend eine prozentuale Zustimmung von 68%. Die Versuchsgruppe stimmte hingegen zu 80% zu, wobei dreimal volle Zustimmung und je einmal überwiegende und gar keine erteilt wurden. Insgesamt stimmten die Probanden beider Gruppen damit zu 74% zu. Statistisch signifikant waren die Ergebnisse beider Gruppen auch hier nicht. Der T-Test ergab mit $t = 0,73855$ und $p = 0,481284 > 0,05$, dass das Ergebnis ein statistisches Rauschen und keinen signifikanten Unterschied darstellt. Der Median der Kontrollgruppe ergab dabei einen Wert von 3, also teilweiser Zustimmung, wohingegen der der Versuchsgruppe mit dem Zahlenwert 5 die volle Zustimmung erreichte.

Zustimmung		Kontroll-gruppe	Versuchs-gruppe
Option	Wert		
voll	5	0	3
überwiegend	4	2	1
teilweise	3	3	0
kaum	2	0	0
gar nicht	1	0	1
Median		3	5

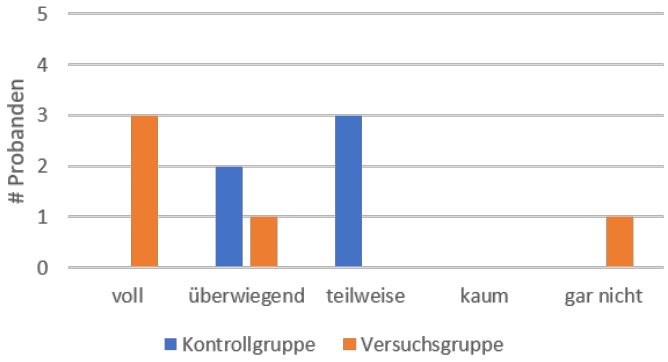


Abbildung 5.14.: Zustimmung zu Aussage 4 in Tabellen- und Diagrammform

Beobachtungen während der Durchführung

Während der Durchführung wurden einige Aspekte ersichtlich, die man in zukünftigen Experimenten zur Nutzung von annotierten Videos berücksichtigen sollte. Zunächst existieren sehr unterschiedliche Herangehensweisen an die Beantwortung der Fragen zu den vorgelegten Use Cases. Manche Probanden nahmen sich zuerst einmal die Zeit, den Use Case gründlich durchzulesen, andere haben direkt das jeweilige Video angesehen, ohne zuvor ein schriftliches Dokument genauer zu betrachten. In Einzelfällen konnte beobachtet werden, dass bereits Fragen ohne Videosichtung beantwortet wurden. Die vorliegende Ontologie-Instanz in ausgedruckter Form war den wenigsten Probanden mehr als einen Blick wert, so dass Fragen nur anhand des Videos und des Use Cases beantwortet wurden. Die zusätzlichen Anwahlmöglichkeiten von annotierten Inhalten, die ObViViAn bietet wurden eher verhalten verwendet.

5.4.2. Interpretation der Ergebnisse

In diesem Teil des Kapitels werden die Ergebnisse des vorangegangenen Abschnittes bewertet und interpretiert. Eine Beantwortung der Forschungsfragen auf Basis der Ergebnisse wird präsentiert und eine statistische Auswertung hinsichtlich ihrer Signifikanz gegeben. Zum Schluss werden die Bedrohungen der Validität erläutert, die zu diesem Experiment identifiziert werden konnten und die nicht statistisch erfassten Beobachtungen, die während der Versuchsdurchführung gemacht wurden präsentiert.

Beantwortung der Forschungsfragen

Forschungsfrage I: *Sind Entwickler mit einer Ontologie-basierten Annotation von Videoinhalten schneller dazu in der Lage, anhand annotierter Videos Fragen richtig zu beantworten als mit nicht annotierten Videos?*

Der statistische Effekt, der bei den gemessenen Zeiten zur Beantwortung von Fragen zu Use Cases errechnet wurde war nur bei zwei von drei Messreihen als signifikant zu bezeichnen. Bei der letzten Messreihe konnte mit $p = 0,202532 > 0,05$ keine Signifikanz mehr festgestellt werden. Zwischen den Ergebnissen des Signifikanztests der ersten beiden Messungen bestand ebenfalls schon ein markanter Unterschied in Bezug auf den p-Wert, dieser war bei Durchgang 2 mit $p = 0,032073$ bereits deutlich näher an der Signifikanzgrenze von 0,05, als es bei Durchgang 1 mit $p = 0,009707$ der Fall war. Insgesamt waren die Teilnehmer der Versuchsgruppe bei jedem Durchgang schneller als die der Kontrollgruppe, da die Eindeutigkeit der statistischen Signifikanz allerdings von der ersten bis zur dritten Messung bis hin zur nicht mehr nachzuweisenden abnahm, kann die Forschungsfrage damit beantwortet werden, dass die Nutzung von Ontologie-basiert annotierten Videos für den Entwickler keinen Unterschied dazu ergibt, nur das Video mit einem herkömmlichen Video-Player anzusehen.

Forschungsfrage II: *Sind Entwickler mit einer Ontologie-basierten Annotation von Videoinhalten dazu in der Lage mehr Fragen richtig zu beantworten als mit nicht annotierten Videos?*

Die ermittelten Ergebnisse lassen nicht darauf schließen, dass mit einer Unterstützung durch annotierte Videos mehr Fragen zu Use Cases richtig beantwortet werden können als mit nicht annotierten. Obwohl die Kontrollgruppe insgesamt mit nur einer falschen im Gegensatz zur Versuchsgruppe mit 3 mehr richtige Antworten gegeben hat, ist das Ergebnis des Vergleichs nicht statistisch signifikant. Mit einer Unterstützung durch Ontologie-basiert annotierte Videos lassen sich daher basierend auf diesem Experiment nicht bedeutend mehr oder weniger korrekte Antworten auf bestehende Fragen geben als mit nicht annotierten Videos.

Schlussfolgerung bezüglich der angestrebten Ziele

Aus der Beantwortung der Forschungsfragen des vorherigen Abschnittes ergeben sich für die in dieser Arbeit verfolgten Ziele mehrere Schlussfolgerungen. Zunächst einmal wird anhand der Zustimmung zur Aussage “Die Videos haben geholfen, die Inhalte der Use Cases besser zu verstehen.“ deutlich, dass Entwickler einen Mehrwert von Vision Videos als Dokumentationsartefakt sehen. Ebenso deutet sich an, dass der Aufwand dazu, Videos als Dokumente heranzuziehen den Entwicklern nicht zu groß ist, so dass ihr Einbezug in die Dokumentation grundsätzlich begrüßenswert erscheint. Der Ontologie-Bezug der grafischen Annotation von Inhalten aus solchen Videos hingegen ist für Menschen offensichtlich eher zu vernachlässigen, da die Zustimmung zur Aussage “Die Videos waren über die Ontologie zugänglicher.“ gerade etwas mehr als die Hälfte betrug. Als hilfreich für das Verständnis von Use Cases und zur Beseitigung von Klärungsbedarf werden Ontologien als Basis für einen Annotation also nicht angesehen. Die fast drei Viertel betragende Zustimmung zur Aussage “Eine Verknüpfung von Use Cases und Videos auf Basis einer Ontologie ist sinnvoll.“ hingegen lässt darauf schließen, dass die Probanden durchaus den Zweck der Ontologie-Basis verstanden haben und diese begrüßen, obwohl sie ihnen nicht direkt bei der Lösung der Aufgaben geholfen hat.

Bezüglich der Zeit, die für eine Klärung von Verständnisschwierigkeiten anhand von Videos benötigt wird kann hier schlussgefolgert werden, dass jedes Heranziehen eines zusätzlichen Artefaktes innerhalb der Dokumentation mit einer Betrachtungszeit einher geht. Dokumente zu lesen, anzuschauen oder darin zu blättern beziehungsweise zu spulen dauert schlicht und ergreifend seine Zeit. Unter Zeitdruck ein neues Tool gezielt zu verwenden ist trotz einer kurzen Einführung grundsätzlich schwierig. Man kann an den gemessenen Zeiten jedoch erkennen, dass ein Lerneffekt beim Ansehen der Videos aufgetreten ist, denn die durchschnittlich benötigte Zeit sank sowohl bei allen Teilnehmern insgesamt von 4:37 Minuten beim ersten Use Case über 3:25 Minuten beim zweiten zu 2:48 beim letzten. Dieser Trend ist ebenso für jede Gruppe einzeln zu beobachten gewesen. So benötigte die Kontrollgruppe für die Beantwortung der Fragen zu Use Case 1 noch 3:11 Minuten, für die zu Use Case 2 dann 2:36 Minuten und bei Use Case 3 waren es nur noch 2:21. Für die Versuchsgruppe sank die der Mittelwert von 6:03 Minuten über 4:14 Minuten auf 3:16 Minuten.

Was die Anzahl korrekt beantworteter Fragen angeht, so hat die Eindeutigkeit der Darstellung von dafür relevanten Inhalten im Video und die Eindeutigkeit des begriffliche Bezugs zu ihnen darauf Auswirkung. Zwei von insgesamt neun Fragen wurden überhaupt falsch beantwortet. Eine davon dreimal, die andere einmal. Bei der dreimal falsch beantworteten Frage ging es um die örtliche Beziehung einer nicht menschlichen Entität zu einer anderen. In der Ontologie-Instanz, die den Probanden vorlag wurde die Beziehung nicht örtlich erwähnt, wohl aber eine Oberklasse davon. Eine der beiden Entitäten war bei der Versuchsgruppe annotiert und ihre Begriffsbezeichnung eindeutig in der Tabelle von Entitäten zu erkennen. Im Video

war zusätzlich eine akkustische Erwähnung zu hören, trotzdem beantworteten zwei Personen aus der Versuchsgruppe und eine der Kontrollgruppe die Frage falsch, was höchst wahrscheinlich mit dem Zeitdruck zu erklären ist.

In Bezug zu den verfolgten Zielen des Experiments kann der Schluss folgen, dass der Wiederbetrachtungswert über einen Ontologie-Bezug und die damit verbundene Steigerung der Zugänglichkeit prinzipiell erreicht werden kann. Dasselbe gilt für die Dokumentierbarkeit von Videoinhalten, denn diese können mit den Annotationen eindeutig Begriffen zugeordnet werden, die in einer übergeordneten Domäne verwendet werden.

Abschließen kann konstatiert werden, dass sich eine Weiterverfolgung des Ontologie-basierten Ansatzes zur Annotation von Videos und die damit verbundene Möglichkeit, Videoinhalte maschinenlesbar in die Dokumentation einzubeziehen lohnen kann.

5.4.3. Bedrohungen der Validität

Echtheit der verwendeten Vision Videos

Da es noch keine Definition für Vision Videos gibt, mussten Videos verwendet werden, die für jeden Probanden einen Visionscharakter beinhalteten. Ob die verwendeten Videos wirklich vor Implementierung des Systems zur Sicherung einer Cockpit-Tür entworfen wurden und zur Ermittlung einer gemeinsamen Wissensbasis von Auftraggeber und Entwicklerteam genutzt wurden, war zum Zeitpunkt der Arbeit nicht ermittelbar. Da keine weiteren, öffentlich zugänglichen Informationen zu diesem System zur Verfügung standen, wurden die Videos als Darstellungen von Visionen für die Nutzung in einer Dokumentation benutzt.

Länge der verwendeten Videos

Da die Videos unterschiedliche Spieldauern besitzen, hat diese Dauer durchaus Einfluss auf die Zeit, welche ein Proband zur Beantwortung der Fragen benötigte. Da diese allerdings separat und somit für alle Probanden beider Gruppen gleich waren, konnte dieser Bedrohung aus dem Weg gegangen werden.

Unterschiedliche Präferenzen der Probanden zu Arten von Dokumenten

Da es durchaus denkbar war, dass die Probanden unterschiedlich viel Zeit für das Lesen der Use Case Tabelle, der Ontologie-Instanz in Diagrammform und dem Ansehen des Videos zubringen würden, ist diese Diskrepanz ebenfalls eine Bedrohung der Validität des Experiments. Zeitlich eindeutiger einzuschränken war der Versuchsablauf allerdings nicht, da das Ziel des Experiments Videos und einen Ontologie-Bezug im Zusammenspiel mit herkömmlichen Dokumentationsartefakten beinhaltete. Es sollten Videos und auf Ontologie-Basis annotierte Videos als Ergänzung zur schriftlichen

Dokumentation von Anforderungen betrachtet werden, weshalb dieser Bedrohung bei Messungen diesen Ausmaßes nicht ohne weiteres begegnet werden konnte.

6. Zusammenfassung und Ausblick

Dieses Kapitel dient dazu, einen abschließenden Überblick über die Inhalte und die wichtigsten Ergebnisse der Arbeit zu geben.

6.1. Fazit

Die Methode, Vision Videos im Requirements Engineering zu verwenden besitzt zur Zeit neben den klaren Vorteilen noch einige Nachteile. So ist es von Vorteil anhand einer visuellen und deutlichen Darstellung ergänzend zu anderen Elicitation-Techniken mit dem Kunden zu klären, wie eine gemeinsame Vorstellung auszusehen hat. Videos können immer wieder neu gedreht und dann gesichtet werden. Im Gegensatz zur Implementierung eines Prototypen kann dies Zeit sparen, benötigt aber trotzdem zusätzliche Zeit. Produktionskosten sind ein weiterer Nachteil, so dass ein erweiterter Einbezug dieser Videos in die Softwareentwicklung sinnvoll und lohnenswert erscheint.

Der in dieser Arbeit verfolgte Ansatz soll die Inhalte von Vision Videos auf eindeutige und für Menschen, sowie Software lesbare Art in die Dokumentation einbeziehen. Das zentrale, konkrete Ziel dieser Arbeit war es, Videoinhalte grafisch annotierbar zu machen und über eine Ontologie die Möglichkeit zur klaren Einordnung in eine oder mehrere Domänen zu schaffen.

Einen Ansatz zur Umsetzung dessen zu entwickeln war der nächste Schritt. Dabei mussten zuerst die Anforderungen an eine Ontologie-basierte Annotationssoftware feiner ausgearbeitet werden. Es kristallisierten sich dabei verschiedene Aspekte heraus, die Videoinhalte besitzen können und die für eine Annotation zu berücksichtigen waren.

Die prototypische Umsetzung entstand anhand der erarbeiteten Aspekte der Darstellungen von Inhalten in Videos und anhand des geforderten Ontologie-Bezugs in der Web Ontology Language. Eine evaluierbare Version konnte erstellt werden.

In der Evaluation wurden zwei Gruppen von Teilnehmern verglichen, wobei jeder Teilnehmer Fragen zu Use Cases zu beantworten hatte. Er konnte diese mit Hilfe von Videos und einem ausgedruckten Instanz-Diagramm einer Ontologie klären. Eine Gruppe bekam annotierte Videos innerhalb des Prototypen, während die andere nur einen herkömmlichen Video Player zur Verfügung hatte. Es wurde ebenfalls die Zeit gestoppt, die pro Teilnehmer für die Beantwortung gebraucht wurde. Zusätzlich

wurden zum Ende der Durchführung hin jedem Probanden Aussagen über Videos als Dokumentationsoption und eine Verknüpfung über Ontologien präsentiert, denen er seine Zustimmung auf einer Skala von 1 (gar nicht) bis 5 (voll) geben konnte. Dabei kam heraus, dass Videos als alternative und visuelle Darstellung von Anforderungen von den Probanden akzeptiert wird. Der Bezug zu einer Ontologie hingegen wurde weniger akzeptiert, er richtete sich allerdings auch vornehmlich an maschinelle Adressaten.

Die Evaluation ergab, dass mit zunehmender Nutzungsdauer alle Probanden schneller mit Videos als Dokumentationsartefakten umgingen. Egal, ob sie mit oder ohne Annotation arbeiteten steigerte sich ihre Geschwindigkeit beim Antworten auf die gestellten Fragen. Die Nutzung einer neuen Software scheint allerdings die Geschwindigkeit zum Lösen einer Aufgabe anfangs eher zu hemmen, denn die Kontrollgruppe ohne annotierte Videos blieb in jedem Durchgang zeitlich gesehen vor der anderen Gruppe. Der Abstand sank allerdings prozentual gesehen. Auch wurden insgesamt sehr wenige falsche Antworten gegeben, was Videos als Darstellungsoption von Anforderungen durchaus eine Berechtigung zuspricht.

Ebenfalls kann festgehalten werden, dass unabhängig von den Ergebnissen der Evaluation eine Ontologie-bezogenen Annotation von Videoinhalten möglich ist. Ein Ansatz zur Verbindung von grafischen Annotationen mit Begriffen und deren Beziehungen aus einer Ontologie konnte entwickelt und evaluiert werden. Nicht alle Arten der Darstellung von Inhalten in Videos sind geeignet um grafisch annotiert zu werden. So sind jegliche akustische Inhalte nicht eindeutig zu markieren, bis auf die Ausnahme einer akustischen Interaktion zwischen zwei annotierten Entitäten.

Insgesamt stellt diese Arbeit einen ersten Ansatz dazu dar, Vision Videos gezielt für eine weitere Nutzung über die Requirements Elicitation hinaus in der Softwareentwicklung zu nutzen.

6.2. Ausblick

Ausblickend auf Basis der erarbeiteten Grundlagen, dem entwickelten Prototypen und den Ergebnissen des durchgeführten Experiments ergeben sich einige Themengebiete, denen bei Interesse weiter nachzugehen wäre. Zuerst sollte man sich weiter und tiefer gehend mit der für eine Dokumentation geeigneten Beschaffenheit von Vision Videos beschäftigen. Wichtig ist dabei vor allem herauszufinden, welche Grundregeln für die Herstellung von Videos gelten müssen, die man sowohl für Klärungsbedarf mit dem Kunden, als auch zur Veranschaulichung von ansonsten rein textueller Dokumentation benutzen möchte. Gefilmte Papierprototypen sind ein interessantes Beispiel, da sie sich durchaus eignen um auf Basis einer Domänen-Ontologie annotiert zu werden. Allerdings ist diese Art der Darstellung möglicherweise zu abstrakt, um schnell und einfach Kunden- und Entwicklerfragen klären zu können. Auch der Einsatz

von eingeblendetem Text und Kommentaren oder Musik auf der Tonspur sind zu untersuchen.

Audiokommentare können sehr hilfreich für das Verständnis des Videoinhaltes, zum Beispiel eines gezeigten Szenarios sein. Aus der Sicht eines annotierenden Requirements Engineers allerdings besteht aber häufig dann ein Problem, wenn sie Entitäten oder Interaktionen ersetzen, die man eigentlich sehen müsste. Ebenfalls problematisch wird es, wenn eine Interaktion eine auditive ist, also beispielsweise ein Anwender mit einem Software-System über eine Spracheingabe interagiert. Diese Interaktion würde mit einer grafischen Annotation zwar sichtbar, deren Inhalt aber nur mit angeschaltetem Ton und vor allem mit einem Hinweis auf die Wichtigkeit der Tonspur.

Man kann zwar eingeblendeten Text ohne weiteres mit einer einfachen grafischen Annotation versehen, stattdessen müsste man eine spezielle Art der Annotation in den Prototypen implementieren, die auf visuelle Art verdeutlicht, dass an der entsprechenden Stelle im Video nicht das genaue Hinsehen, sondern auch das Zuhören von großer Wichtigkeit ist. Für Texteinblendungen können ebenfalls dem Verständnis dienen und man kann sie grafisch annotieren. Deren Nachteil kommt aber dann zum Tragen, wenn statt einer Entität oder Interaktion im Video zu sehen sind und man damit wieder in die textuelle Form verfällt. Lesen, was eine Anforderung an Informationen enthält und welche Voraussetzungen dafür notwendig sind kann man ja bereits in den traditionellen Formen der Dokumentation.

Dann besteht noch grundsätzlicher Untersuchungsbedarf bei der Verknüpfung von Videos mit Anforderungen und weiteren Artefakten der Softwareentwicklung. Eine Projekt-Ontologie aufzubauen, die das gesamte Wissen über das Projekt sinnvoll miteinander verknüpft und damit Kundenwissen mit dem Entwicklerwissen verbindet, kann die Dokumentation unterstützen. In Verbindung mit auf Basis der Ontologie annotierten Videos kann der Wiederanschauungswert der Videos erhöht und deren Inhalte im Projekt weiterführend genutzt werden. Es ist also sinnvoll zu untersuchen, welche Kontexte eine Ontologie verbinden sollte um die Dokumentation zu verbessern und Videos als Artefakte zu etablieren. Dazu könnte man verschiedene Kontexte definieren und enthaltene Entitäten und Interaktionen mit Referenzen, also OWL-Objekteigenschaften oder -Dateneingenschaften versehen. Beispielsweise könnte so eine Verknüpfung entstehen, die zwei aufeinander folgende Schritte aus einem Use Case mit einem Video verbindet. Genauer gesagt mit den darin grafisch annotierten Systemkomponenten und Akteuren. Weiter führend könnte man sich überlegen, ob man über diese Ontologie nicht auch Testfälle verknüpft für den Fall, dass auf Testebene Klärungsbedarf besteht. So hätte man dann die Möglichkeit, sich für ein Dokument oder Artefakt die relevanten anderen abzurufen und diese dann noch einmal anzuschauen und gegebenenfalls auch zu korrigieren.

A. Dokumente zur Evaluation

In diesem Anhang befinden sich die Dokumente, die den Teilnehmern der Evaluation vorgelegt wurden in der folgenden Reihenfolge:

1. die einführende Erläuterung zum Experiment
2. die Fragen zu den einzelnen Use Cases
3. die Ontologie-Instanz zu den Use Cases und Videos
4. die drei Use Cases

Nehmen Sie an, Sie sind Entwickler für ein System zur Sicherung der Cockpittür eines Airbus A320. Dabei gibt es drei verschiedene Prozeduren, die in 3 verschiedenen Situationen ausgelöst werden können. Jede Situation ist in einem Use Case erfasst, allerdings auch in einem Vision Video dargestellt, das Sie sich anschauen können um den Use Case besser zu verstehen.

Im Folgenden bekommen Sie je drei Fragen zu jedem der drei Use Cases gestellt, für deren Beantwortung das Anschauen des Videos von Vorteil sein könnte. Für jeden Use Case wird die Zeit gemessen, die Sie benötigen um die Fragen zu beantworten. Zur Verfügung steht Ihnen dabei Wissen über die Domäne in Form einer Ontologie. Diese enthält schemenhaft Informationen zur Übersicht über in den Szenarien vorkommende Entitäten (Personen und Objekte) und Interaktionen zwischen Entitäten. Zur Beantwortung der Fragen liegen als Dokumente also die Ontologie mit dem Grundwissen vor, sowie jeweils der Use Case und das zugehörige Vision Video.

Jeder Use Case wird für sich gemessen, daher sagen Sie bitte klar wahrnehmbar bescheid, sobald Sie die Fragen beantwortet haben.

Use Case 01 - Normale Zugangsprozedur

Frage 1 In welche Richtung muss der Pilot den **ToggleSwitch** bewegen damit die Tür entriegelt wird?

- ☐ links
- ☐ rechts
- ✓ oben
- ☐ unten

Frage 2 Wo am **ControlPanel** befindet sich das **OpenLight**?

- ✓ links neben dem ToggleSwitch
- ☐ rechts neben dem ToggleSwitch
- ☐ über dem ToggleSwitch
- ☐ unter dem ToggleSwitch

Frage 3 Wozu dient der **Buzzer**?

- ✓ um eine Zutrittsanfrage zu signalisieren
- ☐ als Warnung für einen unbefugten Zutrittsversuch
- ☐ als Rufton des Bordtelefons
- ☐ als akkustisches Feedback für die Eingabe des Sicherheitscodes

Use Case 02 - Verriegelungsprozedur

Frage 1 An welcher Stelle auf dem **CodePad** befindet sich das **RedLight** zur Anzeige einer verriegelten Cockpittür?

- ✓ oben links
- ☐ oben rechts
- ☐ unten links
- ☐ unten rechts

Frage 2 Wie erreicht die Stewardess, dass die Türverriegelung wieder aufgehoben wird?

- ☐ Sie klopft noch einmal
- ☐ Sie tritt gegen die Tür
- ✓ Sie ruft im Cockpit an (*korrekt*)
- ☐ Sie muss 5 Minuten warten und dann erneut die Normale Zugangsprozedur auslösen

Frage 3 Was ist die beispielhafte Situation, die ein unbefugtes Betreten darstellt?

- ☐ Jemand gibt einen falschen Sicherheitscode ein
- ☐ Jemand versucht, die Tür manuell zu öffnen
- ✓ Jemand klopft an die Tür, damit der Pilot die Tür öffnet
- ☐ Jemand unbefugtes nutzt das Bordtelefon und ruft im Cockpit an

Use Case 03 - Notfall-Zugangsprozedur

Frage 1 Warum reagiert niemand aus dem Cockpit auf den Auslöser der normalen Zugangsprozedur?

- ☐ Die Piloten hören Musik über Kopfhörer
- ☐ Die Piloten spielen an ihren Smartphones
- ☐ Der Fluglärm ist zu laut um den Buzzer hören zu können.
- ✓ Die Piloten sind bewusstlos

Frage 2 Wann realisiert die Stewardess, dass es sich um einen Notfall handeln muss?

- ☐ Nach dem ersten Drücken des **HashKey**
- ✓ Nach dem zweiten Versuch im Cockpit anzurufen
- ☐ Nach dem ersten Versuch im Cockpit anzurufen
- ☐ Nachdem sie geklopft hat und die Tür nicht verriegelt wurde

Frage 3 Aus wie vielen Ziffern besteht der Sicherheitscode, den die Stewardess eingibt bevor sie den **HashKey** drückt?

- ☐ 3
- ☐ 5
- ✓ 4
- ☐ 6

Fragen zum Dokumenten und Verfahren

In diesem Abschnitt geht es um Ihre Meinung zu den gegebenen Verfahren und Dokumenten. Kreuzen Sie bitte nur eine auf Ihre Meinung zutreffende Option an.

Die Videos haben geholfen, die Inhalte der Use Cases besser zu verstehen.

stimme voll zu		stimme teilweise zu		stimme nicht zu
☐	☐	☐	☐	☐

Die Videos waren über die Ontologie zugänglicher.

stimme voll zu		stimme teilweise zu		stimme nicht zu
☐	☐	☐	☐	☐

Der Aufwand die Videos einzeln anzuschauen war zu groß.

stimme voll zu		stimme teilweise zu		stimme nicht zu
☐	☐	☐	☐	☐

Eine Verknüpfung von Use Cases und Videos auf Basis einer Ontologie ist sinnvoll.

stimme voll zu		stimme teilweise zu		stimme nicht zu
☐	☐	☐	☐	☐

Angaben zur Person

Vorwissen: Welche der angegebenen Veranstaltungen haben Sie an der Leibniz Universität Hannover gehört? (Mehrfachnennungen möglich)

- ☐ Softwaretechnik
- ☐ Requirements Engineering
- ☐ Softwarequalität

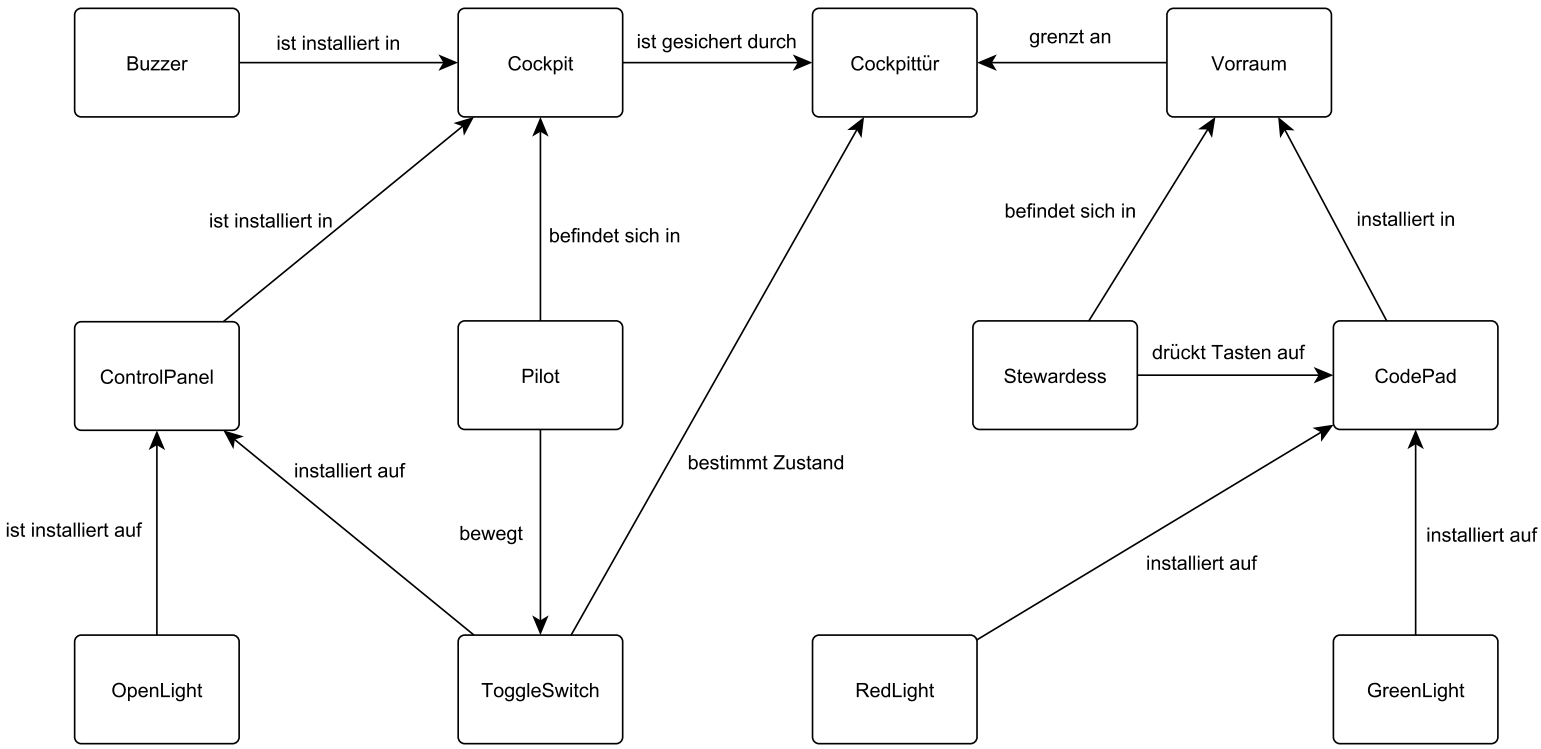
Bildungsgrad: Welches ist Ihr höchster Bildungsgrad / Abschluss?

- ☐ Allgemeine Hochschulreife
- ☐ Diplom
- ☐ Bachelor
- ☐ Master
- ☐ Doktorgrad

Fachrichtung: In welchem Fachgebiet haben Sie den Bildungsgrad erreicht?

- ☐ Informatik
- ☐ anderes Fachgebiet: _____
- ☐ Mathematik
- ☐ Wirtschaft

Vielen Dank für die Teilnahme!


Buzzer
ist installiert in
Cockpit
ist gesichert durch
Cockpittür
grenzt an
Vorraum
ist installiert in
befindet sich in
befindet sich in
installiert in
ControlPanel
Pilot
Stewardess
drückt Tasten auf
CodePad
installiert auf
bestimmt Zustand
ist installiert auf
bewegt
installiert auf
installiert auf
OpenLight
ToggleSwitch
RedLight
GreenLight

UC 01	**Normale Zugangsprozedur**
Umfeld	Cockpit und Vorraum eines Airbus A320
Systemgrenzen	**ControlPanel** der Cockpittür im Cockpit, Anzeige des Zustands im Vorraum durch **GreenLight** oder **RedLight**. Bordtelefon ist nicht Teil des Systems.
Ebene	Hauptaufgabe, Öffnen der Tür
Hauptakteure	Pilot
Stakeholder u. Interessen	Pilot: Möchte Stewardess Einlass ins Cockpit gewähren. Stewardess: Möchte ins Cockpit gelangen.
Voraussetzungen	Die Cockpittür befindet sich im normalen Zustand, der **ToggleSwitch** ist in der Position "NORMAL"
Garantie	Die Tür kann nur vom Vorraum aus geöffnet werden, wenn der Pilot den **ToggleSwitch** in die Position "UNLOCK" bewegt oder die Notfall-Zugangsprozedur ausgelöst wird.
Erfolgsfall	Die Cockpittür ist entriegelt und die Stewardess kann sie öffnen um einzutreten.
Auslöser	Die Stewardess betätigt den **HashKey** auf dem **CodePad** um zu signalisieren, dass sie das Cockpit betreten möchte.
Beschreibung	1 Die Stewardess betätigt den **HashKey** auf dem **CodePad** 2 Im Cockpit ertönt der **Buzzer** für 3 Sekunden 3 Der Pilot schalten den **ToggleSwitch** des **ControlPanels** in die Position "UNLOCK". 4 Das **GreenLight** auf dem **CodePad** im Cockpitvorraum leuchtet auf, die Tür wird entriegelt. 5 Die Stewardess öffnet die Tür und tritt ein, auf dem **ControlPanel** im Cockpit leuchtet das **OpenLight** solange der **ToggleSwitch** in der Position "UNLOCK" gehalten wird.
Erweiterungen	keine Erweiterungen
Technologie	keine technische Variation

UC 02	**Verriegelungsprozedur**	
Umfeld	Cockpit und Vorraum eines Airbus A320	
Systemgrenzen	**ControlPanel** der Cockpittür im Cockpit, Anzeige des Zustands im Vorraum **GreenLight** oder **RedLight**. Bordtelefon ist nicht Teil des Systems.	
Ebene	Hauptaufgabe, Verriegelung der Tür in potenzieller Gefahrensitutation	
Hauptakteure	Pilot	
Stakeholder u. Interessen	Pilot	Möchte die Cockpittür verriegeln um unbefugtes Betreten zu verhindern
	Stewardess	Möchte ins Cockpit gehen um zu fragen, ob alles in Ordnung ist.
Voraussetzungen	Die Stewardess hat an die Cockpittür geklopft und sich damit nicht an die Normale Zugangsprozedur gehalten um das Cockpit betreten zu dürfen. Der Pilot vermutet eine Gefahrensituation.	
Garantie	Die Tür kann nur vom Vorraum aus geöffnet werden, wenn der Pilot den **ToggleSwitch** in die Position "UNLOCK" bewegt oder die Notfall-Zugangsprozedur ausgelöst wird.	
Erfolgsfall	Die Cockpittür befindet sich im Zustand "LOCK" und **CodePad**, **Buzzer** sowie die automatische Türöffnung sind für 5 Minuten blockiert.	
Auslöser	Der Pilot schaltet das **ToggleSwitch** des **ControlPanels** in die Position "LOCK"	
Beschreibung	1	Der Pilot schaltet das **ToggleSwitch** des **ControlPanels** in die Position "LOCK".
	2	Die Cockpittür geht in den verriegelten Zustand über und **CodePad**, **Buzzer** sowie die automatische Türöffnung sind für 5 Minuten blockiert.
Erweiterungen	2a	WENN verifiziert wird, dass keine Gefahrensitutation besteht, DANN schaltet der Pilot den **ToggleSwitch** in die Position "UNLOCK", was die fünfminütige Blockierung von **CodePad**, **Buzzer** und automatischer Türöffnung überschreibt und sofort die Tür entriegelt.
Technologie	keine technische Variation	

UC 03	**Notfall-Zugangsprozedur**
Umfeld	Cockpit und Vorraum eines Airbus A320
Systemgrenzen	**ControlPanel** der Cockpittür im Cockpit, Anzeige des Zustands im Vorraum **GreenLight** oder **RedLight**. Bordtelefon ist nicht Teil des Systems.
Ebene	Hauptaufgabe, Öffnen der Tür im Notfall vom Vorraum aus.
Hauptakteure	Stewardess
Stakeholder u. Interessen	Stewardess — Möchte ins Cockpit um nachzusehen, warum niemand auf ihre Anfragen reagiert. Sie geht von einem Notfall aus.
Voraussetzungen	Die Stewardess hat keine Reaktion aus dem Cockpit erhalten und geht von einer Notfallsituation aus. Die Stewardess kennt den Sicherheitscode für den Notfallzugang.
Garantie	Kein Unbefugter kann das Cockpit betreten. Die Stewardess kann vom Vorraum aus die Tür per Eingabe eines Sicherheitscodes am **ControlPanel** öffnen, wenn im Cockpit niemand auf die normale Anfrageprozedur reagiert.
Erfolgsfall	Die Cockpittür lässt sich 5 Sekunden lang vom Vorraum aus manuell öffnen.
Auslöser	Die Stewardess gibt auf dem **CodePad** den Sicherheitscode ein und drückt danach den **HashKey**.
Beschreibung	1 Die Stewardess gibt auf dem **CodePad** den Sicherheitscode ein und drückt danach den **HashKey**. 2 Der **Timer** wird ausgelöst und läuft für 30 Sekunden. Das **GreenLight** auf dem **CodePad** blinkt, im Cockpit ertönt der **Buzzer** durchgehend, und das **OpenLight** am **ControlPanel** blinkt und zeigt so an, dass die Tür in Kürze geöffnet wird. Nach Ablauf des **Timers** wird die Tür für 5 Sekunden entriegelt. Währenddessen leuchtet das **GreenLight** auf dem **CodePad** dauerhaft auf, sowie das **OpenLight** im Cockpit. 3 Innerhalb der 5 Sekunden Entriegelungszeit öffnet die Stewardess die Tür und betritt das Cockpit.
Erweiterungen	keine Erweiterungen
Technologie	keine technische Variation

B. Inhalte der CD

In diesem Anhang befinden sich eine Auflistung und kurze Beschreibung der Inhalte der beiliegenden CD.

1. die prototypische Implementierung *ObViViAn* als gepacktes exportiertes Eclipse-Projekt
2. die Arbeit selbst im *Portable Document Format (PDF)*
3. die geschnittenen Videos zu den Use Cases der Evaluation
4. das ungeschnittene, ursprüngliche Video zu den Use Cases der Evaluation
5. die **.ecore*-Datei, die den Dokumenten der Evaluation zugrunde lag
6. die **.collaboration*-Datei, die ebenfalls den Dokumenten der Evaluation zugrunde lag
7. die Dokumente zur Evaluation (Anhang A) im *Portable Document Format (PDF)*

Literaturverzeichnis

[1] S. Ambler. *Agile modeling: effective practices for extreme programming and the unified process*. John Wiley & Sons, 2002.

[2] G. Antoniou and F. van Harmelen. *Web Ontology Language: OWL*, pages 67–92. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[3] L. Bai, S. Lao, G. J. Jones, and A. F. Smeaton. Video semantic content analysis based on ontology. In *Machine Vision and Image Processing Conference, 2007. IMVIP 2007. International*, pages 117–124. IEEE, 2007.

[4] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, and G. Caorthers. `https://www.w3.org/TR/turtle/`. letzter Zugriff 10.10.2017.

[5] T. Berners-Lee, J. Hendler, O. Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.

[6] S. Bloehdorn, K. Petridis, C. Saathoff, N. Simou, V. Tzouvaras, Y. S. Avrithis, S. Handschuh, I. Kompatsiaris, S. Staab, and M. G. Strintzis. Semantic annotation of images and videos for multimedia analysis. In *ESWC*, volume 3532, pages 592–607. Springer, 2005.

[7] E. Börger, B. Hörger, D. Parnas, and H. Rombach. Requirements capture, documentation, and validation. In *Dagstuhl Seminar*, 1999.

[8] D. Brickley and G. R. V. Rdf vocabulary description language: Rdf schema. `https://www.w3.org/TR/2014/REC-rdf-schema-20140225/`. letzter Zugriff 08.10.2017.

[9] O. Brill, K. Schneider, and E. Knauss. Videos vs. use cases: Can videos capture more requirements under time pressure? In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 30–44. Springer, 2010.

[10] A. Cockburn. Writing effective use cases, the crystal collection for software professionals. *Addison-Wesley Professional Reading*, 2000.

[11] O. Creighton, M. Ott, and B. Bruegge. Software cinema-video-based requirements engineering. In *14th IEEE International Requirements Engineering Conference (RE'06)*, pages 109–118, Sept 2006.

[12] T. Doran. Iso/iec/ieee international standard - systems and software engineering – life cycle processes –requirements engineering. *ISO/IEC/IEEE 29148:2011(E)*, pages 1–94, Dec 2011.

[13] H. Dubberly. Apple future shock: The knowledge navigator. `https://www.youtube.com/watch?v=3WdS4TscWH8`. letzter Zugriff 15.10.2017.

[14] C. Ebert. *Systematisches Requirements Engineering: Anforderungen ermitteln, dokumentieren, analysieren und verwalten*. dpunkt. verlag, 2014.

[15] D. Fanaie Gahnavie. Unterstützung bei der erstellung und integration von videos in software-spezifikationen. Masterarbeit, Leibniz Universität Hannover, 2016.

[16] G. Fischer. Symmetry of ignorance, social creativity, and meta-design. *Knowledge-Based Systems*, 13(7):527–537, 2000.

[17] T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5-6):907–928, 1995.

[18] N. Guarino et al. Formal ontology and information systems. In *Proceedings of FOIS*, pages 81–97, 1998.

[19] J.-W. Jeong, H.-K. Hong, and D.-H. Lee. Ontology-based automatic video annotation technique in smart tv environment. *IEEE Transactions on Consumer Electronics*, 57(4), 2011.

[20] O. Karras, S. Kiesling, and K. Schneider. Supporting requirements elicitation by tool-supported video analysis. In *Requirements Engineering Conference (RE), 2016 IEEE 24th International*, pages 146–155. IEEE, 2016.

[21] O. Karras, J. Klünder, and K. Schneider. Enrichment of requirements specifications with videos. *Zenodo*, 2016.

[22] H. Knublauch. Ontology-driven software development in the context of the semantic web: An example scenario with protege/owl. In *1st International workshop on the model-driven semantic web (MDSW2004)*, pages 381–401. Monterey, California, USA.[WWW document] http://www. knublauch. com/publications/MDSW2004. pdf, 2004.

[23] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen. The protégé owl plugin: An open development environment for semantic web applications. In *International Semantic Web Conference*, volume 3298, pages 229–243. Springer, 2004.

[24] H. Knublauch, M. Horridge, M. A. Musen, A. L. Rector, R. Stevens, N. Drummond, P. W. Lord, N. F. Noy, J. Seidenberg, and H. Wang. The protege owl experience. In *OWLED*, 2005.

[25] M. A. Musen, T. Tudorache, and S. Tu. protégé. `https://protege.stanford.edu`. letzter Zugriff 05.10.2017.

[26] C. Plaisant and B. Shneiderman. Show me! guidelines for producing recorded demonstrations. In *Visual Languages and Human-Centric Computing, 2005 IEEE Symposium on*, pages 171–178. IEEE, 2005.

[27] C. Rupp. Requirements engineering - der einsatz einer natürlichsprachlichen methode bei der ermittlung und qualitätsprüfung von anforderungen. *OBJEKTspektrum*, 2000.

[28] C. Rupp. *Requirements-Engineering und-Management: professionelle, iterative Anforderungsanalyse für die Praxis*. Hanser Verlag, 2007.

[29] C. Rupp et al. *Requirements-Engineering und-Management: Aus der Praxis von klassisch bis agil*. Carl Hanser Verlag GmbH Co KG, 2014.

[30] C. Schneider. Towards an eclipse ontology framework: Integrating owl and the eclipse modeling framework. In *Proceedings of the 3rd Workshop on Transforming and Weaving Ontologies in Model Driven Engineering, Málaga, Spain*, 2010.

[31] K. Schneider. Anforderungen klären mit videoclips. In *Software Engineering*, pages 93–104, 2010.

[32] K. Schneider. Requirements engineering. Vorlesungsskript, 2015.

[33] M. Uschold and M. Gruninger. Ontologies: Principles, methods and applications. *The knowledge engineering review*, 11(2):93–136, 1996.

[34] R. Van Solingen, V. Basili, G. Caldiera, and H. D. Rombach. Goal question metric (gqm) approach. *Encyclopedia of software engineering*, 2002.

[35] J. G. Walch. *Philosophisches Lexicon*. Gleditsch, 1733.

[36] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

Abbildungsverzeichnis

Tabellenverzeichnis