Master Thesis

Entwicklung und UNICASE-Umsetzung eines Konzepts für die Integration der Modellierung von persönlichen Werten als Ziele in den Softwareentwicklungsprozess

Sommersemester 2011

Eingereicht von:

Markus Fischer

fisch3r@gmail.com

Betreuer:

Prof. Dr. Barbara Paech
Rumyana Proynova

Danksagung

Hiermit danke ich Rumyana Proynova für die sehr gute Betreuung. Die stetigen Treffen und die damit verbundenen fachlichen Diskussionen waren eine große Hilfe bei der Erstellung dieser Arbeit.

Weiterhin gilt mein Dank Frau Prof. Dr. Barbara Paech für die Betreuung, aber auch für das in zahlreichen interessanten Vorlesungen vermittelte Wissen zu Themen aus dem Bereich des Software Engineerings.

Abschließend möchte ich meiner Lebensgefährtin Johanna Emich herzlichst für ihre Unterstützung danken. Ich konnte mich immer auf sie verlassen und sie stand mir stets mit gutem Rat zur Seite.

Zusammenfassung

Im Umfeld des Requirements Engineering spielen Projekt- oder Geschäftsziele eine wesentliche Rolle für die Definition der Anforderungen. Dies wird vor allem im zielorientierten Requirements Engineering umgesetzt. Doch obwohl auch die Entscheidungsträger sehr wichtig für die Definition eines Zielmodells sind, schenkt man bisher persönlichen Zielen dieser Personen keine Beachtung. Mehr Information zu den Zielen von Entscheidungsträgern kann man über die Untersuchung deren persönlicher Werte erlangen. Solche Werte, wie etwa Sicherheit, können beispielsweise zur Bewertung von Alternativen im einen definierten Zielmodell genutzt werden oder sich bei der Prüfung auf Konflikte als hilfreich erweisen.

Im Rahmen dieser Arbeit wird eine Methode zur Nutzung von persönlichen Werten in Zielmodellen für die bereits genannten Anwendungsfälle erarbeitet und implementiert. Als Basis für die Methode werden Ansätze aus der Literatur herangezogen und bezüglich ihrer Eignung bewertet. Auf Grundlage dieser Ergebnisse erfolgt die Definition von Elementen und Vorgehensweise zur Analyse von Alternativen und Konflikte unter Berücksichtigung persönlicher Werte. Für die persönlichen Werte wird das Modell von Schwartz et al. [1] verwendet.

Die Implementierung der erarbeiteten Methode erfolgt in Form eines UNICASE Plugins. UNICASE ist ein Eclipse-basiertes CASE Tool, welches im Zuge der Implementierung um einen Editor zur Zielmodellierung erweitert wird. Zur Integration der zielorientierten Methoden mit den prozess- und aufgabenorientierten Elementen in UNICASE wird ein Metamodell erarbeitet, welches die Verknüpfung von Zielelementen mit anderen Elementen ermöglicht. Die Umsetzung umfasst neben der vorgestellten Methode die gezielte Visualisierung von Teilausschnitten des Zielmodells.

Somit entsteht einerseits eine Methode zur Nutzung von persönlichen Werten im Bereich der Zielmodellierung und im Zuge der Umsetzung in UNICASE ein Editor zur Zielmodellierung, dessen Metamodell eine Integration der zielorientierten Methode mit den UNICASE-Modellelementen ermöglicht. Im Rahmen einer Fallstudie wird schließlich die zuvor definierte Methode zur Nutzung von persönlichen Werten in Zielmodellen eingesetzt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

BPMN Business Process Model and Notation

BPMO Business Process Modeling Ontology

EMF Eclipse Modeling Framework

GMF Graphical Modeling Framework

GMP Graphical Modeling Project

GRL Goal-oriented Requirement Language

KAOS Knowledge Acquisition in autOmated Specification of software systems

OMG Object Management Group

SD Strategic Dependency

SR Strategic Rationale

UCM Use Case Map

UML Unified Modeling Language

URN User Requirements Notation

1 Einleitung

1.1 Kontext und Ziele der Arbeit

Die Definition von Anforderungen ist für das Gelingen eines Softwareprojekts ein sehr wichtiger Aspekt [3]. Die Ergebnisse bilden die Grundlage für alle weitere Tätigkeiten. Dementsprechend können sich Fehler bei der Anforderungsspezifikation auf alle weiteren Teile eines Projektes negativ auswirken oder gar ein Scheitern provozieren. Das sogenannte Requirements Engineering bietet verschiedene Ansätze und Möglichkeiten Anforderungen zu ermitteln und zu dokumentieren.

Die zielorientierten Methoden des Requirements Engineering sind insbesondere für die sehr frühe Anforderungsermittlung geeignet. Sie erfassen grundlegende Ziele für die Erstellung des Systems durch die Betrachtung des Umfelds, bestehender Systeme und durch das Führen von Interviews mit beteiligten Personen - sie werden allgemein Stakeholder genannt. Die Erfüllung der gefundenen Ziele im weiteren Verlauf des Projekts ist maßgeblich für dessen Gelingen verantwortlich. Um diese Ziele zu erreichen stehen oft verschiedene Alternativen zur Verfügung. Die Entscheidungen zwischen diesen Alternativen sind sehr bedeutend, da sie die Weichen für den weiteren Verlauf des Projekts stellen. Gleiches gilt für die Erkennung von Konflikten um frühzeitig Lösungen finden zu können. Daher werden zielorientierte Ansätze, beispielsweise zur Definition und Überprüfung von Anforderungen für Softwaresysteme, in kritischen Bereichen eingesetzt [4].

Obwohl die Stakeholder eine wesentliche Rolle bei der Definition der Anforderungen spielen, werden deren persönliche Ziele von bestehenden Methoden nicht erfasst. Jedoch ist genau die Erfüllung der persönlichen Ziele ein kritischer Aspekt. Sie bilden die Motivatoren der Stakeholder und können weitere Informationen bezüglich der Eignung von verschiedenen Umsetzungen von Zielen liefern [5]. Ein Ziel dieser Arbeit ist dabei, herauszufinden wie persönliche Ziele in die Zielmodellierung eingebracht werden können.

Zur erfolgreichen Umsetzung eines definierten Zielmodells müssen nachfolgend definierte Anforderungen und Prozesse stets im Bezug zu den ermittelten Zielen stehen, damit eine Befriedigung der Wünsche der Stakeholder auch tatsächlich stattfindet. Die Abstimmung der Ziele und Prozesse wird auch „IT Alignment“ genannt [6]. Die praktische Umsetzung gestaltet sich jedoch oft schwierig, da kaum Werkzeuge zur Unterstützung dieser Aufgabe vorhanden sind. Die Verknüpfung von Zielen und nachfolgenden Aufgaben und Prozessen stellt ein weiteres Ziel dieser Arbeit dar. Um dieses zu erreichen wird auf ein bereits bestehendes Werkzeug aufgebaut: UNICASE.

Bei UNICASE handelt es sich um ein CASE-Werkzeug, mit dem die in den verschiedenen Phasen des Software-Entwicklungsprozesses entstandenen Artefakte einheitlich verwaltet und bearbeitet werden können [7]. Typische Artefakte, die in UNICASE abgelegt werden, sind Anforderungsdefinitionen, Use Cases, Unified Modeling Language (UML) Modelle, aber auch Bugs und Features. UNICASE verwaltet nicht nur die einzelnen Artefakte, sondern auch die Beziehungen zwischen diesen, wodurch die Nachvollziehbarkeit von Anforderungen ermöglicht wird. UNICASE-Projekte können auch versioniert werden. Zudem beherrscht UNICASE auch weitreichende Features aus dem Projektmanagement beherrscht, wie beispielsweise das Planen von Iterationen.

Die vorliegende Arbeit umfasst somit die Erarbeitung einer Methode zur Integration von persönlichen Zielen in die Disziplin der Zielmodellierung. Die Methode beinhaltet die Notation von Elementen, sowie Verfahren zur Analyse von Alternativen und Konflikten. Dabei werden die persönlichen Ziele von Stakeholdern in diesen Prozess ein eingebracht. Das Konzept wird schließlich in Form einer Erweiterung für UNICASE umgesetzt und dessen Tauglichkeit anhand einer Fallstudie überprüft.

1.2 Aufbau der Arbeit

Zunächst sollen in Kapitel 2 wichtige Grundlagen für den weiteren Verlauf erläutert werden. Darunter fällt eine genaue Betrachtung der Definition verschiedener Zielarten sowie persönlicher Werte, die auch als persönliche Ziele der Stakeholder gesehen werden können. Weiterhin erfolgt die Vorstellung und kritische Betrachtung verschiedener Ansätze zur Zielmodellierung. Da ein Ziel der vorliegenden Arbeit darin besteht, Ziele mit anderen Elementen des Requirements Engineering zu verknüpfen, werden verschiedene Modelltypen vorgestellt und hinsichtlich ihrer Beziehungen zu Zielen betrachtet.

In Kapitel 3 erfolgt die genaue Definition der Anforderungen für eine zu entwickelnde Methode. Im Zuge dieser Anforderungsdefinition findet die Evaluation von Ansätzen aus der Literatur hinsichtlich der Analysefähigkeiten der Methode statt. Wie bereits angedeutet ist die Analysierbarkeit von Zielmodellen ein wichtiges Werkzeug um frühzeitig richtige Entscheidungen treffen zu können oder Konflikte zu vermeiden. Nach der Definition der Anforderungen wird ein Blick auf bestehende Werkzeuge zur Zielmodellierung geworfen.

Die Definition einer Methode zur Nutzung persönlicher Werte in Zielmodellen ist Inhalt von Kapitel 4. Basierend auf den Ergebnissen des vorangegangen Kapitels und Hinweisen aus der Literatur werden die Elemente der Methode sowie das Vorgehen zur Erstellung und zur Analyse eines bestehenden Zielmodells vorgestellt.

Kapitel 5 umfasst die Umsetzung der vorgestellten Methode in UNICASE. Dies beinhaltet die Erläuterung der funktionalen Anforderungen sowie die Vorstellung eines Metamodells zur Verknüpfung von Zielen mit den bestehenden Elementen von UNICASE.

Dazu können die Erkenntnisse aus dem Grundlagenkapitel verwendet werden. Schließlich erfolgt die Vorstellung der erstellten Komponenten und deren Struktur.

Der Einsatz der Methode wird in Kapitel 6 im Rahmen einer Fallstudie vorgestellt. Dies umfasst die Erstellung des Zielmodells und dessen Analyse. Ebenso werden im Rahmen der Nutzung erlangte Erfahrungen erläutert.

Abschließend wird ein Fazit zu den Ergebnissen der Arbeit gezogen und ein Ausblick auf mögliche Verbesserungen und Weiterentwicklungen für die bisherige prototypische Umsetzung in UNICASE.

2 Grundlagen

In diesem Kapitel werden einige Grundlagen und weitere Informationen vorgestellt die für den weiteren Verlauf der Arbeit bedeutsam sind. Abschnitt 2.1 umfasst die Betrachtung des Zielbegriffs. Die Erläuterung von persönlichen Werten ist in Abschnitt 2.2 dargestellt. Die Aspekte zur Zielmodellierung werden in Abschnitt 2.3 betrachtet. Schließlich folgt die Vorstellung von Zielen in anderen Modellen des Requirements Engineering in Kapitel 2.4.

2.1 Ziele

Allgemein versteht man unter Zielen etwas nach dem Personen streben [8]. Diese können auf verschiedenen Abstraktionsebenen definiert werden, beispielsweise in Form vom strategischen Zielen oder bestimmten technischen Zielen. Im Rahmen des zielorientierten Requirements Engineering nutzt man Ziele um deren Sicherstellung durch das zu modellierende System sicherzustellen [9]. Diese Ziele sind dabei Personen, den sogenannten „Stakeholdern“, zugeordnet. Hierin liegt auf der wesentliche Unterschied von Zielen im Gegensatz zu Anforderungen im Requirements Engineering, diese sind stets einem Produkt oder System zugeordnet [8]. Potentielle Stakeholder sind im Rahmen der Zielmodellierung am Projekt beteiligte Personen, somit auch Entscheidungsträger, und zukünftige Benutzer. Entwickler sind typischerweise keine Stakeholder, da sie zwar Projektbeteiligte sind aber am Betrieb des späteren System nicht beteiligt sind [8].

Die Feststellung ob ein Ziel erreicht ist, muss nicht unbedingt gegeben sein [8], daher unterscheidet man im zielorientieren Requirements Engineering in vielen Methoden [10, 2, 11] zwischen „Goals“ (auch „Hard Goals“ oder „Functional Goals“) und „Soft Goals“ (auch „Quality Goals“). Unter Hard Goals versteht man Ziele die quantifizierbar sind und zu denen klar definierte Kriterien zur Erfüllung existieren - Soft Goals hingegen fehlen solche Kriterien [9]. Ein Hard Goal könnte beispielsweise „Authentifizierung“ lauten, ein Soft Goal hingegen „benutzerfreundliche Authentifizierung“. Soft Goals können somit zu einem gewissen Grad erfüllt sein.

Im zielorientierten Requirements Engineering werden Ziele vielfältig verwendet [9]. Beispielsweise können sie als Kriterium für die Vollständigkeit einer Spezifikation dienen. Ebenso werden irrelevante Anforderungen vermieden, da sich alle Anforderungen an den definierten Zielen orientieren. Hinzu kommt die Möglichkeit, die Anforderungsspezifikation mit höherwertigen Zielen, wie etwa strategischen Zielen, zu verknüpfen. Auch bei der Evaluation von Alternativen können Ziele unterstützend wirken. Da Ziele

in ihrer Form wesentlich ungenauer sein können als Anforderungen - da sie Stakeholdern zugeordnet sind - können Konflikte auftreten. Diese gilt es, für eine im Anschluss folgende konfliktfreie Anforderungsspezifikation, zu erkennen und aufzulösen [8].

Eine der schwierigsten Aufgaben stellt die konkrete Definition der Ziele dar. Nach [9] können die Defizite des derzeitigen Systems zur Definition herangezogen werden, da deren Negation Anhaltspunkte für mögliche Ziele liefert. Ebenso bietet sich die Untersuchung von Interviews und anderen Dokumenten nach Schlüsselwörtern an [8, 9]. Eine so gefundene Menge von Zielen kann nach Lamsweerde [9] mit gezielten Fragen nach dem „Wie?“ und „Warum?“ gezielt verfeinert werden.

2.2 Persönliche Werte

Unter einem persönlichen Wert versteht man in der Sozialpsychologie nach Schwartz und Bilsky [12]:

> „values are (a) concepts or beliefs, (b) about desirable end states or behaviors, (c) that transcend specific situations, (d) guide selection or evaluation of behavior and events, and (e) are ordered by relative importance“

In [13] wird der Begriff „treibende Kraft“ für einen persönliche Wert verwendet. Demnach spielt ein solcher persönlicher Wert auch für das Treffen von Entscheidungen eine Rolle. Schwartz und Bilsky [12] nennen persönliche Werte oft auch persönliche Ziele. Jedoch verstehen die Autoren darunter nicht die in Kapitel 2.1 erläuterten Ziele. Die Definition macht deutlich, dass persönliche Werte auf einer wesentlich höheren Ebene anzuordnen sind.

Eine einheitliche Vorstellung davon welche persönlichen Werte tatsächlich existieren ist in der Literatur nicht vorhanden. Einen guten Überblicken geben die Autoren in [13] zu Ansätzen, die sich mit der Definition einer Menge von Werten beschäftigen. Diese arbeiten auf verschiedenen Ebenen, die jeweils aufeinander aufbauen: Persönlichkeit, Repräsentation und Motivation. Mit der zuletzt genannten Ebene befasst sich der Ansatz von Schwartz et al. [1]. Dieser ist besonders hervorzuheben, da zur Erfassung von persönlichen Werten umfassende Untersuchungen in 60 Ländern angestellt wurden. Basierend auf den Ergebnissen konnten zehn verschiedene Werte gefunden werden, dargestellt in Tabelle 2.1 und Abbildung 2.1. Neben den persönlichen Werten haben Schwartz et al. [1] auch deren Beziehungen zueinander untersucht. Demnach ist eine Anordnung auf zwei Achsen möglich, wie in Abbildung 2.1 zu sehen. Auf diesen stehen sich die Extreme „Offenheit für Veränderungen“ und „Erhaltung“ sowie „Selbsttranszendierung“ und „Selbststeigerung“ gegenüber.

Schwartz et al. [1] konnten außerdem zeigen, dass sich andere Variablen, wie Alter oder Bildung, auf die persönlichen Werte auswirken. Diese sind demnach nicht starr, sondern können sich über die Zeit hinweg verändern. Werte wie „Sicherheit“, „Konformi-

Macht	Dominanz bezüglich dem Verhalten Anderer und bzgl. Ressourcen
Leistung	Persönlicher Erfolg, erreicht durch das Ausspielen von Kompetenz
Hedonismus	Persönlicher Genuss als Ziel
Stimulation	Streben nach dem Reiz des Neuen, nach Herausforderungen
Selbstbestimmung	Streben nach Unabhängigkeit, selbstbestimmtes Handeln
Universalismus	Das Wohl der Allgemeinheit steht über dem persönlichen Wohl
Benevolenz	Das Wohl der näheren Bekannten steht über dem persönlichen Wohl
Tradition	Innerhalb der Familie und Kultur weitergegebene Tradition werden als wichtig erachtet
Konformität	Befolgen von Anweisungen und das Einhalten von Gesetzen
Sicherheit	Sicherheit bezüglich der Gesellschaft, der Beziehungen und sich selbst

Tabelle 2.1: Persönliche Werte nach Schwartz et al. [1]

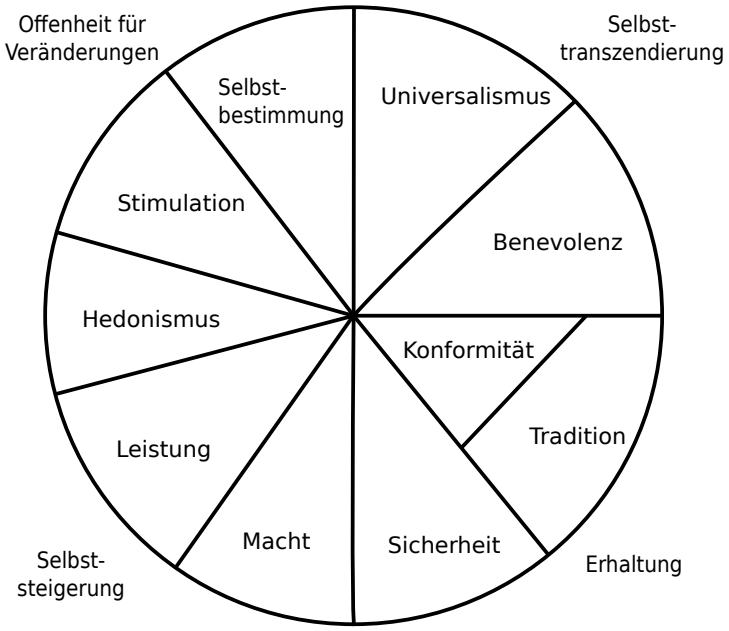


Abbildung 2.1: Beziehungen zwischen persönlichen Werten nach Schwartz et al. [1]

tät“ oder „Tradition“ gewinnen mit zunehmendem Alter an Relevanz. Mit zunehmender Bildung sind vor allem die Werte „Selbstbestimmung“, „Benevolenz“ und „Stimulation“ ausgeprägt.

Vergleicht man persönliche Werte mit den verschiedenen Zielarten im Bereich des zielorientierten Requirements Engineering (siehe Kapitel 2.1), so sind persönliche Werte am ehesten mit Soft Goals vergleichbar. Die Erfüllung eines persönlichen Werts ist, wie bei einem Soft Goal, nicht messbar - es existieren keine klaren Metriken und eine Quantifizierbarkeit ist ebenso nicht möglich. Jedoch gestaltet sich die Bewertung von persönlichen Werten, welche insbesondere bei der Evaluation von Alternativen notwendig ist, deutlich komplexer als bei Soft Goals. Beispielsweise kann ein Soft Goal „Benutzerfreundliche Authentifizierung“ durch verschiedene Kriterien wie „Anzahl notwendiger Interaktionen“ annähernd evaluiert werden, bei persönlichen Werten wie etwa „Macht“ gestaltet es sich wesentlich komplexer sinnvolle Kriterien zu finden. Ein weiteres Problem bei der Nutzung von persönlicher Werte innerhalb einer Methode, die auch Konflikte erkennen soll, ist die nur scheinbare Gegensätzlichkeit der Werte. Nimmt man an, dass in einen Zielmodell zwei Aktoren vorhanden sind, die gegensätzliche persönliche Werte vertreten - beispielsweise „Selbstbestimmung“ und „Sicherheit“ - und beide Aktoren eine Verbindung zu einem gemeinsamen Ziel besitzen, so bedeutet das nicht automatisch, dass ein Konfliktpotential besteht. Ein Beispiel hierfür ist ein Ziel „Keine Weitergabe von Daten an Dritte“, welches von Personen mit dem persönlichen Wert „Selbstbestimmung“ ebenso angestrebt wird wie von Personen mit dem persönlichen Wert „Sicherheit“. Als Unterschied zwischen persönlichen Werten und Soft Goals ist schließlich noch die Beschränktheit der persönlichen Werte zu nennen: Sie können lediglich die zehn möglichen Werte nach Schwartz et al.[1] annehmen.

2.3 Zielmodellierung

Während in Kapitel 2.1 bereits Bezug auf die verschiedenen Arten von Zielen im zielorientierten Requirements Engineering genommen wurde, widmet sich dieser Abschnitt der Modellierung von Zielen. Diese beinhaltet neben der Erfassung auch die Anordnung sowie die Kenntlichmachung von Beziehungen zwischen den Zielen untereinander. So existieren in der Literatur verschiedene Ansätze, die verschiedene Möglichkeiten bieten, um Ziele miteinander in Verbindung zu bringen - wie der nächste Abschnitt zeigt. Der letzte Abschnitt betrachtet die Kritik, die man in der Literatur, für die zuvor betrachteten Ansätze, findet.

2.3.1 Verbreitete Ansätze

Das Erfassen und Anordnen von Zielen in Form eines Modells ist in der Literatur in verschiedenen Ansätzen beschrieben. Im weiteren Verlauf dieser Arbeit werden diese Ansät-

ze immer wieder erwähnt, daher erfolgt nun eine genauere Betrachtung in den weiteren Abschnitten. Tabelle 2.2 gibt einen Überblick zu den fünf betrachteten Ansätzen, hinsichtlich vorhandener Elemente, Stakeholder, Relationen, formaler Definition, Eignung und weiteren Besonderheiten.

2.3.1.1 KAOS

Einer der ältesten und umfassendsten Ansätze zur Zielmodellierung ist Knowledge Acquisition in autOmated Specification of software systems (KAOS). Er wurde 1993 vorgestellt [11] und bietet vielfältige Möglichkeiten zur Modellierung von Zielen - konzeptuell wie auch formal. Das Metamodell von KAOS besteht aus Objects, Actions, Agents und Goals.

Unter einem Object wird in KAOS etwas verstanden, das für die Anforderungen relevant ist und demnach auch referenziert werden kann. Spezielle Ausprägungen eines Objects sind Events, Entities oder Relationships.

Actions stellen Relationen über Objects dar mit Input und Output, deren Anwendung Zustandsübergänge darstellen. Eine Action kann in KAOS mit einer Vor- und Nachbedingung versehen werden, wodurch der Zustandsübergang definiert ist. Ebenso können die Dauer oder die Bedingung für das Stoppen einer Action definiert werden. Als Input und Output kommen Objects in Frage, für die Bedingungen referenziert man einen Event.

Unter Agents versteht man in KAOS im Grunde den Ausführenden einer Action. Dabei muss es sich nicht zwangsläufig um eine Person handeln, es kann beispielsweise auch ein System dahinter stehen. Per Attribut können für jeden Agent die Actions definiert werden, die er ausführen kann (Capability) und die er tatsächlich ausführt (Performs).

Mit Hilfe der Goal Elemente formuliert man in KAOS die Ziele, die mit dem zu entwickelnden System erreicht werden sollen. Man unterscheidet dabei zwischen funktionalen und nicht-funktionalen Zielen. Ein funktionales Ziel kann durch eine Action erreicht werden und ihm ist ein Agent zugeordnet. Dementsprechend werden nicht-funktionale Ziele in KAOS so lange verfeinert, bis sie durch funktionale Ziele ausgedrückt werden können. Diese verfeinerten Ziele können durch UND-/ODER-Relationen angeordnet werden. Für die Ziele existieren verschiedene Muster, beispielsweise „Achieve“, „Avoid“ oder „Maintain“. Weitere Kategorien sind „System“ und „Private“ für die Einordnung der Ziele. Die erste Kategorie bezieht sich auf Ziele die durch das System erbracht werden sollen, wobei noch weitere Unterscheidungen getroffen werden können, wie beispielsweise „Information“ oder „Consistency“. Die zweite Kategorie bezeichnet Ziele von Agents, die aber nicht zwingend durch das System erfüllt werden müssen.

Zur formalen Definition der Elemente in KAOS nutzt man neben den Operatoren der Prädikatenlogik nach [14] auch Operatoren der temporalen Logik [15]. Auf diese Art und Weise lassen sich zeitliche Zusammenhänge ausdrücken, beispielsweise □ (immer in der

Ansatz	Elemente	Stakeholder	Relationen	Formal	Eignung	Besonderheiten
KAOS	Events Entities, Actions, Goals, Soft Goals, verschiedenste Muster für Goals	Agent	UND/ODER	Ja	Definition aller Anforderungen	Sehr ausführliche formale Komponente
NFR-Framework	Soft Goals, Satisficing Goals, Argumentation Goals	-	UND/ODER, qualitative Labels	Nein	Zur Evaluation	-
i*	Goal, Soft Goal, Tasks, Ressource, Belief	Aktor, Agent, Rolle, Position	Decomposition, Means-End, qualitative Labels	Nein	Für grobe Anforderungen	SR- und SD-Modell
TROPOS	Goal, Soft Goal, Task, Ressource, Belief, Capability	Aktor, Agent, Rolle, Position	UND/ODER, qualitative Labels, Delegation und Trust (Secure TROPOS)	Ja, Formal TROPOS	-	
GRL	Goalm Soft Goal, Task, Ressource, Belief	Aktor	UND/ODER/XOR	Nein	Für grobe Anforderungen	Standardisiert (URN), definierte Zuordnung zu UCM-Elementen

Tabelle 2.2: Überblick zu den Ansätzen zur Zielmodellierung

Zukunft), ω (immer in der Zukunft außer) oder $\cdot$ (in der Zukunft). Ein Beispiel aus [16] für die formale Definition des Ziels:

> „A meeting scheduler should know the constraints of the various participants invited to the meeting within some deadline d after invitation“

ist folgende Definition:

$$\begin{aligned} &\forall m\ Meeting, p : Participant, s : Scheduler \\ &\quad Invited(p, m) \wedge Scheduling(s, m) \\ &\quad \Rightarrow \cdot_d Knows(s, p.Constraints) \end{aligned} \tag{2.1}$$

Eine vollständige Spezifikation in KAOS besteht aus verschiedenen Submodellen: dem eigentliche Zielmodell, Object Model, Agent Responsibility Model, Operational Model und Agent Interface Model [14]. Innerhalb der verschiedenen Submodelle werden die bereits vorgestellten Elemente des Metamodells verwendet und die Beziehungen zu den anderen Submodellen definiert.

KAOS eignet sich durch seinen großen Umfang sowohl für die zunächst ungenaue Definition von Zielen als auch für die detaillierte Definition eines Zielmodells. Sogar komplexe Bedingungen und Annahmen können durch den formalen Charakter von KAOS festgelegt werden.

2.3.1.2 NFR-Framework

Das NFR-Framework ist ein spezieller Ansatz zur Modellierung von nicht-funktionalen Zielen [17, 18]. Der Ansatz begründet sich durch die hohe Relevanz von nicht-funktionalen Anforderungen, die die Qualitätseigenschaften eines Systems repräsentieren. Jedoch kommt es oft zu Fällen bei denen sich diese Qualitätseigenschaften widersprechen, beispielsweise Performanz und Modularität. Das NFR-Framework bietet die Möglichkeit, exakt solche Fälle darzustellen und somit explizit kenntlich zu machen.

Das Metamodell des NFR-Frameworks umfasst drei verschiedene Arten von Zielen: NFR-Goals bzw. Soft Goals, Satisficing Goals und Argumentation Goals [17]. NFR-Goals entsprechen Qualitätseigenschaften bzw. nicht-funktionalen Anforderungen eines Systems. Satisficing Goals stellen eine Möglichkeit dar ein NFR-Goal zu erfüllen. Argumentation Goals dienen zur Erfassung von Argumenten bezüglich Abhängigkeiten zwischen den verschiedenen Elementen. Ziele können im NFR-Framework in UND-/ODER-Relationen miteinander verbunden werden. Wirkungen zwischen den NFR-Goals lassen sich mit Hilfe von Contribution-Links darstellen. Diese können positiv (+ bzw. helps) oder negativ (- bzw. hurts) sein, sie werden auch qualitative Labels genannt.

Die Nutzung von Contribution Links ermöglicht die Evaluation von Alternativen sowie die Analyse von Konflikten [18]. So können bereits in einer frühen Phase des Requirements Engineering wichtige Designentscheidungen bezüglich den Soft Goals evaluiert

werden. Da Konflikte erkennbar sind und möglicherweise früh genug erkannt werden können.

2.3.1.3 i*

i* ist ein Ansatz zur Zielmodellierung, der sich auf die Modellierung von Organisationen und deren Informationssysteme spezialisiert [2]. Der Ansatz besteht aus zwei Modellen: Strategic Dependency (SD) und Strategic Rationale (SR). Das erstgenannte Modell wird verwendet um die Beziehungen verschiedener Aktoren in einer Organisation zu modellieren. Das SR-Modell dient zur Beschreibung der Interessen von Stakeholdern und wie diese durch Konfigurationen der Informationssysteme und der Umgebung erreicht werden können.

In i* versteht man unter einem Aktor generell jede Art von organisatorische Einheit, die im betrachteten System eine Rolle spielt [19]. Spezialisierte Formen von Aktoren sind Rollen, Agenten und Positionen. Eine Rolle stellt einen abstrakten Aktor dar, dem Aufgaben und Verantwortlichkeiten zugeordnet werden. Bei Agenten handelt es sich hingegen um einen konkreten Aktor, wobei es sich nicht um eine Person handeln muss. Einem Agenten können mehrere Rollen zugeordnet werden - eine solche Zuordnung nennt man Position.

Ein zentrales Konzept von i* ist der sogenannte „Intentional Actor“. Darunter versteht man das Bewusstsein von Aktoren gegenüber ihren Zielen, Vorstellungen, Fähigkeiten und Bekenntnisse. Um Ziele zu erreichen hängen Aktoren voneinander ab. Dadurch können sie gemeinsam Ziele erreichen, die sie alleine nicht erreichen hätten können. Gleichermaßen geht damit auch eine potentielle Gefahr für die Ziele eines jeden Aktors einher, da er abhängig von den Leistungen anderer Aktoren ist. Die Erfassung dieser Abhängigkeiten erfolgt im bereits erwähnten SD-Modell.

Ein Ziel ist in i* etwas, das ein Aktor erreichen möchte [19]. Handelt es sich um funktionale Belange, wird es Goal, bezüglich Qualitätsanforderungen wird es Soft Goal genannt. Um diese Ziele zu erreichen führen Agenten sogenannte Tasks aus. Darunter versteht man die konkreten Operationen um ein Ziel zu erreichen. Weitere Konzepte in i* sind Ressourcen und Beliefs. Eine Ressource ist eine physische Entität oder eine Information. Ein Belief dient zum Ausdruck von bestimmten Bedingungen oder Annahmen.

Im SD-Modell von i* können zur Darstellung der Abhängigkeiten von Aktoren verschiedene Beziehungstypen verwendet werden. Diese beziehen sich auf Goal, Softgoal, Task oder Resource Elemente [2]. Im SR-Modell werden schließlich die Ziele der im SD Modell verwendeten Aktoren detailliert modelliert. Abbildung 2.2 zeigt ein SR-Modell aus [2]. Die Elemente der Aktoren sind jeweils eingekreist. Die Elemente, die zwischen den Kreisen der Aktoren stehen, stellen die relevanten Elemente der Beziehungen im SD-Modell dar. Zur Darstellung der Abhängigkeiten zwischen einem Ziel und einer Task dient die Means-End-Beziehung [19]. Diese beschreibt im Grunde eine ODER-Relation,

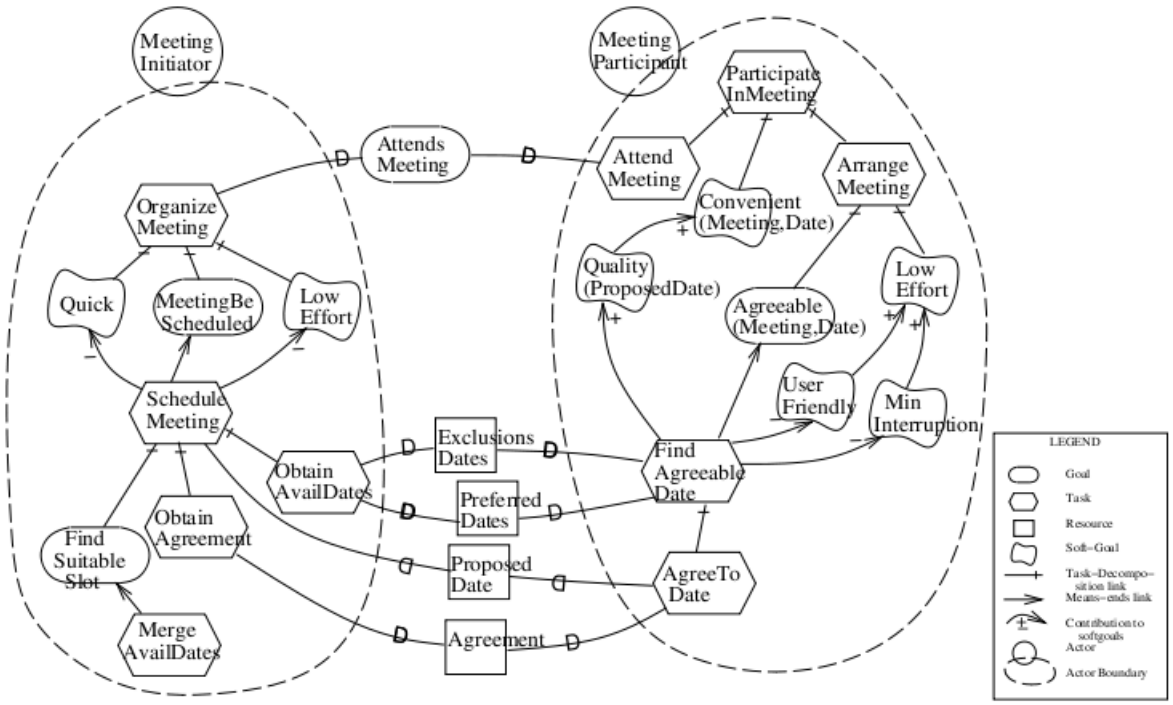


Abbildung 2.2: Beispiel für ein SR Modell [2]

die speziell für die Ziele und Tasks dient. Abhängigkeiten zwischen Tasks und Goals, Softgoals, Resources und Tasks werden per Decomposition dargestellt - vergleichbar mit einer UND-Relation. Zwischen den Elementen im SR-Modell können Contrubution Links eingesetzt werden. Damit können analog zum NFR-Framework Auswirkungen von Elementen aufeinander modelliert werden. i* definiert dazu verschiedene qualitative Labels wie ++ oder –.

2.3.1.4 Tropos

Eine spezielle Form von i* stellt der Ansatz Tropos dar [20]. Im Gegensatz zu i* ist Tropos nicht nur für die Erfassung der ersten groben Anforderungen geeignet, sondern kann in verschiedenen Phasen des Softwareentwicklungsprozesses eingesetzt werden. Dazu teilt Tropos den Softwareentwicklungsprozess in vier Phasen auf: frühe Anforderungen, späte Anforderungen, Design der Architektur und detailliertes Design.

In den ersten beiden Phasen ist Tropos i* sehr ähnlich. Bezüglich der Entitäten wird das i*-Metamodell um Capability erweitert, worunter die Fähigkeiten eines Aktors verstanden werden, einen Plan auszuführen. Pläne sind in Tropos das Gegenstück zu den Tasks in i*. Nach der Definition der Beziehungen der Aktoren, sowie der Abhängigkeiten zwischen

den Elementen des Zielmodells legt Tropos verschiedene Regeln fest, die es schließlich zulassen, anhand der Zielmodelle eine UML Spezifikation zu erstellen.

Eine Formale Erweiterung von Tropos existiert in Form von Formal Tropos [21]. Durch die Nutzung von Datalog können in Tropos Zielmodelle komplett formal erfasst werden, die weitgehend automatisierte Analysen ermöglichen.

Für Tropos gibt es des Weiteren eine Form, die sich speziell an Anforderungen in sicherheitskritischen Applikationen richtet: Secure Tropos [21]. Dazu werden drei neue Beziehungen eingeführt: Ownership, Trust und Delegation. Ownership zeigt den Besitzstatus eines Aktors bezüglich eines Ziels, einer Ressource oder eines Plans an. Die Trust-Beziehung drückt aus, dass ein Aktor einem anderen Aktor bezüglich eines bestimmten Ziels vertraut. Die Delegation-Beziehung ermöglicht die Delegation eines Ziels, die Ausführung eines Plan oder den Zugriff auf eine Ressource von einem Aktor zu einem anderen. Ein weiteres Konzept zur Delegation stellt in Secure Tropos ein Monitor dar. Darunter versteht man einen Aktor, der den Delegierten beaufsichtigt wodurch die Überprüfung ermöglicht wird. Die Delegation- und Trust-Beziehung sowie ein Monitor kann verfeinert werden in at-most- und at-least-Delegationen, -Trust bzw. -Monitor. So bezeichnet eine at-most-Delegation eine Art Erlaubnis etwas zu tun, eine at-least-Delegation hingegen erwartet eine bestimmte Aktion.

2.3.1.5 GRL

Goal-oriented Requirement Language (GRL) ist ein weiterer Ansatz der viele Konzepte von i* übernimmt. Jedoch ist GRL wesentlich kompakter und als einziger Ansatz zur Zielmodellierung in einer bestimmten Form standardisiert [10]. So wurde GRL als Teil der User Requirements Notation (URN) Z.151 der International Telecommunictations Union (ITU-T) akzeptiert [22]. Dieser Standard umfasst neben GRL auch Use Case Maps (UCM), eine Notation für Szenarien.

Die Hauptunterschiede zwischen GRL und i* bestehen in der Definition der Aktoren und den Abhängigkeiten zwischen den Elementen [10]. So gibt es in GRL zunächst nur ein Modell. Es existiert kein gesondertes SD-Modell wie bei i*, sondern im Grunde nur ein SR-Modell. In GRL gibt es zudem nur das Konzept des Aktors; eine Unterscheidung zwischen Rollen oder Agenten wird nicht vorgenommen. Bei der Definition der Abhängigkeiten zwischen Elementen im SR-Modell existieren bei i* einige Einschränkungen bei Means-End- und Decomposition-Beziehungen. GRL bietet hier mehr Freiheit und verzichtet völlig auf Restriktionen. Abhängigkeiten zwischen den Elementen können mit UND-, ODER- und XOR-Relationen beschrieben werden.

Ein GRL-Zielmodell ist im Wesentlichen dem i*-Zielmodell sehr ähnlich, allerdings können beliebige GRL-Elemente zu beliebigen UCM-Elementen zugeordnet werden. Auf diese Art und Weise ist die Verbindung der modellierten Ziele und Szenarien möglich.

2.3.2 Kritik an Methode und Ansätzen

Bei der Betrachtung der Zielmodellierung sind neben den bestehenden Ansätzen und deren Verwendung auch Kritikpunkte aus der Literatur von Interesse. Insbesondere, da eines der Ziele dieser Arbeit die Nutzung und Erweiterung eines Ansatzes ist.

Die gefundene Kritik richtet sich zum einen an bestimmte Ansätze, ist aber auch allgemein für Zielmodellierung gültig. In [23] wurde eine Evaluation der Praxistauglichkeit von KAOS anhand von zwei Fallstudien durchgeführt. Es zeigte sich, dass vor allem die formale Definition des Zielmodells sehr zeitaufwendig ist. Insbesondere die Modellierung von Hemmnissen oder auch verschiedener Alternativen ist sehr erschöpfend. Zudem fiel auf, dass in KAOS Faktoren, wie beispielsweise der Aufwand der möglichen Lösungen, nicht definierbar sind. Gerade bei der Auswahl einer Alternative ist dieser Faktor in der Praxis von wesentlicher Bedeutung. Dadurch sind gefundene Lösungen möglicherweise gar nicht praktisch umsetzbar.

Die formale Komponente der Ansätze bemängelt auch [24]. Diese wird vor allem in der Forschung genutzt und hat kaum praktische Relevanz. In der Praxis müssen vor allem die Werkzeuge den Benutzer unterstützen, beispielsweise mit Mustern. Die Autoren in [25] bemängeln allgemein, dass die Ansätze keinerlei Unterstützung für den praktischen Einsatz bieten. Oft handelt es sich lediglich um einfache Heuristiken - systematische Anleitungen fehlen. Dies wirkt sich insbesondere negativ aus, wenn die Komplexität bei der Modellierung stark ansteigt. Zudem spielen Entscheidungsträger oft eine wichtige Rolle in der Definition der Modelle, es fehlen aber beispielsweise Methoden zu deren Identifizierung oder effektiven Einbindung. Ein weiterer Kritikpunkt stellt die Formalität von Methoden dar. Diese bringt zwar in der Regel eine gute Werkzeugunterstützung mit sich, lässt aber oft auch keine Konflikte zwischen Zielen oder andere Inkonsistenzen zu. Dadurch verliert man Freiheiten in der Definition, semi-formale Methoden sehen die Autoren daher klar im Vorteil. Schließlich bemängeln sie auch die fehlende praktische Erfahrung mit den verschiedenen Ansätzen. Nach [26] könnte eine Ursache für die bisher geringe Nutzung in der Praxis die Menge der verschiedenen Ansätzen sein. Denn obwohl sich diese oft nur gering unterscheiden in ihren Absichten bietet fast jeder Ansatz eine komplett eigene Notation.

In [27] und [28] betrachten die Autoren vor allem die visuellen Aspekte der Zielmodelle als einen sehr wichtigen Aspekt für die praktische Nutzung. Die Betrachtung von i* in [27] ergab unter anderem, dass die visuelle Notation nie begründet wurde. Zudem werden häufig grundsätzlich verschiedene Elemente mit der gleichen Form dargestellt. Die abstrakten Formen in der i* Notation sorgen außerdem dafür, dass der Ansatz aufwendig gelernt werden muss. Zudem bietet i* keine Hierarchiestufen um komplexe Modelle besser zu strukturieren. In ähnlicher Weise wird KAOS in [28] kritisiert. Es fehlt an einer guten Unterscheidbarkeit der Elemente und es fehlen Möglichkeiten zur Strukturierung des Zielmodells.

2.4 Ziele in anderen Modellen des Requirements Engineering

Neben der reinen Modellierung von Zielen stellt sich auch die Frage, inwiefern Ziele in anderen Modellen des Requirements Engineerings eine Rolle spielen. Insbesondere bei Modellen, die auch im aufgabenorientierten Requirements Engineering verwendet werden. Eine solche Evaluation gibt Aufschluss über die Möglichkeiten zur Integration beider Disziplinen. Dazu werden vier verschiedene Modellarten betrachtet.

2.4.1 BPMN Diagramme

Bei der Business Process Model and Notation (BPMN) handelt es sich um einen Standard der Object Management Group (OMG) zur Beschreibung von Geschäftsprozessen, der insbesondere zur Vermittlung zwischen dem Design der Geschäftsprozesse und der anschließenden Implementierung dient [29]. Die BPMN unterscheidet fünf verschiedene Kategorien von Elementen: Flow-Objects, Data, Connecting-Objects, Swimlanes und Artifacts. Die Kategorien Flow-Objects und Swimlanes beinhalten die wichtigsten Diagrammelemente und werden daher im Folgenden genauer betrachtet.

Die Flow-Objects umfassen Events, Activities und Gateways. Insbesondere mit den Activities, worunter Sub-Processes und Tasks fallen, können Abläufe beschrieben werden. Die beiden Elemente unterscheiden sich lediglich in ihrer Granularität. Eine Task ist eine atomare Aktivität im Prozessablauf, wohingegen man unter einem Sub-Process eine komplexe Aktivität versteht. Zur Einteilung der Diagramme in Zuständigkeitsbereiche dienen die Elemente der Kategorie Swimlane: Pool und Lane. Ein Pool bezeichnet eine Rolle, die beispielsweise auch einer organisatorische Einheit zugeordnet sein kann. Eine Lane hingegen unterteilt einen Pool, stellt also beispielsweise eine bestimmte Rolle innerhalb einer organisatorischen Einheit dar [29]. Die OMG definiert in [29] einen Geschäftsprozess als:

> „A defined set of business activities that represent the steps required to achieve a business objective. It includes the flow and use of information and resources.“

Der Gedanke liegt also nahe, dass ein Geschäftsprozess in BPMN in Verbindung mit einem Ziel zu bringen ist. Demnach sind die Elemente der Kategorie Activities den Task-Elementen von Zielmodellen zuzuordnen. Swimlane-Elemente repräsentieren, je nach deren Komplexität, entsprechend Aktoren im Zielmodell. In ähnlicher Form gestaltet sich der Ansatz in [30] zur Abbildung von i*-Zielmodellen zu BPMN-Diagrammen. Organisationen und externe Aktoren aus i* werden in BPMN Pools zugeordnet, interne Aktoren einer Organisation sind im entsprechenden Pool angeordnet. Weiterhin bilden die Autoren Tasks aus i* auf Sub-Process-Elemente in BPMN ab, alternativ kommen auch atomare

Aktivitäten in BPMN in Frage, je nach Granularität der Task. Ansonsten gibt es keine festen Regeln, weitere Abbildungen müssen manuell erschlossen werden.

In [31] wird ein Ansatz zur Abbildung von Formal Tropos-Zielmodellen zur Business Process Modeling Ontology (BPMO) beschrieben, bei der es sich um die Obermenge von BPMN handelt. Eine konkrete Abbildung besteht für die Elemente Goal und Task des Zielmodells. Ein verfeinertes Goal-Element wird auf einen Prozess abgebildet. Entsprechend sind die anderen Goals Sub-Prozesse. Weitere Elemente des Zielmodells wie Actor, Softgoal, Resource oder Entity müssen manuell, entsprechend des Kontextes und der Bedeutung der BPMO-Konstrukte abgebildet werden. Für die definierten Abhängigkeit im Zielmodell wurden keinerlei Abbildungen definiert.

Die Nutzung von Soft Goals spielt in Zielmodellen auch eine große Rolle, allerdings bietet BPMN keine direkten Möglichkeiten zur Darstellung im Diagramm. Die Autoren in [32] betrachten genau diesen Umstand und schlagen eine Erweiterung der bestehenden BPMN-Notation um die Elemente „Operating Condition“ und „Control Case“ vor. Eine Operating Condition bezieht sich dabei auf Aktivitäten eines Prozesses und beschreibt operative nicht-funktionale Anforderungen, beispielsweise bezüglich der Performanz. Ein Control Case bezieht sich auf eine Operating Condition und stellt eine Art Prüfungsprozess dar. Der Control Case beinhaltet auch Information zum Risikomanagement.

2.4.2 UML Use Cases

Use Case-Diagramme sind Bestandteil des UML-Standards der OMG. Sie dienen zur Beschreibung der Anwendungsfälle, die durch das zu entwickelnde System abgedeckt sein müssen und der daran beteiligten Aktoren [33]. Die Kernelemente von Use Case-Diagrammen sind Aktoren, die Anwendungsfälle selbst und der Systemkontext. Ein Anwendungsfall ist hierbei nach [33]:

> „the specification of a set of actions performed by a system, which yields an observable result that is, typically, of value for one or more actors or other stakeholders of the system.“

Die Aktoren definieren Rollen, die an diesen Anwendungsfällen innerhalb eines bestimmten Systemkontexts beteiligt sind. Weiterhin können Aktoren und Anwendungsfälle über Beziehungen miteinander verbunden werden. Beispielsweise kann eine Generalisierung zwischen Aktoren stattfinden oder die Abhängigkeit zwischen Anwendungsfällen mit „include“ und „extend“ gekennzeichnet werden.

Gemäß der Definition eines Anwendungsfalls kann eine Verbindung zu einem Ziel bestehen. Entsprechend ist es möglich, die Aktoren im Use Case-Diagramm mit Aktoren des Zielmodells gleichzusetzen. In [34] wird ein Ansatz vorgestellt, der sich mit der Extraktion von Use Cases aus i*-Zielmodellen beschäftigt. Ein potenzieller Aktor im Use Case-Diagramm ist grundsätzlich ein externer Aktor in i*. Generalisierung kann hier ebenso

verwendet werden. Für die Identifizierung der Aktoren im Diagramm sind vor allem die Beziehungen der jeweiligen Aktoren im Zielmodell wichtig. Eine Abhängigkeit von einem Ziel führt zu einem Anwendungsfall mit dem Namen des Ziels. Tasks, die verfeinert werden können sind auch Anwendungsfälle. Ebenso sind Ressourcen, die mit einem abstrakten Ziel verbunden sind auch durch einen Anwendungsfall ausgedrückt werden. Soft Goals aus i* sind nicht durch Anwendungsfälle spezifiziert, sondern als nicht-funktionale Anforderung für den entsprechenden Anwendungsfall.

2.4.3 Cockburn Use Cases

Während sich die UML-Spezifikation auf die Notation von Use Cases konzentriert, sowie deren Beziehungen zueinander betrachtet Cockburn [35] vor allem die Methode zur korrekten Definition von Anwendungsfällen. Cockburns Sicht auf einen Anwendungsfall ist wesentlich stärker auf Ziele ausgerichtet. Er definiert einen Anwendungsfall in [35] folgendermaßen:

> „A use case is the statement of the goal the primary actor has toward the system's declared responsibilities, and the collection of possible scenarios between the system under discussion and external actors, showing how the primary actor's goal might be delivered or might fail."

Zudem sieht er für die Einordnung von Anwendungsfällen zwei Bereiche vor, bezüglichen Funktionen und dem Design. Der funktionale Bereich ist mit UML-Anwendungsfällen vergleichbar, die Funktionen des zu definierenden System stehen im Vordergrund. Der Bereich bezüglich des Designs kann nicht nur auf Funktionen des Systems sondern auch auf das benötigte Umfeld angewendet werden. Dazu unterteilt Cockburn den Bereich des Designs nochmals in „Enterprise", „System" und „Subsystem". Ersteres umfasst beispielsweise auch die Definition des Verhaltens der gesamten Organisation. System hingegen betrachtet nur den zu erstellenden Hardware- und Software-Part. Entsprechend ist Subsystem für die genaue Definition eines Teils des Systems gedacht. Wichtig ist, dass bei allen Einteilungen immer das Erreichen des primären Ziels des Aktors im Vordergrund steht.

Für dieses Ziel sieht Cockburn drei verschiedene Ebenen vor: „User", „Summary" und „Subfunction". Ziele der User-Ebene entsprechen dem allgemeinen Ziel des primären Aktors. Hingegen kann man auf der Summary-Ebene mehrere Ziele der User-Ebene gruppieren, beispielsweise um den Kontext dieser Ziele zu verdeutlichen oder um allgemein eine bessere Übersicht zu bieten. Die Erfüllung der Ziele der Subfunction-Ebene ist notwendig um die entsprechenden Ziele der User-Ebene zu erreichen, es handelt sich also um eine Art Verfeinerung. Abschließend ist noch zu bemerken, dass Cockburn Soft Goals nicht erwähnt.

2.4.4 Virtual Windows

Unter Virtual Windows versteht man ein Konzept von Lauesen [36] speziell zur Erfassung von Anforderungen und deren Umsetzung in Benutzerschnittstellen. Demnach ist ein Virtual Window:

> „a user-oriented presentation of persistent data.“

Es handelt sich um eine abstrakte, benutzerorientierte Sicht auf die zu erledigenden Aufgaben und dazu benötigten Daten. Der Fokus liegt dabei auf dem Benutzer. In einem Virtual Window sind alle Funktionen zur Erfüllung einer bestimmten Aufgabe angeordnet. Bei der Modellierung von Virtual Windows ist das Ziel, möglichst wenige Virtual Windows zu verwenden und Funktionen, sowie Daten möglichst nur in einem Virtual Window einzusetzen. Diese Art der Repräsentation kann im weiteren Designprozess genutzt werden um die Benutzerschnittstelle klar zu strukturieren.

Die Formulierung der Aufgaben in einem Virtual Window ist abstrakt und lässt sich gut vergleichen mit Tasks in einem Zielmodell, wie beispielsweise i* oder GRL. Allerdings ist ein derartiger Vergleich nur mit solchen Tasks möglich, die nicht weiter verfeinert werden im Zielmodell. Ein Virtual Window stellt also nur einen Teil eines Zielmodells dar, beispielsweise umfasst es einen Teil der Tasks zur Erfüllung eines Hard Goals. Soft Goals lassen sich mit einem Virtual Window nicht verknüpfen.

3 Nutzen und Anforderungen einer Methode

In den Grundlagen wurde bereits gezeigt, dass persönliche Werte als spezielle Art von Zielen gesehen werden können. Dementsprechend sind die Methoden des zielorientierten Requirements Engineering das Mittel der Wahl um eine Integration in den Softwareentwicklungsprozess zu ermöglichen. Innerhalb dieses Kapitels wird zunächst der Nutzen einer solchen Integration genau betrachtet. Darauf folgt ein Abschnitt zu den Anforderungen, die sich ergeben. Für zwei sehr kritische Anforderungen wird jeweils eine Evaluation durchgeführt. Diese sollen Aufschluss über nutzbare Forschungsansätze geben. Abschließend erfolgt eine Betrachtung der bereits auf dem Markt vorhandenen Werkzeuge zur Zielmodellierung.

3.1 Nutzen für den Softwareentwicklungszyklus

In Kapitel 2.1 wurde bereits der Nutzen von Zielen im Requirements Engineering deutlich. Insbesondere in der frühen Phase der Anforderungsermittlung bilden Ziele dabei wesentliche Anhaltspunkte. Durch die gezielte Modellierung von Zielen können Anforderungen begründet formuliert werden. Es ist somit klar warum eine Anforderung notwendig ist, wodurch unnötige Anforderungen auch automatisch ausgeschlossen sein sollten. Ein Zielmodell ermöglicht ebenso die frühzeitige Analyse, beispielsweise auf Alternativen und Konflikte, wodurch wichtige Entscheidungen bereits in einer frühen Phase in einer begründeten und dokumentierten Form getroffen werden können. Bei der Analyse bestehender Ansätze zur Zielmodellierung aus der Literatur in Kapitel 2.3.1 zeigte sich, dass Stakeholder zwar in Zielmodellen in Form von Aktoren auftauchen, aber lediglich für die Zuordnung zu Zielen eine Rolle spielen. Persönliche Präferenzen von Stakeholdern werden in keiner Form umgesetzt, obwohl das zu erstellende System ausschließlich für die modellierten Stakeholder erstellt wird. Dementsprechend können persönliche Ziele oder Präferenzen von Stakeholdern bisher nicht in die Analyse der Zielmodelle einfließen.

Der Einsatz von persönlichen Werten der Stakeholder ermöglicht deren stärkere Einbeziehung in den Prozess der Zielmodellierung. So können beispielsweise mögliche Lösungen zur Erfüllung eines Ziels als störend gegenüber persönlichen Werten eines Stakeholders identifiziert werden. Eine solche Lösung würde sich mit großer Wahrscheinlichkeit im späteren Betrieb des Systems nicht bewähren, da sie der persönlichen Überzeugung des Stakeholders widersprechen. Bisherige Ansätze lassen eine derartige Identifizierung nicht zu - möglicherweise würden die Probleme erstmals bei der Einführung des Systems erkannt werden. Der finanzielle Aufwand um ein Problem in dieser Phase noch zu lösen

ist in vielen Fällen sehr groß. Das System ist deshalb bei den Stakeholdern nicht sehr beliebt, wodurch eine verminderte Arbeitsmotivation und -leistung aufkommen kann.

Neben dem Nutzen der persönlichen Werte ergeben sich auch durch die Integration der Methode in UNICASE weitere Vorteile. UNICASE ermöglicht die Verwaltung von aufgaben- und prozessorientierten Elementen des Requirements Engineering. Die Verwaltung umfasst automatisch die Verknüpfung aller Elemente miteinander. Dadurch können Zusammenhänge zwischen verschiedenartigen Elementen dargestellt werden, welche die Verfolgbarkeit der Anforderungen ermöglichen. Die Integration einer Methode zur Zielmodellierung ist demnach eine Erweiterung, welche zielorientierte Elemente verfügbar macht. Somit kann UNICASE sowohl für die Erfassung von ersten groben Anforderungen als auch für detaillierte Anforderungen im späteren Verlauf der Anforderungsermittlung verwendet werden.

3.2 Anforderungen

Bei der Entwicklung einer Methode zur Zielmodellierung in Verbindung mit persönlichen Werten richten sich die meisten Anforderungen nach den Möglichkeiten, die Ansätze aus der Literatur (siehe Kapitel 2.3.1) bieten. Eine weitere Grundlage bildet die Analyse der Kritik an bisherigen Ansätzen in Kapitel 2.3.2. Da der Umfang dieser Arbeit bei weitem nicht ausreicht um auf einen Großteil der vorgestellten Kritikpunkte einzugehen, liefert diese Methode bezüglich der vorgestellten Kritik nur Verbesserungen in einzelnen Punkten. Neben Information aus den Grundlagen werden weitere Anforderungen aus der Evaluation von Werkzeugen zur Zielmodellierung von Matulevičius et al. [37] herangezogen. Auch die Autoren führten eine Literaturrecherche durch und ermittelten weitere Anforderungen durch die Anwendung bestimmter Frameworks zur Evaluation. Weitere Anforderungen ergeben sich aus den Fakten bezüglich des Nutzens einer Methode, der bereits im vorherigen Abschnitt erläutert wurde.

Die folgende Liste bietet eine Übersicht der gefundenen Anforderungen:

- **Modellierung von Stakeholdern**
  Neben der klassischen Methode Stakeholder in Form von Aktoren oder Agenten zu modellieren, soll eine detaillierte Möglichkeit zur Modellierung bestehen. So können Beziehungen zwischen Stakeholdern erstellt werden, worunter beispielsweise die Zugehörigkeit eines Stakeholders zu einer Gruppe von Stakeholdern zählt - vergleichbar mit Mitarbeitern einer Abteilung.
- **Modellierung verschiedener Zieltypen**
  Wie bei den bereits vorgestellten Methoden aus der Literatur können verschiedene Arten von Zielen modelliert werden. Es erfolgt generell eine Unterscheidung zwischen Zielen, deren Erfüllung klar ersichtlich ist und Zielen, bei denen die Messbarkeit des Erfüllungsgrades schwer oder nicht messbar ist - den Soft Goals.

- **Methoden zur Analyse des Zielmodells**
  Das definierte Zielmodell kann hinsichtlich Alternativen und Konflikten analysiert werden. Die Analyse der Alternativen bietet die Möglichkeit frühzeitig im Softwareentwicklungsprozess Entscheidungen treffen zu können. Entsprechend ist die Analyse hinsichtlich Konflikten möglich, die gemäß dem Modell auftreten können. Die Methoden sollen zudem praxistauglich sein und eine spätere Nachvollziehbarkeit, insbesondere bei der Evaluation von Alternativen, bieten.
- **Nutzung persönlicher Werte**
  Die Modellierung von persönlichen Werten der Stakeholder ist möglich. Die Methoden zur Analyse des Zielmodells nutzen die persönlichen Werte um die Bedeutung der Stakeholder und deren Präferenzen für den Prozess der Zielmodellierung hervorzuheben.
- **Integration mit anderen Modellen des Requirements Engineering**
  Für die Elemente des Zielmodells können Beziehungen zu anderen Elementen des Requirements Engineering definiert werden. Dadurch können Elemente des Zielmodells im weiteren Verlauf verfolgt werden.
- **Einfache Notation für Elemente des Zielmodells**
  Bezüglich der Notation des Zielmodells soll auf eine minimale Anzahl an Elementen geachtet werden. Ebenso sollen die Elemente gut erkennbar und auch ohne Unterstützung einer Software nutzbar sein.

Um anhand dieser Anforderungen eine Methode entwickeln zu können, sind zunächst verschiedene Vorarbeiten zu erledigen. Insbesondere für die Analyse von Zielmodellen hinsichtlich Alternativen und Konflikten müssen existierende Ansätze hinsichtlich ihrer Verwendbarkeit evaluiert werden. Die folgenden beiden Abschnitte widmen sich dieser Thematik. Es wurde jeweils eine Literaturrecherche durchgeführt und gefundene Ansätze wurden bezüglich verschiedener Kriterien evaluiert.

3.2.1 Auswahl von Alternativen

Zur Evaluation von Ansätzen zur Auswahl und Bewertung von Alternativen in Zielmodellen wurde eine Literaturrecherche durchgeführt. Schlüsselwörter für die Suche waren „goal-orientied“, „goal“, „requirements engineering“, „alternatives“, „options“ und Namen von bekannten Zielmodellen wie beispielsweise „KAOS“. Als Quellen dienten ACM[1], IEEE[2] und Google Scholar[3]. Die Menge der Ergebnisse variierte, bedingt durch Schlüsselwörter wie „goal“, das sehr oft verwendet wird, stark, zwischen $\approx$10 und $\approx$10000. Es wurden 25 potentiell interessante Veröffentlichungen identifiziert. Schließlich wurden aus diesen sieben als relevant eingestuft.

[1] http://portal.acm.org
[2] http://ieeexplore.ieee.org
[3] http://scholar.google.com

In den folgenden Abschnitten erfolgt zunächst eine Erklärung der Kriterien für die Evaluation der Ansätze. Anschließend werden die einzelnen Ansätze erklärt und entsprechend der Kriterien bewertet.

3.2.1.1 Kriterien

Die Kriterien zur Evaluation der Ansätze entstanden durch deren Überprüfung. Die extrahierten Kriterien dienen zur Unterscheidung und einer schnelleren Bewertung. Folgende Kriterien werden verwendet:

- Art des Ansatzes
- Benötigte Informationen
- Ablauf
- Ausgabe
- Automatisierbarkeit
- Detailgrad

Das erste Kriterium bezieht sich auf die beiden verschiedenen Arten, die unter den Ansätzen existieren: qualitativ und quantitativ. Bei der ersten Art werden qualitative Labels zur Bewertung von Alternativen verwendet, beispielsweise + oder –. Die zweite Art hingegen verwendet Werte wie 0.6 zur Bewertung von Alternativen. Die Kriterien 2-4 betrachten die allgemeine Funktionsweise des Ansatzes. Dazu gehören die notwendigen Eingaben damit der Ansatz funktioniert, die einzelnen Schritte des Ansatzes sowie die Ausgabe. Da ein geeigneter Ansatz auch in UNICASE funktionieren soll, ist die Automatisierbarkeit interessant. Sie wird mit hoch, mittel und niedrig bewertet. Eine hohe Automatisierbarkeit ist dann gegeben, wenn das Verfahren mit den initialen Informationen ohne oder nur mit geringen Benutzerinteraktionen auskommt. Eine niedrige Automatisierbarkeit zeugt dementsprechend von vielen notwendigen Benutzerinteraktionen. Das letzte Kriterium betrachtet den allgemeinen Detailgrad der Ansätze, in welchen sie sich oft unterscheiden. Damit ist zum Beispiel gemeint, dass auch beschrieben wird wie Gewichtungen objektiv ermittelt werden können. Die Ausprägung sind analog zur Automatisierbarkeit. Entsprechend ist ein Ansatz mit einem hohen Detailgrad sehr genau beschrieben, bei einem niedrigen Detailgrad bleiben wichtige Fragen offen.

3.2.1.2 Ansätze

Die sieben verschiedenen Ansätze werden im Folgenden jeweils kurz zusammengefasst. Die Tabellen 3.1 und 3.2 geben zudem einen Überblick bezüglich der zuvor definierten Kriterien.

Ansatz	Art des Ansatzes	Benötigte Informationen	Ablauf	Ausgabe	Automatisierbarkeit	Detailgrad
Liu und Yu [38]	qualitativ	Informationen von Stakeholdern	GRL-Zielmodell erstellen → UCM für Designalternativen → Qualitative Evaluierung → Verfeinerung	Vollständiges GRL-Zielmodell, UCM	nicht vorhanden	hoch
Xie et al. [39]	quantitativ	i*-Zielmodell	Gewichtung der Soft Goals → Relevanz von Tasks einschätzen durch Experten → Identifizierung Strategien → Evaluierung Strategien bzgl. Soft Goals	optimale Alternative für Aktor	hoch	mittel
Yamamoto et al. [40]	quantitativ	UND/ODER-Graph	Ermittlung von Soft Goals → Gewichtung → Berechnungsregeln für Teilaspekte → Berechnung beste Alternative	beste Alternative	hoch	niedrig
Kaiya et al. [41]	quantitativ	UND/ODER-Graph	Präferenzmatrix für Ziele → Bewertung Präferenzmatrix für Rollen → Auswahl der besten Alternative	beste Alternative	niedrig	niedrig

Tabelle 3.1: Ansätze zur Evaluation von Alternativen 1/2

Ansatz	Art des Ansatzes	Benötigte Informationen	Ablauf	Ausgabe	Automatisierbarkeit	Detailgrad
Liaskos et al. [42]	qualitativ / quantitativ	beliebiges Zielmodell	Zielmodell splitten → Gewichtung im optionalen Teil → qualitatives bewerten zwischen verbindlichem u. optionalem Teil → Evaluierung des Zielmodells	beste Alternative	mittel	hoch
Letier und Lamsweerde [43]	quantitativ	KAOS oder anderes Zielmodell	Wichtige Ziele identifizieren → Zielfunktionen, Zielvariablen definieren → Regeln für Propagierung der Werte festlegen → Bottom-Up-Analyse	beste Alternative	mittel	hoch
Lamsweerde [44]	qualitativ / quantitativ	KAOS oder anderes Zielmodell	Erfassen der Soft Goals → Qualitative oder quantitative (mit Gewichtung) Bewertung bzgl. Soft Goals → Bottom-Up-Analyse	beste Alternative	mittel	hoch

Tabelle 3.2: Ansätze zur Evaluation von Alternativen 2/2

Der Ansatz von Liu und Yu [38] vereint die Nutzung von Szenarien und Zielmodellen um möglichst strukturiert Designalternativen offenzulegen und zwischen diesen zu entscheiden. Zur Darstellung des Zielmodells wird GRL verwendet. Die Szenarien sind durch Use Case Maps (UCMs) dargestellt. Ausgangspunkt sind gesammelte Information wie Problembeschreibungen, Geschäftsziele, existierende Technologien und Prozesse. Parallel wird mit diesen Daten ein GRL-Modell und ein UCM-Modell erarbeitet. Das GRL-Modell umfasst neben wichtigen Stakeholdern (Aktoren) und deren sozialen Beziehungen zueinander (mit Abhängigkeiten) auch Ziele (Soft Goals). Das UCM-Modell stellt die konkreten Geschäftsprozesse dar. Iterativ wird das GRL-Modell verfeinert. Mögliche Alternativen zur Erfüllung von Zielen sind durch jeweils ein UCM-Modell dargestellt. Nach und nach wird versucht die besten Alternativen für den Stakeholder zu wählen. Dazu wird jede mögliche Designalternative anhand der Erfüllung der Soft Goals evaluiert. Diese Evaluation ergibt qualitative Labels im Zielmodell wie „Hurts“ oder „Helps“ für die möglichen Designalternativen. Im weiteren Verlauf wird jeweils die beste Alternative weiter verfeinert, bis keine weiteren Ziele und Designalternativen auftauchen. Das Endergebnis ist ein vollständiges Designdokument. Der Ansatz ist nicht automatisierbar, alle Schritte müssen manuell durchgeführt werden.

Xie et al. [39] stellen eine quantitative Methode zur Auswahl der besten Alternative vor. Sie verwenden dazu i* als Zielmodell und erweitern dieses um einige Notationen zur einfachen Benennung von i*-Elementen, wie Zielen oder Beziehungen. Für jeden Aktor kann nun eine optimale Strategie ermittelt werden. Eine Strategie ist eine bestimmte Kombination von Tasks, um alle Ziele und somit auch das primäre Ziel zu erfüllen. Jedes Soft Goal eines Aktors wird mit einem Gewicht belegt (zwischen 0 und 1, Summe aller Gewichte = 1). Durch die Befragung von Experten wird der Einfluss einer Task auf ein Soft Goal ermittelt (mit gewichteten mengenwertigen Abbildungen, Intervallschätzung). So kann mit einer Strategie ein Wert für die Erfüllung der Soft Goals berechnet werden. Durch die Summe all dieser gewichteten Werte kommt man schließlich zu einem Wert für den Nutzen der verwendeten Strategie. Die Berechnung der besten Alternative im Zielmodell kann vollständig automatisiert erfolgen, nachdem die Gewichtung und der Einfluss der einzelnen Tasks benannt wurde. Der Ansatz eignet sich auch für größere Zielmodelle mit mehreren Aktoren. Dabei verbindet man die verschiedenen Modelle durch Abhängigkeiten miteinander. Es wird dann eine optimale Systemstrategie gesucht: eine Strategie die aus denen der Aktoren besteht und das Systemziel erfüllt.

Einen ähnlich Ansatz verfolgen Yamamoto et al. [40] mit ihrem quantitativen Ansatz zur optimalen Auswahl aus alternativen Softwarekomponenten. Im ersten Schritt werden die Abhängigkeiten zwischen den Alternativen durch ein Zielmodell visualisiert. Das Zielmodell ist einfacher Graph mit UND/ODER-Relationen. Jeder Knoten bildet ein Ziel, weitere Elemente sind nicht vorhanden. Anschließend werden die Kriterien zur Evaluation der Alternativen ermittelt und gewichtet (zwischen 0 und 1, Summe ist 1); diese Kriterien entsprechen Soft Goals. Nun werden im dritten Schritt für Teilaspekte der Soft Goals Berechnungsregeln aufgestellt, beispielsweise „Datendurchsatz“ für das Soft Goal „Performanz“. Anhand dieser Berechnungsregeln können schließlich Präferenzwerte be-

rechnet werden. Für die Definition dieser Regeln existieren keine Vorgaben; sie werden an die Ziele im Modell attribuiert. Im finalen Schritt werden nun die Alternativen mit Hilfe der TOPSIS (Technique for Order Preference by Similarity to Ideal Solution)-Methode evaluiert um die Abstände der Alternativen zu der theoretisch besten Alternative zu berechnen. Das Ergebnis ist schließlich ein Präferenzwert für die Alternativen, wodurch eine optimale Alternative gewählt werden kann.

Kaiya et al. [41] stellen mit der AGORA (Attributed Goal-Oriented Requirements Analysis Method) eine weitere quantitative Methode vor zur Evaluierung von Alternativen im Zielmodell. Dazu werden einfache UND/ODER-Graphen um Attribute, die an die Knoten (Ziele) und Kanten (Abhängigkeiten zwischen Zielen) attribuiert werden können, erweitert. Die Ziele können mit einer Matrix versehen werden, welche die Präferenzen eines jeden Entscheidungsträgers umfasst (zwischen -10 und 10). Die Kanten werden mit einen Wert für den Beitrag den ein Ziel zum entsprechenden Ziel auf der nächsten Ebene schafft (zwischen -10 und 10) versehen. Nach dem initialen Aufbau des Zielmodells, der Verfeinerung, sowie der Attribuierung kann die optimale Alternative anhand der vergebenen Werte selektiert werden. Allerdings fehlt dem Ansatz eine genaue Beschreibung dieses Schrittes, wodurch der Ansatz kaum automatisierbar ist.

Der Ansatz von [42] unterscheidet sich von den anderen Ansätzen dadurch, dass das Zielmodell in einen notwendigen und einen optionalen Teil gesplittet wird. Diese Art von Aufteilung soll dafür sorgen, dass sich optionale Anforderungen besser erfassen lassen. Ziel des Ansatzes ist es, die Alternative im notwendigen Teil des Zielmodells zu selektieren, die den optionalen Teil des Zielmodells am besten unterstützt. Die beiden Teile des Zielmodells sind miteinander verbunden. Diese Verbindungen geben qualitativ an, inwiefern Ziele oder Tasks sich positiv oder negativ auf optionale Ziele auswirken. Diese können zusätzlich quantitativ gewichtet werden (zwischen 0 und 1, Summe = 1). Um eine automatisierte Berechnung der optimalen Alternativen durchzuführen erfolgt eine „Übersetzung“ des grafischen Zielmodells in HTN (Hierarchical Task Network) und PDDL 3.0. Dabei handelt es sich um bekannte Sprache im Bereich der Planungsprobleme. Der verbindliche Part wird in HTN und der optionale Part des Modells in PDDL 3.0 übersetzt. Die „Übersetzung“ erfolgt mit Tool-Support transparent, wodurch auch Personen ohne Wissen über HTN und PDDL 3.0 den Ansatz verwenden können.

Letier und Lamsweerde [43] stellten 2004 einen quantitativen Ansatz vor, der sich auch der Definition von objektiven Kriterien zur Evaluierung der Alternativen widmet. Voraussetzung für den Ansatz ist ein KAOS-Zielmodell, dessen wichtigste Ziele (Ziele auf der obersten Ebene, sowie wichtige Ziele auf unteren Ebenen) mit Zielfunktionen und Qualitätsvariablen versehen werden. Anhand dieser kann mit der Zielfunktion ein Wert für die vorhandene Abdeckung des Ziels ermittelt werden. Für die Identifizierung der Zielfunktionen und Qualitätsvariablen werden fünf Heuristiken vorgestellt. Die Zielfunktionen sind oft nicht leicht zu definieren, daher führen die Autoren die Möglichkeit zur Festlegung von Wahrscheinlichkeitsdichtefunktionen ein. Schließlich sind Regeln zur Weitergabe der ermittelten Abdeckungswerte notwendig. Diese sind Domänenspezifisch und

deren Definition sehr kritisch für den Gesamtprozess, weshalb die Autoren einen Katalog mit Mustern bereitstellen. Für jede Alternative erfolgt nun die Bottom-Up-Berechnung aller vorhandenen Zielfunktionen, wodurch schließlich ein Vergleich möglich ist. Die Berechnung kann vollständig automatisiert werden.

Aufgrund des sehr rechenintensiven Charakters stellte Lamsweerde [44] 2009 einen qualitativen und quantitativen Ansatz vor. Die Erfassung der möglichen Alternativen und Soft Goals an den Blättern des Zielmodells bildet den ersten Schritt. Für den qualitativen Ansatz werden die Soft Goals in eine Tabelle eingetragen und es wird evaluiert, inwiefern sich die Alternativen auf diese auswirken (++ sehr positiv, – sehr negativ etc.). Anschließend kann mit Hilfe von Regeln eine Bottom-Up-Analyse der Soft Goals für jede Alternative automatisiert durchgeführt werden. Der quantitative Ansatz bildet eine Erweiterung und bietet die Möglichkeit zur Berechnung der besten Alternative. Jedes Soft Goal erhält dazu ein Gewicht und wird bezüglich jeder Alternative durch einen Wert zwischen 0 und 1 bewertet. In diesem Schritt unterscheidet sich der Ansatz klar von den anderen Methoden. Lamsweerde stellt zur Berechnung dieser Werte die Formel 3.1 vor.

$$Score(opt, lsg) = 1 - \frac{|g_T - g_V|}{g_{max}} \tag{3.1}$$

Für jedes Soft Goal wird eine Metrik definiert, beispielsweise die Zeit bis zum Öffnen einer Datei für die Performanz. Bezüglich der Metrik legt man einen tolerierbaren oder akzeptablen Maximalwert g_{max}, beispielsweise 10 Sekunden, sowie einen Zielwert g_T, beispielsweise 0 Sekunden, fest. Für jede Alternative schätzt man nun den erwarteten Wert g_V. Bei einem geschätzten Wert von 4 Sekunden ergibt sich somit ein Wert von 0.6. Die Formel funktioniert auch für den Fall, dass größere Werte einen besseres Ergebnis darstellen. Dazu wird der Maximalwert g_{max} dem Zielwert g_T gleichgesetzt. Zu beachten ist lediglich, dass die Formel stets von einem besten bzw. schlechtesten Wert von 0 ausgeht. Ist dieser Wert jedoch >0, so muss eine Verschiebung der Eingabewerte vorgenommen werden. Die Schätzung ist durch die Festlegung einer Metrik verhältnismäßig objektiv und nachvollziehbar. Falls eine Alternative durch eine UND-Beziehung durch weitere Alternativen verfeinert wird, werden die Schätzwerte für die Kriterien addiert und bilden die Schätzung für das verfeinerte Element. Bottom-Up erfolgt jeweils die Berechnung der besten Alternative auf jeder Ebene. Somit ergibt sich pro Ebene eine beste Alternative. Lamsweerde zeigt dabei, dass der quantitative dem qualitativen Ansatz überlegen ist. Dies äußert sich hauptsächlich in der Aussagekraft des Ergebnisses. Der qualitative Ansatz liefert lediglich ein Label. Im vorgestellten Beispiel wurde gezeigt, dass ein häufig erhaltenes Ergebnis „undetermined“ darstellt. Durch die Hinzunahme von konkreten Werten für die Soft Goals zeigt sich die bessere Aussagekraft der quantitativen Methode. Allerdings bleibt zu bemerken, dass dazu wesentlich mehr Informationen notwendig sind. Die qualitative Methode eignet sich somit für eine grobe Evaluierung, beispielsweise bei einer ebenso grobe Spezifikation. Die quantitative Methode hingegen ist insbesondere für detaillierte Spezifikation gedacht.

3.2.1.3 Bewertung

Bei der Betrachtung der verschiedenen Ansätze fällt zunächst auf, dass die qualitativen Ansätze kaum vertreten sind. Dies ist vor allem durch die von Lamsweerde genannten Punkte bedingt - qualitative Ansätze eignen sich lediglich zu einer groben Abschätzung. Die quantitativen Ansätze hingegen lassen eine genaue Evaluierung zu. Jedoch besteht hier das Problem der Ermittlung der entsprechenden Werte. Lediglich Lamsweerde bietet eine Art der Unterstützung um solche Werte nachvollziehbar zu ermitteln. Andere Ansätze wie der von Xie et al. [39] verlassen sich auf die Einschätzung von Experten. Es bleibt allerdings offen, wie diese Einschätzung zu Stande kommt. Die quantitativen Ansätze bieten zusätzlich die Möglichkeit zur Gewichtung. Die Ermittlung der Gewichte wird in keinem Ansatz näher thematisiert.

Nahezu alle Ansätzen benötigen ein Zielmodell oder einen ähnlichen Graphen für den Ablauf und liefern als Ausgabe die beste Alternative im Modell. Lediglich der Ansatz von Liu und Yu [38] benötigt nur Informationen von Stakeholdern, da er bereits bei der Erstellung des Zielmodells angewandt wird. Dadurch können möglicherweise viele unnötige Pfade vermieden werden. Dieser Umstand kann aber auch dazu führen, dass andere alternative Pfade, die sich erst im späteren Verlauf ergeben würden, gar nicht entdeckt werden. Bei der Ausgabe generiert der Ansatz gar ein komplettes Designdokument.

Der Ablauf gestaltet sich bei vielen Ansätzen ähnlich. Neben der bereits vorgestellten Ausnahme von Liu und Yu [38] ist noch das Vorgehen von [42] hervorzuheben. Sie teilen das Zielmodell in einen notwendigen und optionalen Teil auf und evaluieren anhand der Unterstützung durch den optimalen Teil die Alternativen im notwendigen Teil. Dieses Vorgehen unterscheidet sich klar von den anderen Ansätzen, bei denen die Erfüllung von Soft Goals evaluiert wird.

Die Automatisierbarkeit unterscheidet sich zwischen den Ansätzen recht stark. Bei Liu und Yu [38] ist sie nicht vorgesehen, da nur qualitative Bewertungen vorgenommen werden und die Stakeholder manuell die beste Alternative wählen müssen. Xie et al. [39] und Yamamoto et al. [40] sind zwar sehr gut automatisierbar, allerdings handelt es sich um verhältnismäßig simple quantitative Techniken. Die weiteren Ansätze unterstützen zwar die automatisierte Berechnung von Alternativen, benötigen aber viele manuell zu erstellende Informationen.

Der Detailgrad ist bei der Mehrheit der Ansätze hoch. Lediglich Yamamoto et al. [40] und [41] fallen negativ auf. Dies äußert sich in fehlender Unterstützung bei der Bewertung und späteren Selektion der Alternative.

Zusammenfassend wirkt der Ansatz von Lamsweerde [44] am ausgereiftesten und durchdachtesten. Die Ermittlung der Werte für die Erfüllung der Soft Goals ist nachvollziehbar, wird automatisch dokumentiert und hat einen klaren Bezug zu den Anforderungen des Systems. Zumal die schließliche Berechnung der besten Alternative zu einem gewissen Umfang automatisierbar ist.

3.2.2 Erkennen und Auflösen von Konflikten

Um Ansätze zum Erkennen und Auflösen von Konflikten zu finden, wurde ebenfalls eine Literaturrecherche durchgeführt. Die Wahl der Schlüsselwörter gestaltete sich ähnlich wie bei der Suche nach Ansätzen zur Bewertung von Alternativen. Statt „alternatives“ und „options“ wurde nach „conflict“ gesucht. Die Ergebnismenge variierte auch hier sehr stark. Jedoch fiel die Menge der potentiell interessanten Veröffentlichungen mit zehn geringer aus. Schließlich wurden vier Veröffentlichungen als relevant eingestuft. Die Quellen für die Recherche stimmen mit denen der vorherigen Recherche überein.

Wie im Abschnitt zuvor erfolgt auch hier eine Erklärung der Kriterien, der Ansätze und eine entsprechende Bewertung.

3.2.2.1 Kriterien

Die Kriterien für die Evaluation der Ansätze zum Erkennen und Auflösen von Konflikten unterscheiden sich nur geringfügig von denen in Kapitel 3.2.1.1. Lediglich das Kriterium für die Art des Ansatzes ist hier nicht vertreten, da eine solche Unterscheidung keinen Sinn macht. Stattdessen ist eine Betrachtung des Kriteriums „Auflösung“ interessanter. Dieses Kriterium betrachtet ob ein Ansatz die Auflösung von erkannten Konflikten unterstützt. Wichtig ist auch, dass das Kriterium Automatisierbarkeit nicht nur die Erkennung von Konflikten betrachtet, sondern auch dessen Auflösung.

3.2.2.2 Ansätze

Die vier relevanten Ansätze werden in diesem Abschnitt jeweils kurz vorgestellt. Eine Übersicht bezüglich der Kriterien ist durch Tabelle 3.3 gegeben.

Der Ansatz von Chung [45] befasst sich mit Interessenskonflikten verschiedener Rollen in i*-Zielmodellen. Die Analyse auf Konflikte im Zielmodell erfolgt durch paarweisen Vergleich von Zielen zweier oder mehrerer Rollen statt. Es wird jeweils ein Ziel auf einer niedrigen Abstraktionsebene einer Rolle mit einem Ziel auf einer höheren Abstraktion einer anderen Rolle verglichen. Sobald ein Konflikt entdeckt wurde, folgt die Auflösung mit Hilfe des NFR-Frameworks. Die Nutzung des NFR-Frameworks für die Bewältigung von Konflikten wird damit begründet, dass hier eine passende und gut verständliche Notation vorhanden ist. Die Funktionsweise der Auflösung ist nicht genauer spezifiziert. Lediglich die Definition von vermittelnden Zielen wird genannt. Aufgrund der Art und Weise wie Konflikte detektiert werden, ist eine Automatisierung dieser Methode nicht möglich.

Auch Giorgini et al. [46] betrachten Konflikte, die durch konkurrierende berufliche oder persönliche Interessen entstehen. Als Zielmodell wird Secure Tropos (Erweiterung von Tropos) genutzt und um die Möglichkeit erweitert, nicht nur Zuwendungen für Ziele

Ansatz	Benötigte Informationen	Ablauf	Ausgabe	Automatisierbarkeit	Detailgrad	Auflösung
Chung [45]	i*-Zielmodell	paarweiser Vergleich von Zielen verschiedener Rollen → Aufbau von NFR-Modell für Konflikte → Auflösung	Konflikte zwischen Zielen verschiedener Rollen oder Zielmodell ohne Konflikte	nicht möglich	mittel	Ja
Giorgini et al. [46]	TROPOS oder anderes Zielmodell	Prüfung des Zielmodells mit Regeln	Konflikte zwischen Zielen verschiedener Rollen	mittel	niedrig	Nein
Kaiya et al. [41]	UND/ODER-Graph	Präferenzmatrix für Ziele → Bewertung Präferenzmatrix für Rollen → Überprüfung Varianz der Matrix und negative Auswirkungen auf direkt miteinander verbundenen Ziele	Konflikte im Zielmodell oder verfeinertes konfliktfreies Zielmodell	hoch	niedrig	Ja
Lamsweerde et al. [16]	KAOS-Zielmodell mit formaler Definition	Aufstellen von „boundary conditions“ → Überprüfen auf Konfliktpattern → Auflösen des Konflikts	Konflikt oder konfliktfreies Zielmodell	niedrig	hoch	Ja

Tabelle 3.3: Ansätze zum Erkennen und Auflösen von Konflikten

aus der Sicht eines Aktors, sondern für das gesamte System zu analysieren. Zur Erkennung der betrachteten Konfliktart gibt es fünf Regeln. In Form von Datalog, einer Sprache für Fakten und Regeln, nutzt man die Regeln um eine automatisierte Erkennung zu ermöglichen. Die Lösung von Konflikten ist nicht weiter ausgeführt, da hier viele nicht erfassbare Faktoren wichtig sind, beispielsweise Zuständigkeiten. Die Autoren verweisen außerdem auf ein Werkzeug zur Generierung von entsprechendem Datalog Code aus grafischen Zielmodellen, die Schritte hierfür sind aber nicht aufgeführt.

Die bereits bei den Ansätzen zur Auswahl von Alternativen vorgestellte Methode AGORA beinhaltet auch die Erkennung und das Auflösen von Konflikten. Kaiya et al. [41] nutzen dazu die bereits vorgestellten Präferenzwerte, die in Form einer Matrix bei den Zielen hinterlegt werden. Konflikte ergeben sich hier durch negative Werte zwischen direkt verbundenen Zielen oder durch einen hohen Varianzwert in der Matrix der Präferenzwerte. Demnach lässt sich die Methode auch automatisiert durchführen. Eine hohe Varianz in der Matrix spricht für das Vorhandensein von Verständnisproblemen zwischen Entscheidungsträgern. Die Lösung von Konflikten kann durch eine weitere Verfeinerung erreicht werden oder durch die Aufnahme von Verhandlungen zwischen Entscheidungsträgern.

Ein formaler Ansatz zum Erkennen und Auflösen von Divergenzen stammt von Lamsweerde et al. [16]. Damit sind beispielsweise konkurrierende Ziele gemeint. Als Basis für die Methode muss ein KAOS-Zielmodell mit vollständiger formaler Definition vorhanden sein. Nun beschreibt der Ansatz zwei formale Techniken, die genutzt werden, um Konflikte zu detektieren. Die erste Technik nutzt die Rückwärtsverkettung der Annahmen zu Zielen. Dabei wird davon ausgegangen, dass die Negation einer Annahme für ein Ziel einen Konflikt auslöst. Jede Annahme in Zielen bildet nun eine Art Randbedingung. Es erfolgt die Suche nach Zielen, deren negierte Randbedingung genau der Randbedingung eines anderen Zieles entspricht, wodurch ein Konflikt entdeckt wurde. Die zweite Technik basiert auf Mustern, die eingesetzt werden, um innerhalb der Randbedingungen mögliche Konflikte zu erkennen. Durch den ausgeprägten formalen Charakter der Methode ist eine Automatisierung weitgehend möglich. Allerdings funktioniert das Vorgehen nur, wenn alle Annahmen auch sauber und korrekt spezifiziert wurden.

3.2.2.3 Bewertung

Die Ansätze zum Erkennen und Auflösen von Konflikten unterscheiden sich stark bezüglich der Konfliktarten, die sie erkennen können. Während Lamsweerde et al. [16] viele verschiedene Arten von Konflikten erkennen können, konzentrieren sich die anderen Ansätze auf Interessenskonflikte. Diese können durch persönliche Interessen entstehen, beispielsweise durch die Ausübung einer bestimmten Rolle oder durch anderweitige Präferenzen.

Alle Ansätze beginnen direkt mit der Verarbeitung eines Zielmodells, das Vorgehen ist durch die unterschiedlichen Konfliktarten aber verschieden. Chung [45] und Giorgini et al. [46] spezialisieren sich beide auf rollenbasierte Interessenskonflikte. Bei dieser Art

von Konflikten steht der Vergleich von Zielen verschiedener Rollen im Vordergrund. Im Bereich der anderweitigen Interessenkonflikte nutzen Kaiya et al. [41] den im vorherigen Kapitel vorgestellten Ansatz mit einer Präferenzmatrix. Dabei reichen einfache Berechnungen mit der Matrix aus, um Aussagen über Konflikte treffen zu können. Der Ansatz von Lamsweerde et al. [16] ist durch seine allgemeine Konflikterkennung grundverschieden. Hier wird hauptsächlich mit formaler Logik gearbeitet.

Bis auf einen Ansatz wird auch die Auflösung der gefundenen Konflikte beschrieben. Die allgemein geringe Automatisierbarkeit zeigt sich vor allem hier. Die Auflösung besteht meist aus der manuellen Verfeinerung des Zielmodells. Chung [45] schlägt hierzu die Übertragung der Konfliktelemente in das NFR-Modell vor. Mit Hilfe der qualitativen Bewertung im NFR-Modell soll ein leichterer Zugang zur Lösung der Konflikte möglich sein. Auf Basis von formalen Regeln beschreiben auch Lamsweerde et al. [16] eine Methode zur Auflösung. Diese besteht in der Modifikation der Randbedingungen der Ziele und der Verfeinerung des Modells.

Der Detailgrad ist teilweise nur niedrig. Oft werden die Konfliktarten selbst und die Heuristiken zur Erkennung nur kurz beschrieben. Lediglich Lamsweerde et al. [16] bieten eine ausführliche Beschreibung.

Zusammenfassend zeigte sich, dass nur der Ansatz von Lamsweerde et al. [16] wirklich umfassend ist und auch gut beschrieben wird. Hinzu kommt, dass die Methode zumindest zu einem Teil automatisierbar ist. Gerade beim Aufspüren von Konflikten kann beispielsweise ein paarweiser Vergleich aller Ziele der verschiedenen Rollen - wie er von Chung [45] vorgeschlagen wird - kaum durchgeführt werden. Die Methode von Lamsweerde et al. [16] erscheint allerdings auch nicht wirklich praktikabel, da durch die formale Natur nur Konflikte erkannt werden können, falls auch entsprechende Vorbedingungen formal korrekt spezifiziert wurden. Für ein umfassendes Zielmodell ist dies allerdings sehr aufwendig und höchst unwahrscheinlich.

3.3 Werkzeuge zur Zielmodellierung

Eine Übersicht zu den auf dem Markt vorhandenen Werkzeugen zur Zielmodellierung soll Aufschluss zu folgenden Fragen geben:

1. Wie viele Werkzeuge existieren?
2. Welche Zielmodelle werden unterstützt?
3. Welche Möglichkeiten bieten diese Werkzeuge?
4. Können die Werkzeuge weitere Elemente des Requirements Engineering integrieren?
5. Welche Werkzeuge werden in der Industrie eingesetzt?

Bei der Suche nach Werkzeugen zur Zielmodellierung wurde nach Veröffentlichungen über Google Scholar und allgemein bei Google gesucht. Typische Suchwörter waren „goal“, „model“, „case tool“, kombiniert mit den Namen von bekannten Zielmodellen wie beispielsweise „KAOS“. Auf diese Art und Weise konnten insgesamt 13 verschiedene Werkzeuge zur Zielmodellierung gefunden werden. Allerdings handelte es sich dabei bei einem Großteil um veraltete Prototypen, die im Zuge einer Veröffentlichung entstanden sind. Das führte zu der Erkenntnis, dass im Grunde für jedes Zielmodell mindestens ein Werkzeug existiert. Ein Großteil der Funktionalität dieser Prototypen ging jedoch in größere Projekte über. Diese größeren Projekte, die nach wie vor noch aktiv weiterentwickelt werden, werden in diesem Abschnitt genauer betrachtet. Es handelt sich dabei um OpenOME, jUCMNav und Objectiver.

OpenOME ist ein auf Eclipse basierendes Open-Source-Werkzeug zur Zielmodellierung für i*, NFR sowie GRL. Es handelt sich um eine Weiterentwicklung von Organization Modelling Environment (OME) und wird an der Universität Toronto entwickelt [47]. Die Hauptfunktion von OpenOME bildet das grafische Modellieren von Zielmodellen. Diese Modelle können dann durchsucht und analysiert werden. Bei der Analyse steht vor allem das Top-Down- bzw. Bottom-Up-Reasoning im Vordergrund. Darunter versteht man im Wesentlichen die systematische Propagierung von qualitativen Labels in eine bestimmte Richtung. So kann beispielsweise überprüft werden, wie sich Labels an den Blättern des Zielmodells auf die oberen Elemente auswirken. Weiterhin sind auch Analysen basierend auf den Metadaten der Elemente möglich. So können auch Abfragen formuliert werden, wobei dies kaum dokumentiert ist. Eine Integration weiterer Elemente des Requirements Engineering ist nicht möglich. So können lediglich die von i*, NFR und GRL bekannten Elemente verwendet werden. Über den Einsatz von OpenOME im industriellen Umfeld konnten keine Informationen gefunden werden. Allerdings verzeichnet das Projekt nur sehr wenige Downloads der aktuellen Version (< 10), was darauf schließen lässt, dass es nur einen sehr kleinen Nutzerkreis gibt.

Das zweite Werkzeug jUCMNav ist, wie OpenOME open-source und basiert ebenfalls auf Eclipse. Es wird an der Universität Ottawa entwickelt und eignet sich zur Erstellung von Modellen der User Requirements Notation (URN). Somit können Szenarien in Form von Use Case Maps und Zielmodelle mit GRL erstellt werden [48]. Weiterhin bietet auch jUCMNav vielfältige Möglichkeiten zur Analyse [49]. So können durch den Benutzer Strategien definiert werden, unter denen man eine mögliche Konfiguration des Zielmodells versteht. Das Werkzeug unterstützt drei Arten der Evaluation des Zielmodells: quantitativ, qualitativ sowie eine Mischung der beiden Methoden. Weiterhin können in jUCMNav auch sogenannte „Key Performance Indikatoren“ verwendet werden. Diese Indikatoren können in Strategien zur Evaluation der Auswirkungen auf das Zielmodell verwendet werden. Weiterhin ermöglicht das Werkzeug die Definition von semantischen Prüfroutinen [48]. Durch Nutzung der Object Constraint Language (OCL) können Checks für das erstellte Zielmodell erstellt werden. Eine Integration von weiteren Elementen des Requirements Engineering ist auch hier nicht möglich. Es gibt lediglich die Option Elemente aus dem GRL-Graphen mit Use Case Maps zu verbinden. Der Einsatz im industri-

ellen Umfeld ist auch hier nicht bewertbar. Das Werkzeug wirkt jedoch wesentlich weiter entwickelt als OpenOME, zumal laut [48] 10 Entwickler an dem Projekt arbeiten und bisher $1,9 Millionen Kosten angefallen sind.

Objectiver ist ein kommerzielles Werkzeug zur Erstellung von Anforderungsbeschreibungen in Verbindung mit KAOS Zielmodellen [50]. Der Hersteller der Software bietet neben dem Werkzeug auch Trainings zur KAOS-Methode und Consulting an. Mit dem Werkzeuge können grafisch Zielmodelle wie auch formal, Zielmodelle definiert werden. Auch Objectiver bietet durch FAUST ähnliche Methoden zur Analyse wie OpenOME und jUCMNav [51]. Mit FAUST können Modelle beispielsweise auf Konflikte hin analysiert werden. Weiterhin ermöglicht FAUST die vollständige formale Überprüfung des Zielmodells auf Fehlerfreiheit. Neben den Analysen unterstützt Objectiver auch die direkte Verknüpfung mit anderen Elementen des Requirements Engineerings. So können Diagramme oder andere wichtige Artefakte direkt mit Objectiver verwaltet werden [51]. Denn neben der Funktionalität zur Erstellung von KAOS-Zielmodellen besteht der Fokus auf der Erstellung von Anforderungsbeschreibungen. Objectiver ermöglicht dazu die Erzeugung einer IEEE 830-kompatiblen Beschreibung der Anforderungen. Nach [50, 51] wurde Objectiver bereits in vielen Industrieprojekten eingesetzt, beispielsweise in der Telekommunikationsindustrie.

Bei der Analyse der Werkzeuge zur Zielmodellierung zeigte sich, dass es mit Objectiver, nur einen Vertreter, der auch nachweislich in der Industrie eingesetzt wird. Zwar unterstützen alle Werkzeuge die Analyse von Zielmodellen, jedoch bietet nur Objectiver Methoden zur Erkennung von Konflikten und nur mit Objectiver können andere Elemente des Requirements Engineering leicht integriert werden.

Auch Matulevičius et al. [37] betrachteten den Markt für Werkzeuge zur Zielmodellierung und führten eine ausführliche Evaluation durch. Betrachtet wurden schließlich Objectiver und OME, der Vorläufer zu OpenOME. Zur Durchführung der Evaluation wurden Studenten in verschiedene Gruppen eingeteilt, die jeweils für zwei Wochen mit einem der Werkzeuge arbeiteten. Die Studenten evaluierten schließlich das jeweilige Werkzeug bezüglich gegebener relevanter Anforderungen. Diese ermittelten die Autoren durch Literaturrecherchen sowie durch die systematische Anwendung von Frameworks zur Evaluation, beispielsweise R-TEA. Die Ergebnisse zeigten, dass Objectiver gegenüber OME besser abschneidet. Als Hauptvorteile werden Punkte wie bidirektionale Links der Elemente, verschiedene Sichtweisen auf das Modell sowie Wiederverwendbarkeit genannt. Jedoch kann OME bei der einfachen Verständlichkeit gegenüber Objectiver punkten. Allgemein wird kritisiert, dass die Werkzeuge nur unzureichende Möglichkeiten zur Erstellung einer vollständigen Anforderungsbeschreibung zur Verfügung stellen. Allerdings bietet Objectiver mittlerweile entsprechende Mittel.

4 Definition einer Methode zur Nutzung persönlicher Werte in Zielmodellen

Nachdem im vorherigen Kapitel die Anforderungen genauer betrachtet wurden, folgt nun die Nutzung der zuvor gefundenen Ergebnisse um eine Methode zu definieren, welche die gestellten Anforderungen erfüllen kann. Die Beschreibung der Elemente der Methode, sowie deren Notation bildet den Anfang des Kapitels. Im anschließenden Abschnitt wird die Nutzung der Methode genauer erläutert. Die Auswahl von Alternativen und das Erkennen und Auflösen von Konflikten bilden die Grundpfeiler der Methode und werden jeweils in einem Abschnitt betrachtet. Dabei steht zum einen die allgemeine Funktionsweise der Methode im Fokus und zum anderen die Nutzung persönlicher Werte.

4.1 Elemente der Methode

Die Elemente der Methode orientieren sich an den bereits vorgestellten Ansätzen zur Zielmodellierung, entsprechen jedoch nicht exakt einer bereits existierenden Methode. Ein Grund dafür ist die Nutzung des Metamodells zur Verknüpfung von aufgaben-, prozess- und zielorientierten Elementen in UNICASE. Dieses wird ausführlich in Kapitel 5.4 betrachtet. Ähnliches gilt auch für Notation der Elemente. Die folgenden Abschnitte geben einen Überblick zu den Elementen der Methode, deren Metamodell in Abbildung 4.1 dargestellt ist.

4.1.1 Ziele und Lösungen

Die verschiedenen Zielelemente basieren auf dem bereits erwähnten Metamodell zur Integration verschiedener Elemente. Im Wesentlichen orientieren sie sich aber an bestehenden Anätzen des zielorientierten Requirements Engineering. In Kapitel 2.1 wurde bereits aufgegriffen, dass in der Regel eine Unterscheidung zwischen Goals und Soft Goals stattfindet. Ein Goal zeichnet sich durch die mögliche Quantifizierbarkeit aus. Im Rahmen der Methode existiert dafür - angelehnt an KAOS [11] - das Element Achievement Goal, welches einen gewünschten Zustand eines Systems beschreibt. Im Gegensatz zu anderen Ansätzen der Zielmodellierung werden Soft Goals feiner unterschieden. Generell gibt es als Gegensatz zum Achievement Goal ein Property Goal. Es bezieht sich explizit auf eine gewünschte Eigenschaft eines Systems, beispielsweise eine Qualität. Eine spezielle

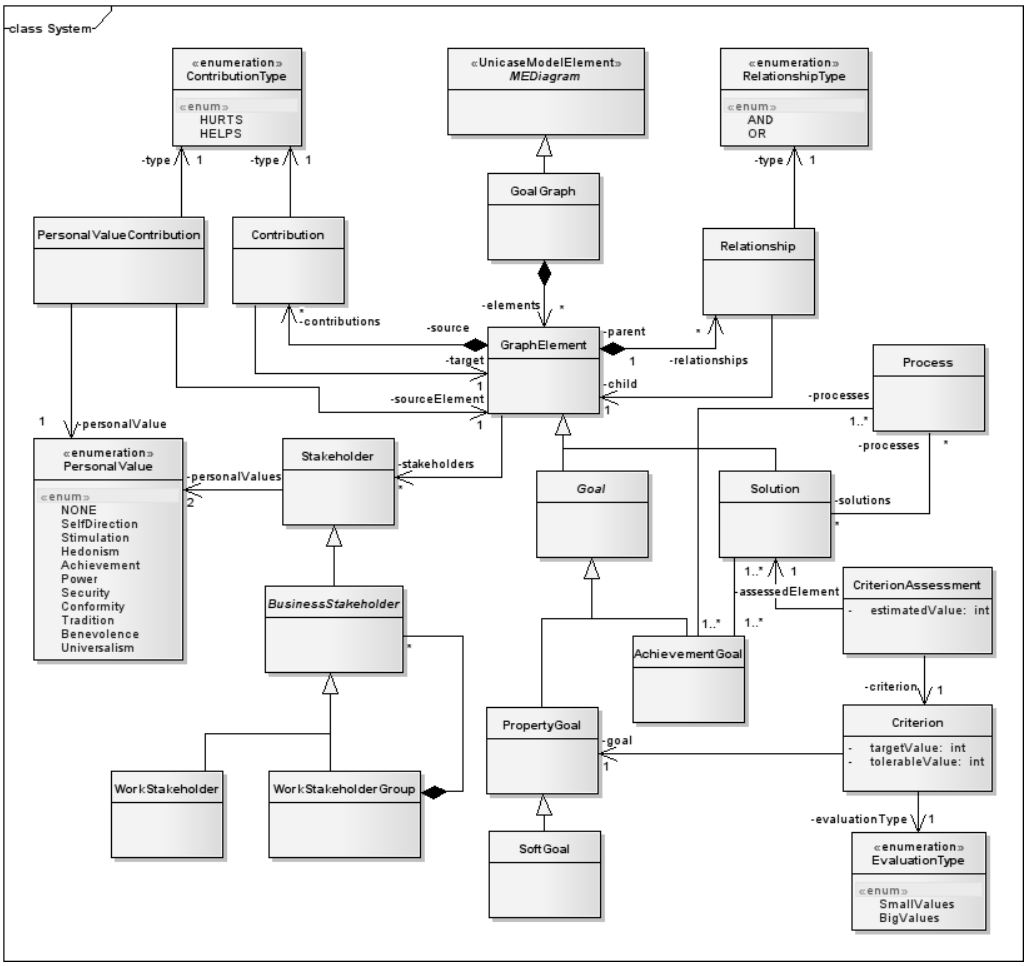


Abbildung 4.1: Die Elemente der Methode

Version des Property Goals ist schließlich das Soft Goal. Die Definition des Soft Goals gleicht der von anderen Ansätzen - es bezieht sich wie ein Property Goal auf eine bestimmte Qualität, aber es fehlen messbare Kriterien für die Erreichung des Ziels. Diese zusätzliche Unterscheidung zwischen Property und Soft Goal ermöglicht eine genauere Definition des Zielmodells, da zwischen messbarer und nur in gewissem Maße messbarer Erfüllung von Qualitätseigenschaften unterschieden wird. Ein Beispiel für ein Property Goal könnte sein: „Eine Anfrage muss in 5 Sekunden durch das System abgearbeitet sein“, ein Soft Goal hingegen: „Das System muss benutzerfreundlich gestaltet sein“.

In vielen Ansätzen zur Zielmodellierung (KAOS,i*) existiert ein Task-Element, welches notwendige Tätigkeiten beschreibt um ein Ziel erfüllen zu können. Jedoch hat dieses Element oft den Charakter einer Lösung. Insbesondere wenn man Ansätze zur Evaluation von Alternativen in Zielmodellen betrachtet, stellen Tasks immer Lösungen zur Zielerfüllung dar. Da die Bezeichnung Task somit irreführend ist, nutzt die Methode ein Solution-Element. Dabei kann es sich um jede Art von Möglichkeit zur Erfüllung eines

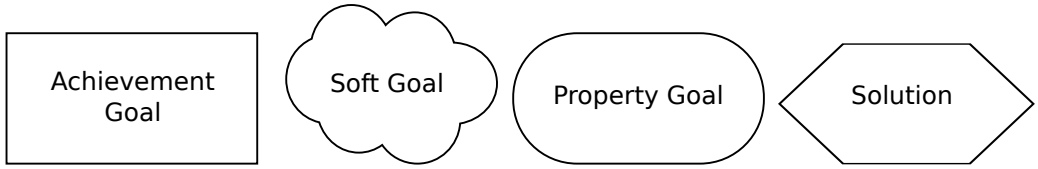


Abbildung 4.2: Notation der verschiedenen Ziele und Lösungen

Ziels handeln. Wie genau die Ausführung schließlich aussieht kann durch eine Reihe von Process-Elementen spezifiziert werden. Solution-Elemente sind stets als Erfüllung von Achievement Goals zu sehen. Sie stellen eine Art Container für die frühe Phase der Zielmodellierung dar. Die genaue Definition von Process-Elementen ist anfangs kaum möglich, das Solution-Element kann genau dann genutzt werden.

Abbildung 4.2 verdeutlicht die Notation der beschriebenen Elemente. Die Notation für das Soft Goal orientiert sich an der des NFR-Framework [18], die des Solution Elements an GRL [10] und i* [2] (Task). Es wurde darauf geachtet einen möglichst hohen Erkennungswert zu schaffen. Ebenso diente aber auch die Kritik in Kapitel 2.3.2 zur Auswahl der Formen. So sind die Elemente leicht unterscheidbar und lassen sich auch von Hand einfach zeichnen.

4.1.2 Stakeholder

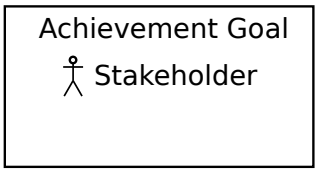


Abbildung 4.3: Notation der Stakeholder

Die Stakeholder-Elemente basieren, wie die Ziele, auf dem Metamodell zur Integration verschiedener Elemente. In bekannten Methoden zur Zielmodellierung existiert oft nur ein Aktor oder Agent. Die genaue Modellierung von Stakeholdern ist nicht möglich. In dieser Methode können Stakeholder, Work Stakeholder sowie Work Stakeholder Groups genutzt werden. Über das Stakeholder-Element lassen sich Aktoren modellieren, die an jeder Art von Prozess teilnehmen können. Dabei muss es sich nicht um zukünftige Benutzer des Systems handeln. Ein Work Stakeholder hingegen ist ein Benutzer des zu entwickelnden Systems. Unter einer Work Stakeholder Group versteht man beispielsweise eine Abteilung, die aus Work Stakeholdern besteht. Somit ist es möglich organisatorische Zusammenhänge zu modellieren.

Die Stakeholder können mehreren Goal- bzw. Solution-Elementen zugeordnet werden. Die Zuordnung erfolgt grafisch direkt beim entsprechenden Element, wie Abbildung 4.3 illustriert. Im Gegensatz zu Methoden wie i* [2], wo die Zuordnung von Zielen zu Stakeholdern erfolgt, hat dies den Vorteil, dass Ziele nicht mehrfach auftauchen. Bei i* kann ein Ziel, das für zwei Stakeholder gilt nur durch jeweils ein Ziel pro Stakeholder ausgedrückt werden, die dann durch eine Ressource miteinander verbunden sind.

4.1.3 Persönliche Werte

Abbildung 4.4: Persönliche Werte

Persönliche Werte besitzen einige Merkmale, die bei der Verwendung in einer Methode zu berücksichtigen sind. Diese sind zunächst, wie der Name schon sagt, immer einer Person bzw. einem Stakeholder zugeordnet. Hinzu kommt der Umstand, dass persönliche Werte nur schwer in Beziehung mit klassischen Zielen in Zielmodellen zu bringen sind. Der persönliche Wert eines Stakeholder gibt zwar Auskunft über seine „höheren“ Ziele (siehe Kapitel 2.2), jedoch handelt sich eher um eine Art Präferenz oder Leitlinie, die allenfalls durch Ziele im Modell positiv oder negativ beeinflusst werden kann. Eine direkte Zuordnung der persönlichen Werte zu Stakeholdern ist im Zielmodell aber nur schlecht zu verdeutlichen. Das liegt daran, dass, bedingt durch die vorgestellte Notation im vorherigen Abschnitt, die Stakeholder im Modell mehrfach auftauchen. Somit werden die persönlichen Werte für Stakeholder zwar ermittelt und festgelegt, im Modell stehen die Werte aber nur bei Goal- und Solution-Elementen. Es kann somit verdeutlicht werden inwiefern sich ein Element auf einen persönlichen Wert auswirkt. In Abbildung 4.4 ist zu sehen, dass positiv beeinflusste Werte am entsprechenden Element grün, negativ beeinflusste Werte entsprechend rot dargestellt sind.

Die möglichen persönlichen Werte orientieren sich an dem Modell von Schwartz et al. [1], das bereits in Kapitel 2.2 vorgestellt wurde. Die Ermittlung erfolgt mit einem Fragebogen, dem sogenannten „Schwartz Value Survey“ [1] und richtet sich nach dem Vorgehen von Proynova et al. [5]. Diese Methode ermittelt je die beiden Werte mit der höchsten Signifikanz für einen Stakeholder. Somit können einem Stakeholder maximal zwei persönliche Werte zugeordnet werden.

4.1.4 Relationen

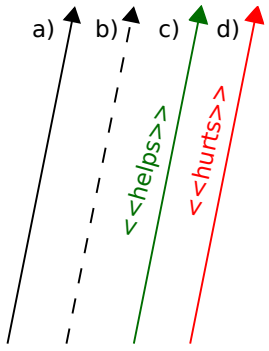


Abbildung 4.5: Beziehungen

Das Relationship-Element bietet die Möglichkeit zur Verfeinerung von Goal bzw. Solution-Elementen auf zwei verschiedene Arten. Wird ein Element durch eine UND-Relation (Abbildung 4.5, a) verfeinert, so müssen alle dem Element untergeordneten Elemente erfüllt sein, bevor das Elternelement erfüllbar ist. Bei einer ODER-Relation (b) muss lediglich eines der untergeordneten Elemente erfüllt sein. Da die Relationen nicht gemischt sein dürfen, ist es nicht möglich, dass ein Element durch UND-Relationen und ebenso durch eine oder mehrere ODER-Relationen verfeinert wird.

Zur Darstellung von positiven (c) bzw. negativen (d) Auswirkungen eines Goal- bzw. eines Solution-Elements auf andere Elemente können Contributions genutzt werden. Diese bieten die Möglichkeit zur qualitativen Bewertung von Elementen zueinander und eignen sich vor allem zur groben Abschätzung von Auswirkungen.

4.1.5 Elemente zur Bewertung

Um die Bewertung von Solution-Elementen zu ermöglichen, existieren die Criterion Assessment- und Criterion-Elemente. Für jedes relevante Property bzw. Soft Goal können durch die Criterion-Elemente bewertbare Kriterien erstellt werden. Somit lassen sich Solution Elemente bezüglich dieser definierten Kriterien bewerten. Die Elemente sind für die Analyse der Alternativen im Zielmodell relevant. Auf die genaue Funktionsweise wird in Kapitel 4.3 eingegangen. Da die Elemente nicht grafisch dargestellt werden, existiert auch keine Notation.

4.2 Schritte zum Zielmodell

Die Schritte zur Definition eines Zielmodells orientieren sich sehr stark an dem allgemein etabliertem Vorgehen [9]. Maßgeblich für alle Schritte ist zunächst die Sammlung von Informationen, beispielsweise durch Interviews mit zukünftigen Nutzern oder der Analyse bestehender Dokumente oder Software. Anhand dieser Informationen können erste Stakeholder und Ziele bestimmt werden. Anschließend müssen die Ziele gemäß den Kriterien aus Kapitel 4.1.1 einem Zieltyp zugeordnet werden, ebenso die Stakeholder. Bei letzteren muss zusätzlich auf die Beziehungen geachtet werden, die zwischen Work Stakeholdern und Work Stakeholder Groups bestehen können. Nachdem die identifizierten Stakeholder den Zielen zugeordnet wurden, erfolgt die sukzessive Erweiterung des Zielmodells durch das Beantworten der „Warum“- und „Wie“-Fragen. Die „Warum“-Fragen liefern übergeordnete Ziele zu den bisherigen Zielen. Die „Wie“-Fragen dienen zur Verfeinerung, beispielsweise in weitere Ziele. Hier könnte aber auch ein Solution-Element stehen. Die Verknüpfung der Ziele bzw. Solutions kann über die vorgestellten Relationen (siehe Kapitel 4.1.4) geschehen. Die Zuordnung von Stakeholdern erfolgt auch für jedes neu gefundene Element. Ein Endkriterium für das Finden übergeordneter Ziele oder verfeinerter Elemente ist leider nicht definierbar, da es sehr stark vom konkreten Szenario abhängt. Außerdem ist die Granularität, insbesondere bei Solutions nicht eindeutig bestimmbar. Jedoch gilt es zu beachten, dass eine zielorientierte Methode in einer frühen Phase des Requirements Engineering genutzt wird und die gefundenen Ziele und Solution dementsprechend grob sein sollten.

Sobald die Elemente des Zielmodells und deren Beziehungen zueinander bekannt sind, kann mit der Erfassung der persönlichen Werte für jeden Stakeholder per Fragebogen begonnen werden (siehe Kapitel 4.1.3). Schließlich folgt die Untersuchung aller Elemente

hinsichtlich positiver bzw. negativer Auswirkungen auf diese Werte. Ebenso können nun die Auswirkungen von Elementen zueinander positiv bzw. negativ bewertet werden.

Das Zielmodell ist nach Durchführung der genannten Schritten vollständig definiert. Die Analyse des Zielmodells auf Alternativen und Konflikte wird in den nachfolgenden Abschnitten genau betrachtet.

4.3 Evaluation von Alternativen

Bei der Vorstellung der Ansätze zur Evaluation von Alternativen wurden qualitative wie auch quantitative Methoden beschrieben. Lamsweerde zeigten, dass qualitative Ansätze im Gegensatz zu quantitativen Ansätze oft eine geringe Aussagekraft bieten. Allerdings sind genaue Werte, wie sie zur Evaluation benötigt werden, schwer zu ermitteln. Nur der quantitative Ansatz von Lamsweerde bietet Unterstützung zur Erlangung konkreter Evaluationswerte durch klar definierte Kriterien, wodurch die Werte nachvollziehbar ermittelt werden können.

Die Nutzung von persönlichen Werten innerhalb dieser Methode gestaltet sich durch die Besonderheiten, die in Kapitel 2.2 genannt wurden, problematisch. Eine Bewertung der persönlichen Werte wie bei Soft Goals ist nicht möglich. Daher bietet sich zu deren Bewertung lediglich ein qualitativer Ansatz an. In Kapitel 4.1.3 wurde dazu bereits die Möglichkeit zur Bewertung von Elemente hinsichtlich persönlicher Werte vorgestellt. Falls der Benutzer eine Alternative negativ bezüglich einem persönlichen Wert eines relevanten Stakeholders bewertet hat, so ist die Alternative speziell zu kennzeichnen. Nur auf diese Art und Weise kann man mit Alternativen umgehen, die eventuell zu Konflikten führen könnten. Durch die fehlende Möglichkeit der quantitativen Bewertung ist eine Integration in die Berechnung des Wertes für die Alternative jedoch nicht möglich. Das vorgestellte Vorgehen für persönliche Werte gilt ebenso für die qualitativen Labels.

Die Einschränkung von Lamsweerde nur Soft Goals für die Evaluation zuzulassen, die direkt mit den Alternativen verbunden sind, wird nicht umgesetzt. Dies wäre eine starke Einschränkung der Freiheitsgrade für das Zielmodell. Stattdessen muss das Zielmodell mindestens ein Property oder Soft Goal enthalten, wo dieses genau steht ist nicht relevant. Das als Ausgangspunkt für die Evaluation gewählte Element muss durch mindestens eine ODER-Relation und mindestens ein Solution-Elementen verfeinert sein. Die Methode besteht aus zwei wesentlichen Abschnitten. Zunächst müssen Kriterien zur Evaluation der relevanten Property und Soft Goals definiert werden und die verschiedenen Solution-Elemente bezüglich dieser bewerten. Anschließend kann die Berechnung der Ergebnisse für die Alternativen beginnen.

Konkret werden folgende Schritte ausgeführt:

1. Es wird mindestens ein Kriterium für mindestens ein Property oder Soft Goal definiert. Ein Kriterium umfasst eine Beschreibung, einen tolerierbaren Wert und einen Zielwert sowie eine Evaluationsart. Dabei wird festgelegt, ob große oder kleine Werte dem Optimum entsprechen. Weiterhin wird die Gewichtung des jeweiligen Soft Goals definiert (Summe aller Gewichtungen ergibt 1).
2. Die Alternativen im Zielmodell, die durch Solutions dargestellt sind, werden bezüglich der relevanten Property und Soft Goals und deren Kriterien bewertet. Es erfolgt jeweils die Schätzung eines Werts, der im Intervall des tolerierbaren und des Zielwerts liegt.
3. Um die Evaluation zu beginnen muss ein Startelement und die relevanten Kriterien selektiert werden. Die Menge der Alternativen die unterhalb des gewählten Elements stehen werden bottom-up evaluiert. Es wird dazu wie in Lamsweerde [44] beschrieben verfahren (siehe auch Kapitel 3.2.1.2).
4. Falls eine Alternative negativ bezüglich eines persönlichen Wertes eines relevanten Stakeholders oder eines Elements gekennzeichnet wurde, so wird wie zuvor beschrieben verfahren.
5. Sobald die komplette Ebene abgearbeitet ist wird mit der nächst höheren Ebene analog verfahren. Für jede Ebene werden die Ergebnisse abgelegt.

Die Methode wird anhand des Beispiels in Abbildung 4.6 demonstriert: Um eine „Autorisierung“ beim Zugriff auf die Daten zu ermöglichen ergeben sich drei verschiedene Alternativen. Zum einen wäre es möglich, die Autorisierung zum Zugriff auf Daten über eine schriftliche Genehmigung einzuholen. Eine weitere Möglichkeit stellt die Autorisierung mit der vorhanden Arztkarte (Chipkarte im Klinikum) dar. Schließlich gäbe es noch die Möglichkeit, allen Ärzten Zugriff auf alle Daten zu ermöglichen. Diese Lösung widerspricht allerdings stark dem auf Sicherheit bedachten Datenschutzbeauftragten. Um nun die Alternativen zu evaluieren müssen zunächst die relevanten Property und Soft Goals gewählt, gewichtet und Kriterien erstellt werden. Für die Evaluation werden die Soft Goals „Geringe Kosten“ und „Schneller Zugriff auf Patientendaten“ selektiert. Die Gewichte werden mit 0.4 („Geringe Kosten“) und 0.6 („Schneller Zugriff auf Patientendaten“) vergeben. Die Kriterien, sowie deren Maximalwerte g_{max} und Zielwerte g_T, lauten:

1. Anschaffungskosten, in Euro („Geringe Kosten“), $g_{max} = 1000$, $g_T = 0$
2. Zeit bis Genehmigung für Zugriff auf Daten erteilt ist, in Minuten („Schneller Zugriff auf Patientendaten“), $g_{max} = 15$, $g_T = 0$

Da nun alle notwendigen Daten vorhanden sind kann eine Abschätzung für das jeweilige Kriterium erfolgen. Die Ergebnisse dieser Abschätzung und der Gesamtwert für die verschiedenen Lösungen sind in Tabelle 4.1 gelistet.

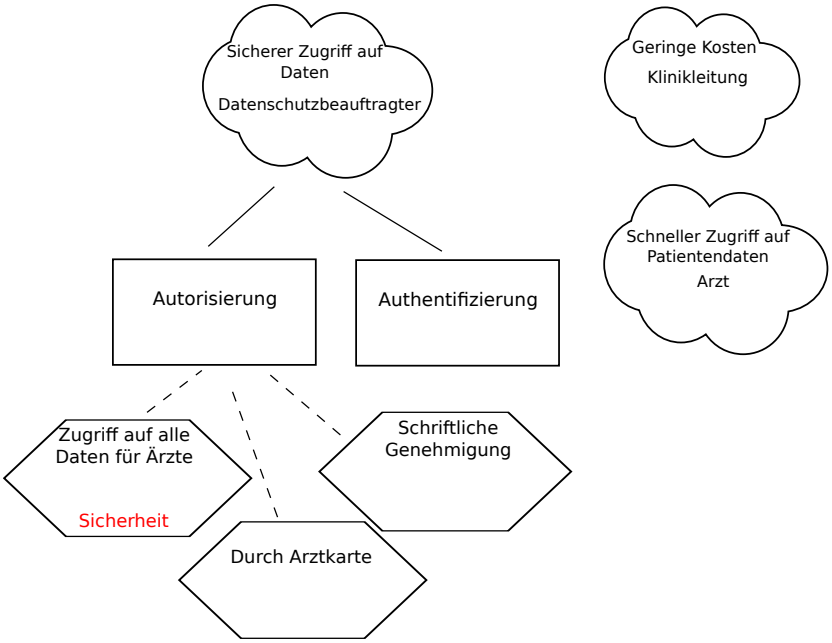


Abbildung 4.6: Beispiel eines Zielmodells zur Evaluation von Alternativen

		Zugriff auf alle Daten	Arztkarte	Schriftliche Genehmigung
Anschaffungskosten, in Euro	0.4	1.0 (0 Euro)	0.0 (1000 Euro)	1.0 (0 Euro)
Zeit bis Genehmigung für Zugriff auf Daten erteilt ist, in Minuten	0.6	1.0 (0 Minuten)	0.93 (1 Minute)	0.0 (15 Minuten)
		1.0	0.558	0.4

Tabelle 4.1: Ergebnisse der Evaluation der Alternativen

Das Ergebnis zeigt, dass die Lösung allen Ärzten Zugriff auf alle Daten zu geben überlegen ist. Es entstehen keinerlei Anschaffungskosten und für die Genehmigung des Zugriff ist keinerlei Zeit notwendig. Die Möglichkeit der Autorisierung per Arztkarte schneidet bei den Anschaffungskosten zwar sehr schlecht ab, gewährleistet aber eine schnelle Autorisierung. Die dritte und letzte Lösung kann nur bezüglich der Kosten überzeugen.

Bei der Überprüfung der besten Lösung fällt die negative qualitative Bewertung zum persönlichen Wert „Sicherheit“ des Datenschutzbeauftragten auf, wodurch eine Kennzeichnung erfolgen muss. Es bleibt abzuwägen ob die Nutzung der Arztkarte zur Autorisierung möglicherweise die bessere Wahl ist. Eine weitere Möglichkeit wäre die Nutzung zusätzlicher Property und Soft Goals sowie Kriterien. Dadurch ergibt sich ein detailliertes Ergebnis, das eine solche Problematik möglicherweise verhindert.

4.4 Erkennen und Auflösen von Konflikten

Bei der Evaluation der Ansätze zum Erkennen und Auflösen von Konflikten in Kapitel 3.2.2 zeigte sich, dass häufig eine Spezialisierung auf eine bestimmte Art von Konflikten stattfindet. Nur der Ansatz von Lamsweerde et al. [16] betrachtete verschiedenste Konflikte, zeichnete sich aber durch eine geringe Praktikabilität aus. Aufgrund dieses Umstandes soll zur Erkennung und Auflösung von Konflikten eine Methode vorgestellt werden, die Teile der betrachteten Ansätze nutzt, aber nicht genau diesen entspricht. Diese Methode umfasst drei verschiedene Konfliktarten:

1. Konflikte durch negative qualitative Auswirkungen auf andere Elemente
2. Rollenbasierte Interessenkonflikte
3. Konflikte durch negative Auswirkungen auf persönliche Werte

Die erste Konfliktart kann durch die Untersuchung der negativen qualitativen Auswirkungen auf andere Elemente erkannt werden. Diese sind bereits explizit im Diagramm sichtbar, es ist aber dennoch sinnvoll auf sie hinzuweisen. Eine solche Auswirkung zeigt zwar, dass der Analyst sich dem Problem bewusst ist, allerdings können solche Konflikte, vor allem in größeren Zielmodellen, nicht mehr auffallen, wenn sehr viele Relationen eingesetzt werden. Die Erkennung dieser Konfliktart läuft folgendermaßen ab:

1. Zunächst werden alle negativen qualitativen Auswirkungen auf andere Elemente im Zielmodell ermittelt.
2. Jeder Auswirkung ist ein Quell- und ein Zielelement zugeordnet. Ein Konflikt ist genau dann vorhanden, wenn sich das Quell- und das Zielelement in einem Teilbaum des Zielmodells befinden und die beiden Elemente nur durch UND-Relationen miteinander verbunden sind.

Abbildung 4.7 zeigt einen solchen Konflikt. Das Soft Goal „Motiviertes Personal“ kann hier durch die Achievement Goals „kostenlose Leistungen“ und „Individuelle Förderung“ erreicht werden. Jedoch wirkt sich das Achievement Goal „kostenlose Leistungen“ negativ auf das Soft Goal „Geringe Kosten“ aus. Dieses gilt es jedoch zu erreichen, da sich das Soft Goal „Hohe Effizienz“ aus beiden Soft Goals gleichermaßen definiert. Bei einer ODER-Relation zu den beiden Soft Goals kann der Konflikt umgangen werden und eine Erkennung würde nicht erfolgen. Ebenso falls es sich bei „kostenlose Lieferungen“ um eine reine Alternative handelt.

Die zweite Konfliktart, der rollenbasierte Interessenskonflikt, beruht auf dem Ansatz von Chung [45]. Dabei werden alle Ziele verschiedener Rollen miteinander verglichen und der Analyst muss feststellen ob es zu Konflikten kommen könnte. Wie bereits dargestellt in Kapitel 3.2.2.3, ist ein solches Vorgehen zwar möglich aber insbesondere bei größeren Zielmodellen kaum praktisch durchführbar. Bei nur 20 zu vergleichenden Zielen müssten bereits 190 manuelle Vergleiche durchgeführt werden. Um diese Menge an

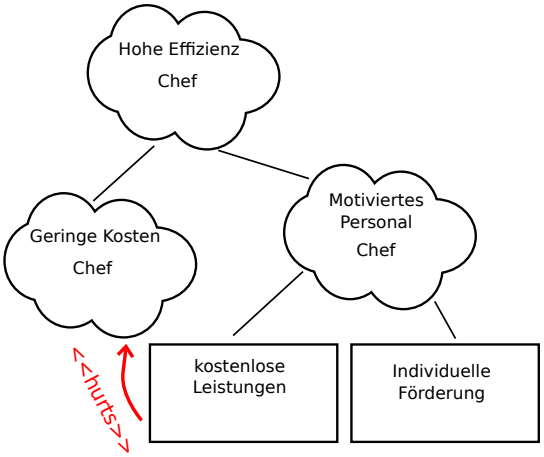


Abbildung 4.7: Ein Konflikt durch negative qualitative Auswirkungen auf ein anderes Element

Vergleichen zu umgehen werden stattdessen die negativen qualitativen Auswirkungen genutzt, um potentielle rollenbasierte Interessenkonflikte zu ermitteln. Dadurch kann es natürlich vorkommen, dass Konflikte nicht erkannt werden, beispielsweise wenn der Analyst ein notwendige negatives Auswirkung nicht erstellt hat. Eine anderweitige Prüfung ist jedoch nicht automatisierbar möglich und auch nicht manuell in akzeptabler Zeit. Die Erkennung von rollenbasierten Interessenkonflikten gestaltet sich wie folgt:

1. Es werden alle negativen qualitativen Auswirkungen auf andere Elemente ermittelt.
2. Falls das Ziel- und Quellelement des Labels zu verschiedenen Stakeholdern, einem Work Stakeholder und einer Work Stakeholder Group gehört und einer der Work Stakeholder ebenso Mitglied der Work Stakeholder Group ist, so handelt es sich um einen Konflikt.

Abbildung 4.8 zeigt ein Beispiel für einen rollenbasierten Interessenkonflikt. Mitglieder der „Klinikleitung“ können auch „Ärzte“ sein, wodurch die negative qualitative Auswirkungen auf das Soft Goal „Geringe Kosten“ zu einem Konflikt führt.

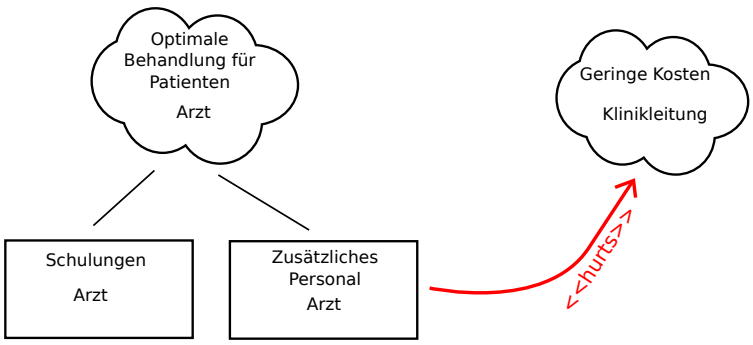


Abbildung 4.8: Ein rollenbasierter Interessenkonflikt

Die dritte Konfliktart nutzt die zuvor vorgestellte Möglichkeit zur Kennzeichnung von negativen Auswirkungen von Elementen im Zielmodell hinsichtlich persönlicher Werte. Die Art der Erkennung ist an die erste Konfliktart angelehnt und ist folgendermaßen definiert:

1. Es werden alle Elemente ermittelt, die sich nach der Kennzeichnung negativ auf persönliche Werte von Stakeholdern auswirken.
2. Alle Stakeholder, denen der entsprechende persönliche Wert zugeordnet wurde, werden ermittelt.
3. Ähnlich zur ersten Konfliktart wird überprüft, ob die Elemente, denen die ermittelten Stakeholdern zugeordnet sind und die negativ gekennzeichneten Elemente nur durch UND-Relationen miteinander verbunden sind.

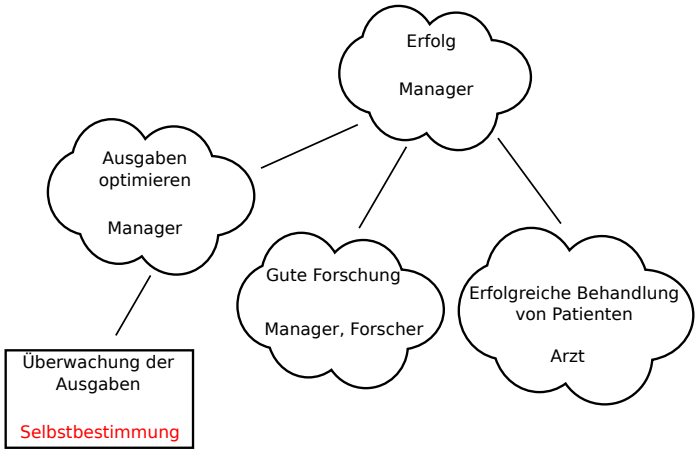


Abbildung 4.9: Ein Konflikt durch negative Auswirkungen auf persönliche Werte

Abbildung 4.9 zeigt ein Beispiel für diese Konfliktart. Den Stakeholdern „Arzt“ und „Forscher“ wurde der persönliche Wert „Selbstbestimmung“ zugeordnet. Das Achievement Goal „Überwachung der Ausgaben“ wirkt sich jedoch negativ aus diesen Wert aus. Dadurch, dass alle Elemente durch UND-Relationen miteinander verbunden sind werden zwei Konflikte erkannt. Die Interessen des Forschers wie auch des Arztes könnten durch das Achievement Goal negativ beeinflusst werden.

Bezüglich der Auflösung von Konflikten wurde bereits in Kapitel 3.2.2 gezeigt, dass die betrachteten Ansätze meist nur die Verfeinerung des Zielmodells bis zu einer konfliktfreien Variante oder die Aufnahme von Verhandlungen vorschlagen. im Rahmen von Verhandlungen, bietet sich nach [52] die Möglichkeit, dass der Analyst als Mediator zwischen den verschiedenen Stakeholdern fungiert. Das vorgestellte Verfahren zur Mediation besteht aus drei Phasen: der Identifikation, der Definition und der Auflösung des Konflikts. Die bereits vorgestellten Methoden können insbesondere bei den ersten beiden Phasen hilfreich sein, da die Kennzeichnung der Konflikte im Zielmodell auch die Beziehungen zu anderen Elementen offenlegt. Die Phase zur Konfliktidentifikation umfasst

auch die genaue Modellierung der Stakeholder, wozu sich die persönlichen Werte eignen. Sie geben Auskunft über die höheren Ziele Beteiligter und sind somit vor allem für die Modellierung von Charakterzügen geeignet.

5 Umsetzung der Methode in UNICASE

Nachdem die Methode vollständig im vorherigen Kapitel definiert wurde, erfolgt nun Beschreibung der Umsetzung in Form eines Editors zur Zielmodellierung in UNICASE. Der erste Abschnitt gibt eine kurze Übersicht zu den eingesetzten Frameworks und wie sie mit UNICASE in Verbindung stehen. Anschließend werden die Anforderungen für die Umsetzung genauer betrachtet. Der dritte Abschnitt befasst sich mit dem Metamodell, welches im Rahmen der Umsetzung notwendig war, um die neuen zielorientierten Elemente mit den bereits bestehenden aufgaben- und prozessorientierten Elementen von UNICASE zu verknüpfen. Anschließend erfolgt die Vorstellung der erstellten Komponenten, sowie weiteren Strukturen. Den Abschluss dieses Kapitels bildet eine Beschreibung der Erfahrungen im Rahmen der Umsetzung.

5.1 Frameworks und UNICASE

UNICASE ist in Form eines Eclipse-Plugins implementiert. Es setzt somit auf die Eclipse-Plattform auf und nutzt den Plugin-Mechanismus von Eclipse. Alle Artefakte in UNICASE basieren auf einem einheitlichen Modell, das wiederum das Eclipse Modeling Framework (EMF) nutzt. Die Artefakte nennt man daher auch Modellelemente, durch deren einheitliches Modell UNICASE auch generische Editoren anbietet.

Das EMF-Framework setzt den Gedanken der modellgetriebenen Softwareentwicklung um und ermöglicht die Definition von Modellen, mit denen die Generierung von Java-Klassenstrukturen möglich ist. Basierend auf den in EMF definierten Modellen können neben einfachen Klassenstrukturen auch grafische Editoren generiert werden. Dieser Aufgabe widmet sich das Graphical Modeling Project (GMP) [53]. Es umfasst das Graphical Modeling Framework (GMF) welches aus einer Tooling- und Runtime- Komponente besteht. Mit Hilfe der Tooling-Komponente können die grafischen Elemente, sowie Werkzeuge des Editors definiert werden. Ein Mapping ermöglicht schließlich die Verknüpfung mit dem zu Grunde liegenden EMF-Modell. Basierend auf diesem Mapping kann ein Generator-Modell erstellt werden, welches zur Generierung des Codes für den grafischen Editor dient. Die Runtime-Komponente ermöglicht schließlich die Ausführung des generierten Editors als Eclipse-Plugin. Somit können Zusammenhänge im EMF-Modell grafisch in einem Editor modelliert werden.

Im Rahmen der Implementierung des Editors zur Zielmodellierung erfolgt die Definition eines EMF- und GMF-Modells. Das damit erzeugte Plugin wurde dann manuell um

weitere Funktionalität erweitert. Eine genaue Erklärung der erzeugten Plugins und der weiteren Klassenstruktur erfolgt in Abschnitt 5.5.

5.2 Anforderungen an die Implementierung

Die Anforderungen an die Implementierung basieren hauptsächlich auf den Anforderungen an die Methode aus Kapitel 3.2. Es folgen zunächst die funktionalen Anforderungen, anschließend die nicht-funktionalen Anforderungen.

5.2.1 Funktionale Anforderungen

Das Nutzungsdiagramm in Abbildung 5.1 zeigt die funktionalen Anforderungen an die Implementierung. Es ergeben sich zwei wesentliche Aufgaben: das Erstellen und Bearbeiten des Zielmodells und die Analyse von Zielmodellen. Die erstgenannte Aufgabe bildet den zentralen Use Case. Die Erweiterung zur Analyse des Zielmodells hinsichtlich Konflikten und Alternativen ist durch zwei weitere Use Cases gegeben. Das Nutzungsdiagramm zeigt zudem die gefundenen Systemfunktionen. Der Großteil davon ergab sich aus den Anforderungen an die Methode. Jedoch gibt es eine weitere Funktion, die speziell für die Umsetzung eines Editors zur Zielmodellierung interessant ist. Es handelt sich um die Visualisierung von Teilausschnitten des Modells und dessen Abhängigkeiten und um das Zurücksetzen der Ansicht. In Kapitel 5.3 wird diese Funktionalität ausführlich besprochen und ein entsprechendes Konzept vorgestellt.

Auf eine detaillierte Vorstellung der Use Cases wird im Rahmen dieses Kapitels verzichtet, da bereits in Kapitel 4 eine ausführliche Beschreibung des Vorgehens erfolgte.

5.2.2 Nicht-Funktionale Anforderungen

Die nicht-funktionalen Anforderungen für die Implementierung orientieren sich größtenteils an der ISO/IEC 9126-Spezifikation [54], die eine Reihe von Qualitätsmerkmalen für Software definiert. Diese Qualitätsmerkmale sind in Haupt- und Untermerkmale eingeordnet, die bei der jeweiligen Anforderung genannt werden. Die nicht-funktionalen Anforderungen:

- **Klare Unterscheidbarkeit der Elemente (Benutzbarkeit - Verständlichkeit)**
  Die Elemente im Zieleditor sind klar voneinander zu unterscheiden. Kein Elementname, sowie Symbol darf doppelt vorhanden sein.

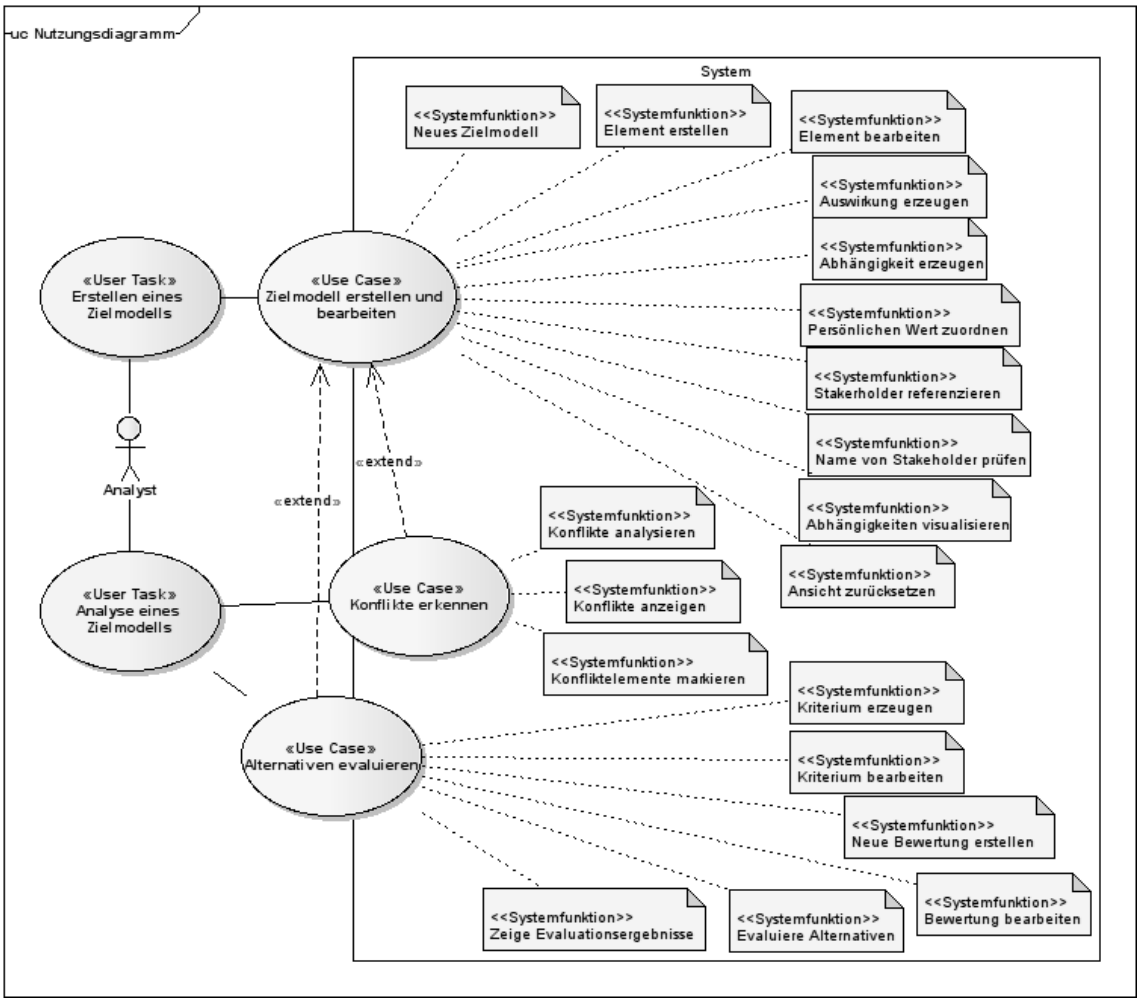


Abbildung 5.1: Nutzungsdiagramm

- **Aufteilung der Funktionalität auf Komponenten (Änderbarkeit - Modifizierbarkeit)**
  Die Funktionalität ist auf verschiedene Komponenten verteilt. Diese zeichnen sich durch eine hohe innere Kohäsion und eine geringe Kopplung aus.
- **Genauigkeit bei Evaluationsergebnissen (Funktionalität - Richtigkeit)**
  Bei der Evaluation der Alternativen erfolgt die Berechnung des Wertes jeder Alternative auf 2 Stellen genau.
- **Eindeutigkeit von Stakeholdern (Funktionalität - Richtigkeit)**
  Es muss sichergestellt sein, dass ein Stakeholder nur durch ein Modellelement repräsentiert wird.
- **Verwendung von UNICASE**
  Der Editor zur Zielmodellierung muss in Form eines UNICASE-Plugins vorliegen

(Kompatibel zu Version 0.4.5)

5.3 Konzept zur Visualisierung von Teilausschnitten im Zielmodell

Die Visualisierung von Abhängigkeiten im Zielmodell ist vor allem bei großen Zielmodellen eine interessante Funktion. Das Ziel ist es, lediglich eine Auswahl von Elementen deren Abhängigkeiten anzuzeigen. Zunächst muss untersucht werden, wie genau sich Abhängigkeiten in einem Zielmodell darstellen. Innerhalb eines Zielmodells können Ziele durch weitere Ziele verfeinert werden. Dazu existiert zwischen den feingranularen Zielen und dem übergeordneten Ziel eine UND- oder eine ODER-Relation. Die Erfüllung des übergeordneten Ziels ist somit abhängig von den untergeordneten Zielen. Erfolgt die Verfeinerung über mehrere Ebenen, so handelt es sich nicht mehr um direkte Abhängigkeiten. Ein weitere Möglichkeit der Abhängigkeit stellt eine positive bzw. negative Auswirkung auf ein anderes Element im Zielmodell dar.

Nun muss zwischen der Menge an gewollten Informationen und den angezeigten Elementen eine Abwägung stattfinden. Eine Methode, die für das gewählte Element auch Abhängigkeiten auf der dritten und vierten Ebene anzeigt bietet zwar sehr viele Informationen, jedoch wird kaum eine Einschränkung der Ansicht vorgenommen und wodurch möglicherweise viele unnötige Elemente angezeigt werden. Abbildung 5.2 verdeutlicht vier

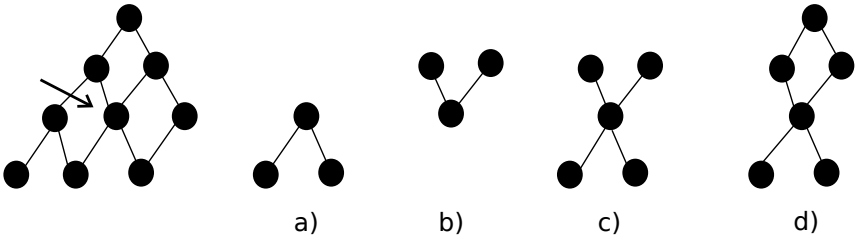


Abbildung 5.2: Verschiedene Möglichkeiten zur Visualisierung von Abhängigkeiten

Möglichkeiten zur Visualisierung von Abhängigkeiten des markierten Knotens. Möglichkeit a) zeigt die direkten Abhängigkeiten an. Auch wenn man so nur wenig Informationen erhält, besteht der Vorteil in der sehr überschaubaren Menge von Knoten. Entsprechend zeigt Variante b) nur die Knoten an, die vom gewählten Knoten abhängig sind. Variante c) kombiniert die beiden zuvor genannten Möglichkeiten. Die letzte Variante bietet zusätzlich die Möglichkeit über mehrere Ebenen zu operieren.

Die letzte Möglichkeit kann zwar Abhängigkeiten in beide Richtungen und auf mehreren Ebenen anzeigen, jedoch entspricht dies möglicherweise zu viel Information. Die erste Variante erfüllt die Anforderung zur Anzeige der Abhängigkeiten vollständig. Jedoch ist nicht ersichtlich, welche anderen Elemente von diesem Element abhängig sind.

Allerdings könnte dies eine interessante Information für eine weitere Visualisierung sein. Auf diese Art und Weise wäre beispielsweise die Bewegung im Zielmodell in beide Richtungen denkbar. Variante c) erscheint somit am besten. Es werden Abhängigkeiten des Elements in beide Richtungen angezeigt und in Kombination mit der Möglichkeit, eine Menge an Elementen zur Visualisierung zu nutzen, können auf Wunsch auch mehrere Ebenen dargestellt werden.

Die Schritte zur Visualisierung von Teilausschnitten und deren Abhängigkeiten im Zielmodell gestalten sich somit folgendermaßen:

1. Der Aktor selektiert eine Menge von Elementen im Zielmodell und wählt die Option zur Visualisierung.
2. Das System stellt die Elemente dar, die eine direkte Abhängigkeit zu den gewählten Elementen besitzen. Die Richtung der Abhängigkeit spielt dabei keine Rolle. Zudem stellt das System alle direkten Abhängigkeiten zu positiven bzw. negativen Auswirkungen dar.
3. Der Aktor wählt die Option zum Zurücksetzen der Ansicht.
4. Das System zeigt wieder alle Elemente des Zielmodells an.

Schritt 1 kann der Aktor wiederum auf eine Teilmenge der bisher dargestellten Elementmenge anwenden.

5.4 Metamodell zur Integration von Zielen in UNICASE

In Kapitel 4.1 wurde bereits in Form von Abbildung 4.1 das Metamodell der Methode vorgestellt. Zur Integration in UNICASE, welches bisher eine Reihe von aufgaben- und prozessorientierten Elementen umfasst, fehlt die Verknüpfung zu diesen Elementen. Da Elemente des zielorientierten Requirements Engineering typischerweise in einer frühen Phase erfasst werden, ist insbesondere deren Verknüpfung zu den aufgaben- und prozessorientierten Elementen, interessant. Die dadurch entstehende Infolgedessen sind die Elemente während des gesamten Entwicklungsprozess verfolgbar, was maßgeblich zur Sicherung der Qualität der Spezifikation beiträgt. In Zusammenarbeit mit den Mitarbeitern der Arbeitsgruppe Software Engineering von Prof. Dr. Barbara Paech in Heidelberg entstand im Rahmen der Einführung von zielorientierten Elementen in UNICASE ein Metamodell, mit dem die Integration von aufgaben-, prozess- und zielorientierten Elementen möglich ist. Als Basis für die integrierten UNICASE Elemente wurde die UNICASE Version 0.4.5 verwendet. Das Metamodell besteht aus einer Vielzahl von Elementen, deren Beschreibung und Beziehungen ausführlich im Anhang A beschrieben sind. Im Rahmen dieses Abschnitts erfolgt lediglich eine kurze Erklärung.

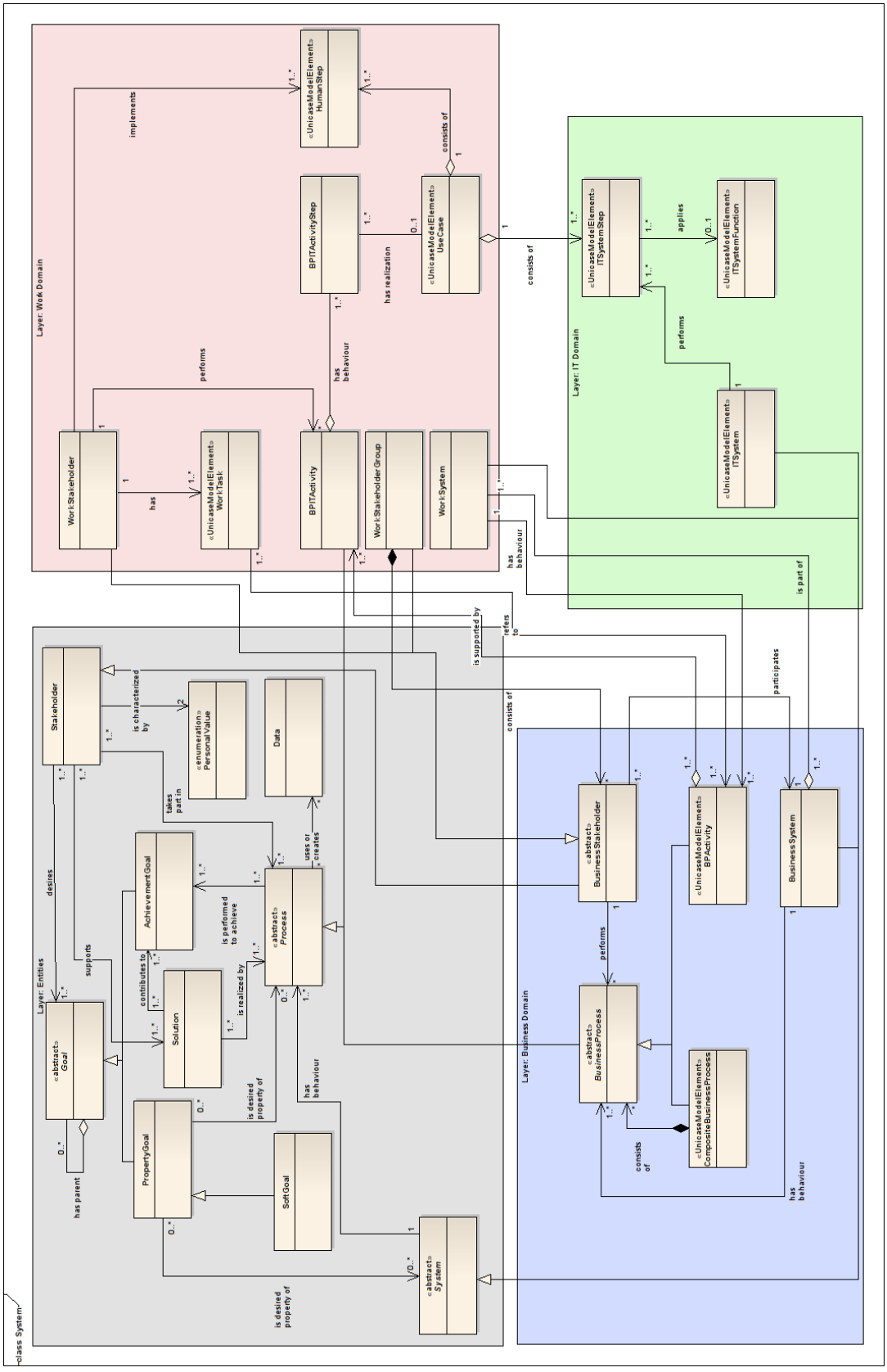


Abbildung 5.3: Metamodell zur Integration von ziel-, prozess- und aufgabenorientierten Elementen

Das Metamodell in Abbildung 5.3 unterteilt sich in vier verschiedene Schichten. Die Erste umfasst allgemeine Entitäten, die anderen Schichten spiegeln verschiedene Sichtweisen wider:

1. Entities
2. Business Domain
3. Work Domain
4. IT Domain

Die erste Schicht umfasst Entitäten, welche die Basis für die speziellen Versionen der anderen Schichten darstellen und beinhaltet die bereits vorgestellten Goal- und die Stakeholder-Entität. Das Process-Element ist abstrakt und bildet die Grundlage für jeden konkret ausgeprägten Prozess. Stakeholder können einem solchen Prozess zugeordnet werden, ebenso Goal- und Solution-Elemente. Somit können ziel- und prozessorientierten Elemente miteinander verbunden werden. Weiterhin ist jeder Prozess Teil eines Systems und beschreibt dessen Verhalten.

Die zweite Schicht betrachtet die bereits vorgestellten Elemente bezüglich der Geschäftssicht. So gibt es einen Business Stakeholder, der abstrakt ist und bei dem es sich um eine Rolle oder organisatorische Einheit im zu Grunde liegenden Geschäftssystem, in Form des Business System, handelt. Auf dieser Schicht können Geschäftsprozesse und deren Aktivitäten für ein Geschäftssystem definiert werden.

Die Sicht auf die zu erbringenden Tätigkeiten wird in der dritten Schicht thematisiert. Beim bereits vorgestellten Work Stakeholder handelt es sich um eine konkrete Ausprägung des Business Stakeholder. Das Work System bildet seinen Zuständigkeitsbereich ab. Ein Work System stellt demnach einen Teil eines Business System dar. Die Verknüpfung zwischen den aufgaben- und prozessorientierten Elementen wird durch die Elemente BPITActivity und BPITActivityStep geleistet. Diese definieren, wie die Aktivität eines Geschäftsprozesses durch Work Stakeholder unterstützt wird. Entsprechend kann ein BPITActivityStep durch einen Use Case beschrieben sein. Dieser wiederum besteht aus Schritten die durch Menschen oder IT Systeme ausgeführt werden können.

Die Sicht auf die IT Systeme bildet die vierte Schicht. Ein IT System führt die genannten Schritte, die Teil eines Use Case sein können, aus. Ein solcher Schritt kann die ausführung einer IT System Function umfassen.

5.5 Struktur

Die zukünftigen Elemente der Methode, sowie deren Verknüpfungen zu anderen UNICASE Elementen wurden bereits in Kapitel 4.1 und im vorangegangenen Abschnitt vorgestellt. Das EMF-Modell ist eine Kombination dieser beiden Metamodelle. Anhand dieses

Modells erfolgt die Erstellung von drei Komponenten, die im nächsten Abschnitt vorgestellt werden. Anschließend werden Erweiterungen betrachtet, die notwendig waren um alle Funktionen umzusetzen.

5.5.1 Komponenten

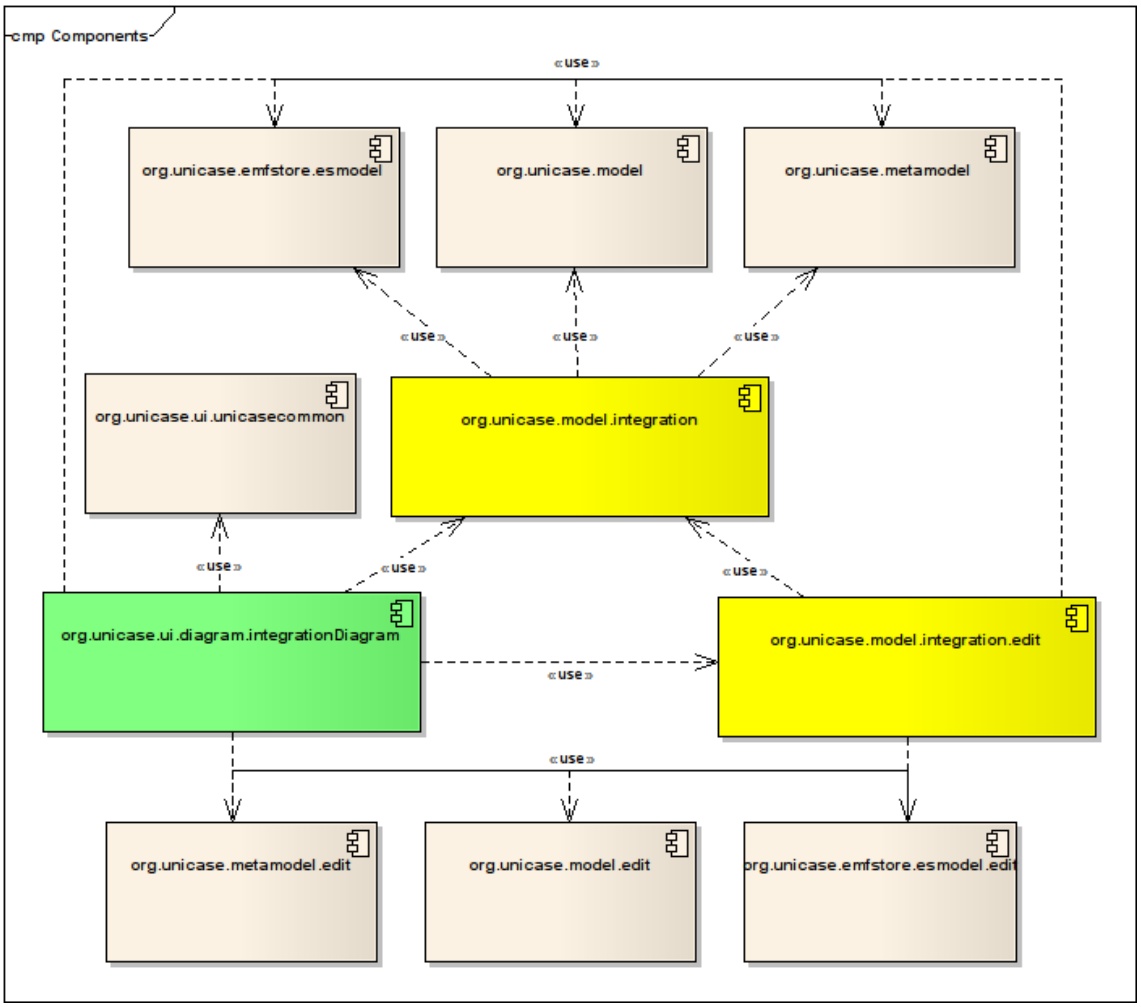


Abbildung 5.4: Die Komponenten und deren Beziehungen

Abbildung 5.4 zeigt die Komponenten der Umsetzung und deren Beziehungen. Das Komponentendiagramm umfasst nicht alle Abhängigkeiten zu anderen Komponenten, da das Diagramm sonst zu unübersichtlich wäre. So fehlen beispielsweise die Abhängigkeiten zu den GMF-Komponenten. Stattdessen sind die Abhängigkeiten zu bestehenden UNICASE-Plugins und untereinander abgebildet. Die farblich hervorgehobenen Komponenten sind im Rahmen der Umsetzung erstellt worden. Die Klassen der gelb gekennzeichneten Komponenten „org.unicase.model.integration“ und „org.unicase.model.integration.edit“

wurden durch das EMF-Modell generiert. Die Komponente „org.unicase.model.integration“ beinhaltet das EMF- Modell, sowie das zuvor erwähnte GMF-Modell und Mapping. Die Abhängigkeiten zu den UNICASE-Plugins sind durch die Verknüpfung von EMF- Modellelementen mit bestehenden UNICASE-Modellelementen zu erklären. Die zugehörige Komponente „org.unicase.model.integration.edit“ beinhaltet die Funktionen zum Erzeugen, Bearbeiten und Löschen der Modellelemente von „org.unicase.model.integration“.

Die Komponente „org.unicase.ui.diagram.integrationDiagram“ stellt den Editor zur Zielmodellierung dar. Die Komponente ist grün markiert, da sie zwar aus dem GMF-Modell generiert wird, jedoch enthält sie auch zusätzliche Funktionalität, die nach der Generierung manuell hinzugefügt wurde. Dazu zählt die Analyse des Zielmodells hinsichtlich Alternativen und Konflikte und die Definition spezieller Werkzeuge. Abhängigkeiten bestehen hier zusätzlich zur Komponente „org.unicase.ui.unicasecommon“, welche verschiedene Basisfunktionen zur Nutzung von Diagrammen innerhalb des UNICASE-Plugins beinhaltet.

Die notwendigen Erweiterungen des Editors sind im nächsten Abschnitt dargestellt.

5.5.2 Erweiterung des Editors

Die Erweiterung des generierten Editors fand auf mehreren Stufen statt und ist in Abbildung 5.5 in Form eines Klassendiagramms dargestellt. Zunächst musste die Funktionalität zur Analyse des Zielmodells auf Konflikte und Alternativen hinzugefügt werden. Dazu wurde das Interface „IAnalyser“, sowie dessen Realisierung „Analyser“ erstellt. Die Klassen „EvaluationResults“, „SingleEvaluationResult“ und „ConflictResult“ bilden jeweils die Analyseergebnisse ab. Die Klasse „EvaluationResults“ wird pro evaluierten Knoten abgelegt. Ein Knoten stellt dabei ein Element im Zielmodell dar, das durch mindestens ein Solution-Element verfeinert wird. Für jedes Solution-Element ergibt sich dann eine Instanz der Klasse „SingleEvaluationResult“, die den erzielten Wert umfasst. Die Klasse zur Abbildung der Konflikte ist „ConflictResult“. Der gefundene Konflikte wird zum einen im Attribut „message“ in Form eines Textes festgehalten und zum anderen werden, zur späteren Hervorhebung der Konflikte, die entsprechenden Elemente referenziert.

Neben der reinen Logik zur Analyse existieren auch Views zur Darstellung der Ergebnisse beider Analysearten: „ResultsView“ und „ConflictsView“. Das dritte View „StakeholdersView“ hat keinen direkten Bezug zur Analyse, zeigt jedoch die im Modell vorhandenen Stakeholder und deren persönliche Werte, da diese sonst nicht direkt einsehbar sind.

Die Klasse „GoalGraphUtil“ ist ein Singleton mit vielen allgemeine Methoden für die Arbeit mit dem Zielmodell. So können Knoten, Elemente die durch andere Elemente verfeinert sind, extrahiert oder Konflikte hervorgehoben werden. Die Darstellung dieser Klasse ist im Diagramm zu Gunsten der Übersicht vereinfacht.

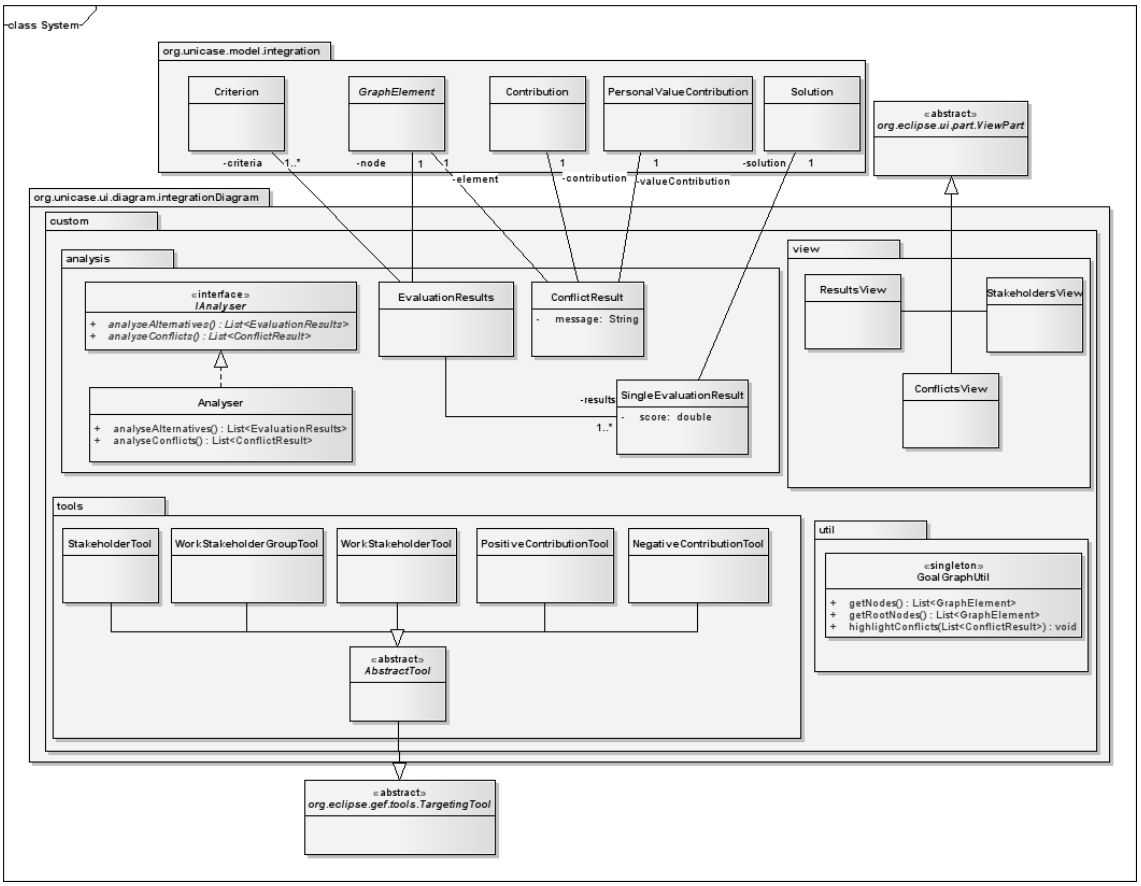


Abbildung 5.5: Die Erweiterung des Editor-Plugins

Einen weiteren großen Teil der Erweiterung stellt das Package „tools“ da. GMF bietet leider keine Möglichkeit Relationen abzubilden, die auf bidirektionalen Beziehungen im Metamodell basieren. Ein Beispiel dafür sind die Elemente Stakeholder und Goal. Ein Stakeholder kann mehreren Goal-Elementen zugeordnet werden; ebenso können Goal-Elementen mehrere Stakeholder zugeordnet werden. GMF kann einen solchen Sachverhalt nur durch eine vermittelnde Klasse darstellen, welche dann in grafischer Form auf einen Link zwischen den beiden Elementen abgebildet wird. Im Editor sollen Stakeholder jedoch direkt zugeordnet werden und nicht als eigene Elemente im Zielmodell auftauchen. Um dies zu erreichen mussten für die direkt zuzuordnenden Elemente wie Stakeholder, Work Stakeholder sowie Work Stakeholder Group und die positiven bzw. negativen Auswirkungen auf persönliche Werte eigene grafische Werkzeuge erstellt werden. Die Darstellung im Diagramm ist stark vereinfacht - neben den gezeigten „Tool“-Klassen sind viele weitere Klassen notwendig.

5.6 Erfahrungen im Rahmen der Umsetzung

Die Umsetzung des Metamodells mit EMF war problemlos möglich. Auch das GMF-Modell sowie das entstandene Mapping war verhältnismäßig leicht zu erstellen. Jedoch ergaben sich einige Probleme bei der Erstellung und Anpassung des Editors. GMF ist ein vielseitiges Werkzeug, jedoch ermöglicht es für den hier gegebenen Anwendungsfall nur unzureichende Unterstützung, vor allem durch mangelnde Anpassbarkeit. Da die Zuordnung von Stakeholdern und persönlichen Werten zu Elementen nicht direkt realisiert werden konnte, musste mit viel Aufwand verbunden, die manuelle Erstellung von passenden Werkzeugen erfolgen. Dies wurde aufgrund mangelnder Dokumentation und schlechter Lesbarkeit des Quellcodes zusätzlich erschwert.

Hinzu kommt, dass das Zielmodell durch einen Bug in UNICASE nachdem es mehrfach (4-5 Mal) geöffnet und geschlossen wurde nicht mehr angezeigt werden kann. Die einzige Möglichkeit zur Behebung des Problems besteht darin, das bestehenden Projekt zu löschen und eine Sicherheitskopie wiederherzustellen. Dieser Fehler konnte auch mit anderen bestehenden Editoren in UNICASE nachgewiesen werden.

Insgesamt betrachtet wirkt EMF wesentlich ausgereifter als GMF und bietet mehr Raum für manuelle Erweiterungen. Ein einfacher Editor ist mit GMF sehr leicht zu erstellen, manuelle Anpassungen sind jedoch sehr aufwendig und teilweise auch komplex. Der Vorteil der Generierung des Quellcodes ist somit leider nur in Teilen vorhanden.

6 Fallstudie

Nach Vorstellung der Methode und der Umsetzung des Plugins für UNICASE folgt nun die praktische Evaluation anhand einer Fallstudie. Dazu dient ein Szenario, gestellt durch den Lehrstuhl für Software Engineering der Universität Heidelberg, zum Aufbau eines Informationsportals des Nierenzentrums (NZ) des Universitätsklinikums Heidelberg. Der erste Abschnitt umfasst die Erläuterung des erstellten Zielmodells. Anschließend folgt die Vorstellung der Verfahren zur Evaluation hinsichtlich Alternativen und Konflikte mit den zuvor erarbeiteten Methoden. Den Abschluss dieses Kapitels bilden die Erfahrungen mit der Ausführung der Fallstudie.

6.1 Das Zielmodell

Die Fallstudie wird nur in diesem Abschnitt nur in Teilen vorgestellt, die vollständige Beschreibung ist in Anhang B zu finden.

Die Erstellung des Zielmodells folgt dem in Kapitel 4.2 beschriebenen Vorgehen, da dieses sich in anderen Ansätzen zur Zielmodellierung gleich darstellt, soll hier nicht weiter darauf eingegangen werden. Stattdessen folgte eine Beschreibung der vorhandenen Elemente des Zielmodells aus Abbildung 6.1, das mit Hilfe des Editors zur Zielmodellierung erstellt wurde. Die Bearbeitung des Szenarios fand unter zusätzlichen Freiheiten statt. So durften zusätzliche technische Alternativen, sowie verfeinerte Ziele modelliert werden. Jedoch war durch die Aufgabenstellung das gezielte Nachfragen zur Vervollständigung des Modells nicht möglich.

Zu den übergeordneten Zielen im Modell gehören die Soft Goals „Wirtschaftlicher Erfolg“ und „Guter Ruf“. Der wirtschaftliche Erfolg kann durch den effizienten Umgang mit vorhandenen Mitteln erreicht werden. Des Weiteren ist eine möglichst geringe Aufenthaltszeit der Patienten und eine gleichmäßige Bettenauslastung erwünscht. Schließlich muss die Patientenanzahl erhöht werden, um weitere Gelder zu erlangen.

Der gute Ruf der Klinik ist ein zentraler Aspekt um diese Ziele zu erreichen. Da Fachärzte in der Regel die Wahl haben zu welcher Klinik sie ihre Patienten überweisen, steht die Überzeugung der Fachärzte für die Klinikleitung und den Manager im Vordergrund. Auch die Meinung der Patienten ist wichtig, da auch sie potentiell Rückmeldungen an den Facharzt geben. Die Zufriedenheit der Patienten hängt hauptsächlich von dem Erfolg der Behandlung ab. Jedoch haben die Patienten auch das Ziel, möglichst geringe Restriktionen durch eine Behandlung zu erfahren. Dieser Aspekt ist für den Arzt des Nierenzentrums essentiell um die erfolgreiche Behandlung sicherzustellen. Die beiden Ziele stehen somit im Konflikt zueinander, wodurch die Kennzeichnung durch eine negative

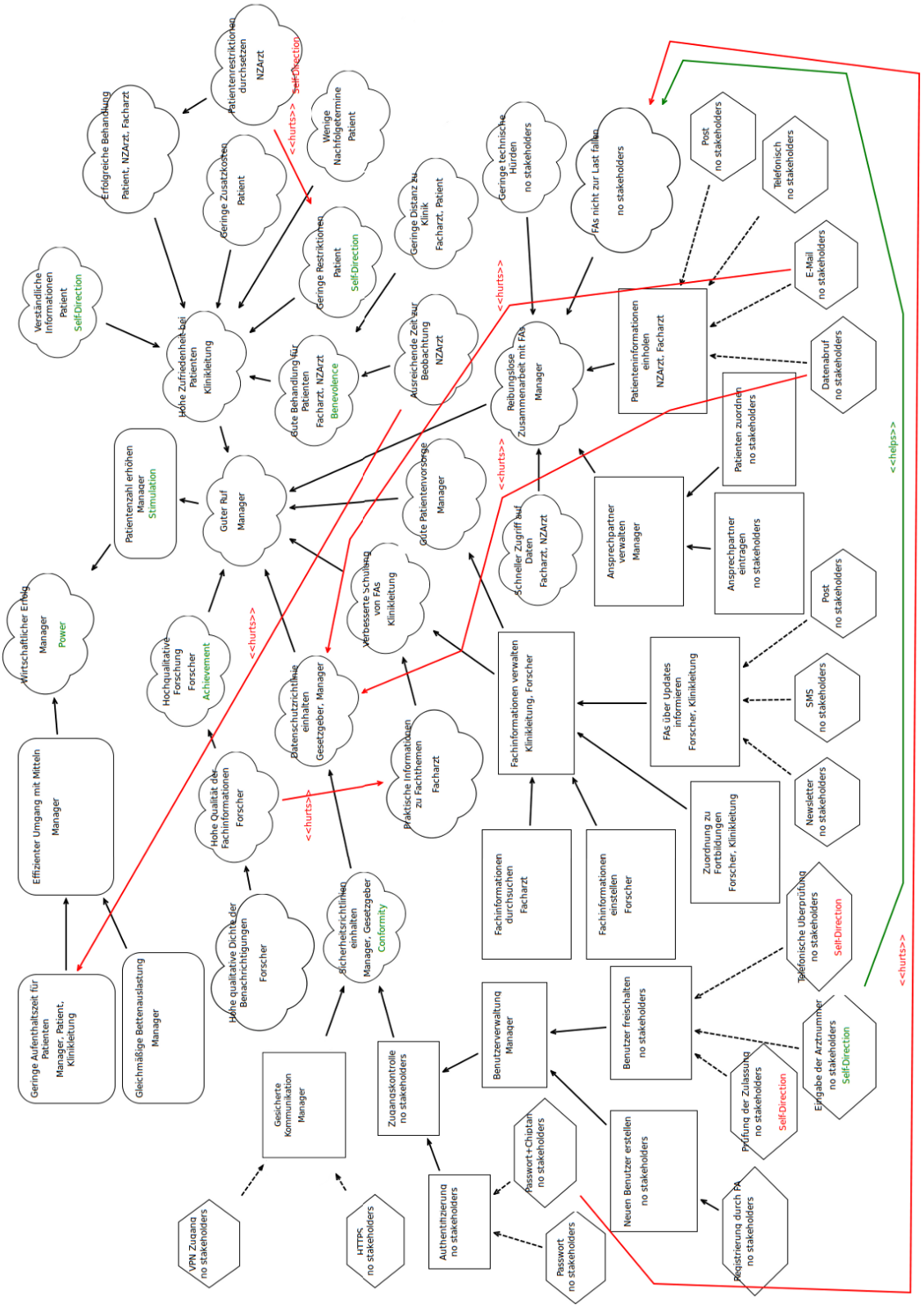


Abbildung 6.1: Das Zielmodell zur Fallstudie

Auswirkung vom Soft Goal „Patientenrestriktionen durchsetzen“ zum Soft Goal „Geringe Restriktionen“ notwendig ist. Zu einer erfolgreichen Behandlung gehören für Facharzt und Patient eine geringe Distanz zur Klinik. Der Arzt des Nierenzentrums möchte vor allem ausreichend Zeit zur Beobachtung der Patienten. Dieses Soft Goal wirkt sich negativ auf das Property Goal „Geringe Aufenthaltszeit für Patienten“ aus, welches zuvor unter Betrachtung der wirtschaftlichen Aspekte genannt wurde.

Kritisch für den guten Ruf der Klinik ist auch die Zusammenarbeit mit den Fachärzten - daher das Soft Goal „Reibungslose Zusammenarbeit mit Fachärzten“. Um dies zu erreichen soll den Fachärzten nicht zur Last gefallen werden, beispielsweise durch vermeidbare Überprüfungen oder Anfragen. Ebenso muss das Informationsportal sich durch geringe technische Hürden auszeichnen und einen schnellen Zugriff auf die Patientendaten ermöglichen. Die Achievement Goals „Ansprechpartner verwalten“ und „Patienteninformationen einholen“ bilden erste Zustände des zu entwickelnden Systems ab. Für die Einholung der Patientendaten stehen eine Reihe von Alternativen bereit. Der Abruf der Daten könnte direkt auf der Plattform erfolgen, beispielsweise in dem Informationen auf die Plattform übertragen werden. Die Einholung wäre aber auch per E-Mail möglich, telefonisch oder per Post. Die ersten beiden Lösungen sind jedoch nicht mit der Einhaltung der Datenschutzrichtlinie verträglich, daher die negative Kennzeichnung im Modell.

Das Soft Goal „Verbesserte Schulung von Fachärzten“ trägt ebenfalls zum guten Ruf der Klinik bei. Es kann durch Erfüllung des Soft Goals „Praktische Informationen zu Fachthemen“ und durch die verschiedenen Achievement Goals bezüglich der Verwaltung von Fachinformationen erreicht werden. Alternativen ergeben sich hier für das Achievement Goal „Fachärzte über Updates informieren“. Das Ziel der Forscher, in der Klinik eine möglichst hohe Qualität der Fachinformationen zu erreichen, wirkt sich negativ auf das Ziel der Fachärzte aus, die möglichst praktische Themen verlangen.

Der linke Teil des Zielmodells geht insbesondere darauf ein wie die Sicherheitsrichtlinien im Rahmen des Informationsportals eingehalten werden können. In der Fallstudie wird dazu die gesicherte Kommunikation, die in Form eines Achievement Goals dargestellt ist, genannt und durch die Alternativen „VPN Zugang“ und „HTTPS“ erreicht werden kann. Weitere Alternativen wurden für die Achievement Goals „Authentifizierung“ und „Benutzer freischalten“ identifiziert.

Im Anschluss an die Erstellung der Zielelemente fand die Bewertung hinsichtlich persönlicher Werte statt. Dazu wurden hauptsächlich Soft Goals verwendet, die höherwertige Ziele darstellen, also durch eine Reihe weiterer Ziele verfeinert wurden. Eine Ausnahme stellt die Bewertung der Alternativen für das Achievement Goal „Benutzer freischalten“ dar. Hier wurden die Alternativen „Prüfung der Zulassung“ und „Telefonische Überprüfung“ negativ bezüglich Selbstbestimmung bewertet, da eine gewisse Abhängigkeit entsteht und nicht sofort auf das Portal zugegriffen werden kann. Stattdessen muss bis zur Freischaltung durch eine verantwortliche Person gewartet werden. Die Alternative „Eingabe der Arztnummer“ erhielt eine positive Bewertung für den genannten Wert, da bei

Abbildung 6.2: Visualisierung eines Teilausschnitts

erfolgreicher Eingabe eine sofortige Freischaltung erfolgt und keine Abhängigkeit zu anderen Personen entsteht.

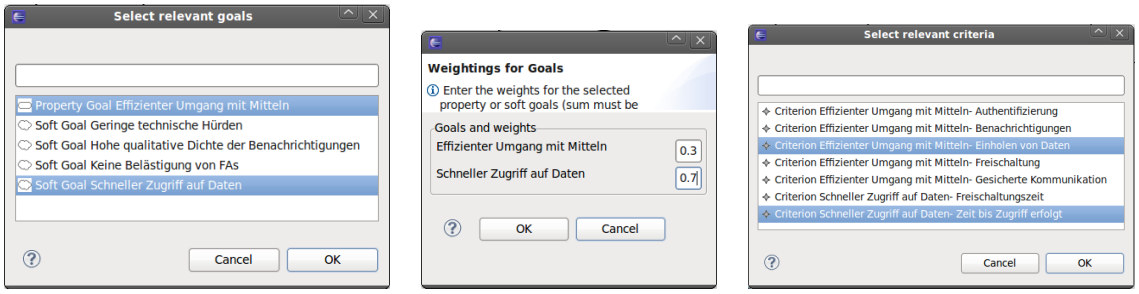
Das Zielmodell in Abbildung 6.1 besteht zwar bereits aus einer ansehnlichen Menge von Elementen, in der Praxis sind jedoch wesentlich größere Modelle denkbar. Dennoch ist bereits in diesem Zustand eine Übersicht schwer zu erlangen. Dabei kann die Funktionalität zur Visualisierung von Teilausschnitten behilflich sein. Abbildung 6.2 stellt die Visualisierung der Abhängigkeiten der Soft Goals „Guter Ruf“ und „Hochqualitative Forschung“ dar. Das Ergebnis ist wesentlich leichter erfassbar, wodurch die Problematik der Übersicht abgemildert wird.

6.2 Evaluation von Alternativen

Im vorgestellten Zielmodell sind insgesamt fünf Möglichkeiten zur Evaluation von Alternativen gegeben. Damit eine Evaluation überhaupt stattfinden kann mussten zunächst Kriterien für relevante Soft und Property Goals erstellt werden. Das Ergebnis ist in Tabelle 6.1 dargestellt. Bei der Betrachtung der Kriterien fällt auf, dass oft nur laufende Kosten relevant. Für eine genauere Definition fehlen entsprechende Informationen. Um Anschaffungskosten oder Wartungskosten für Teilsysteme genau beziffern zu können, wären beispielsweise zukünftige Nutzerzahlen notwendig gewesen.

Ziel	**Kriterium**	**Beschreibung**	g_T	g_{max}
FAs nicht zur Last fallen	Zeit für Freischaltung seitens FA	Die Zeit in Minuten, die durch Fachärzte aufgebracht werden muss für eine Freischaltung.	0	10
Schneller Zugriff auf Daten	Zeit bis Zugriff auf Daten erfolgen kann	Die Zeit in Stunden bis der Zugriff auf die angeforderten Daten erfolgen kann	0	24
Schneller Zugriff auf Daten	Zeit bis Account freigeschaltet	Die Zeit in Stunden bis der erstellte Account freigeschaltet ist	0	4
Effizienter Umgang mit Mitteln	Einholen von Patientendaten	Die Kosten pro Einholung von Patientendaten in Euro	0	5
Effizienter Umgang mit Mitteln	Freischaltung	Die Kosten pro Freischaltung eines neuen Benutzers	0	2
Effizienter Umgang mit Mitteln	Authentifizierung	Die Anschaffungskosten für eine Lösung pro Arbeitsplatz in Euro	0	10
Effizienter Umgang mit Mitteln	Benachrichtigung über neue Informationen	Die Kosten in Euro pro Benachrichtigung von 100 Benutzern	0	50
Geringe technische Hürden	Installationszeit	Die Zeit in Minuten zur Installation der Lösung für die gesicherte Kommunikation	0	10
Geringe technische Hürden	Passwortlänge	Die Anzahl Zeichen, die notwendig sind	4	8
Hohe qualitative Dichte der Benachrichtigungen	Eignung	Die Eignung der Lösung bewertet durch Experten auf einer Skala von 1 bis 10. Hinsichtlich verfügbaren Medien, Zeichenlänge, Zielgruppeneignung	10	4

Tabelle 6.1: Kriterien der Soft und Property Goals



(a) Soft und Property Goals (b) Gewichtung (c) Kriterien

Abbildung 6.3: Dialoge zur Vorbereitung der Evaluation


Analysis results

Goal	Weight	Criterion	Target	Tolerable	Datenabruf (0.83)	E-Mail (0.76)	Telefonisch (0.76)	Post (0.12)
Effizienter Umgang mit Mitteln	0.3	Effizienter Umgang mit Mitteln- Einholen von Daten	0	5	0 (1.0)	1 (0.8)	1 (0.8)	3 (0.4)
Schneller Zugriff auf Daten	0.7	Schneller Zugriff auf Daten- Zeit bis Zugriff erfolgt	0	24	6 (0.75)	6 (0.75)	6 (0.75)	48 (0.0)

Abbildung 6.4: Das Ergebnis der Evaluation

Die Evaluation wird nun beispielhaft für das Achievement Goal „Patienteninformationen einholen“ vorgestellt. Im ersten Schritt müssen die verfügbaren Alternativen „Datenabruf“, „E-Mail“, „Telefonisch“ und „Post“ hinsichtlich relevanter Kriterien von Soft bzw. Property Goals bewertet werden. Dazu eigenen sich die Kriterien „Zeit bis Zugriff auf Daten erfolgen kann“ des Ziels „Effizienter Umgang mit Mitteln“ und „Einholen von Patientendaten“ des Ziels „Schneller Zugriff auf Daten“. Zum Start der Evaluation müssen nun die relevanten Soft und Property Goals selektiert werden. Abbildung 6.3a zeigt den entsprechenden Dialog im Editor. Da zwei verschiedene Ziele gewählt wurden kann eine Gewichtung vorgenommen werden. Im Beispiel in Abbildung 6.3b ist der schnelle Zugriff auf die Daten wichtiger als der effiziente Umgang mit Mitteln und wird mit 0.7 bewertet. Abschließend erfolgt die Auswahl der Kriterien durch den Dialog in Abbildung 6.3c. Nach der Bestätigung der Auswahl können die Ergebnisse für das Achievement Goal eingeblendet werden. Abbildung 6.4 zeigt das entsprechende View. Die ersten vier Spalten bieten einen Überblick zu den selektierten Zielen, deren Gewichte, Kriterien sowie Werte. Jede weitere Spalte ist eine entsprechende Alternative mit dem Gesamtwert. Das Ergebnis zeigt, dass ein direkter Datenabruf über das Informationsportal sehr günstig und schnell ist, da keine größeren Transportzeiten zu erwarten sind, wie beispielsweise bei Sendung per Post. Es wird von einer allgemeinen Reaktionszeit ausgegangen, die hier 6 Stunden beträgt. Die Alternative „Datenabruf“ hat zwar mit 0.83 den besten Wert erreicht, der Eintrag ist jedoch gelb gekennzeichnet, da negative Auswirkungen von dieser Lösung auf andere Elemente des Zielmodells bestehen - in diesem Fall auf das Soft Goal „Datenschutzrichtlinie einhalten“. Ebenso ist die Lösung „E-Mail“ rot gekennzeichnet, da es sich nicht um die beste Alternative handelt und sich ebenso negativ auf das genannte Soft Goal auswirkt. Die Lösung für eine solche Konstellation könnte entweder die Nutzung der nächstbesten Alternative sein - in diesem Fall „Telefonisch“ - oder der Versuch die negativen Auswirkungen aufzulösen.

Das Beispiel zeigt, dass die Methode über die Berechnung hinaus nachvollziehbare Ergebnisse liefert. Durch das View in Abbildung 6.4 sind die verwendeten Ziele, deren Gewichtung sowie Kriterien klar erkennbar. Negative Auswirkungen auf andere Elemente oder persönliche Werte werden, wenn auch nicht rechnerisch, beachtet.

6.3 Konfliktanalyse

Für die Konfliktanalyse sind insbesondere die negativen Auswirkungen auf andere Elemente im Zielmodell bzw. auf persönliche Werte von Interesse. Die Analyse des vorlie-

Conflicts

Source	Target	Type	Description
Patientenrestriktionen durchsetzen	Geringe Restriktionen	Negative Labels	Geringe Restriktionen is not satisfiable since there is a negative contribution from Patiente
Hohe Qualität der Fachinformationen	Praktische Informationen zu Fachthemen	Negative Labels	Praktische Informationen zu Fachthemen is not satisfiable since there is a negative contrit
Ausreichend Zeit zur Beobachtung	Geringe Aufenthaltszeit für Patienten	Role-based	The stakeholder NZArzt is also member of the work stakeholder group Klinikleitung, there
Ausreichend Zeit zur Beobachtung	Geringe Aufenthaltszeit für Patienten	Negative Labels	Geringe Aufenthaltszeit für Patienten is not satisfiable since there is a negative contributic
Patientenrestriktionen durchsetzen	Fachinformationen verwalten	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Erfolgreiche Behandlung	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Verbesserte Schulung von FAs	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Geringe Distanz zu Klinik	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Zuordnung zu Fortbildungen	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Hohe Zufriedenheit bei Patienten	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	FAs über Updates informieren	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Praktische Informationen zu Fachthemen	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Patienteninformationen einholen	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Geringe Aufenthaltszeit für Patienten	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Gute Behandlung für Patienten	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Fachinformationen durchsuchen	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak
Patientenrestriktionen durchsetzen	Schneller Zugriff auf Daten	Value-based	Element Patientenrestriktionen durchsetzen hurts the personal value Self-Direction of stak

Abbildung 6.5: View mit gefundenen Konflikten

genden Zielmodells konnte insgesamt 17 verschiedene potentielle Konflikte identifizieren. Abbildung 6.5 zeigt das entsprechende View mit dem Analyseergebnis.

Drei der 17 Konflikte sind durch negative Auswirkungen auf andere Elemente bedingt. Diese entstehen durch die vorhandenen UND-Relationen im Zielmodell zwischen den betreffenden Elementen. So besteht beispielsweise eine negative Auswirkung vom Soft Goal „Hohe Qualität der Fachinformationen“ und dem Soft Goal „Praktische Informationen zu Fachthemen“. Beide Elemente sind zur Erreichung höherer Ziele notwendig, wie beispielsweise „Guter Ruf“. Solange diese Problematik jedoch besteht und die Forscher eine hohe wissenschaftliche Qualität der Fachinformationen unbedingt beibehalten möchten, können daraus kaum einfache, praktische Informationen extrahiert werden, da dies dem Streben der Forscher widerspricht.

Ein rollenbasierter Interessenkonflikt konnte ebenfalls identifiziert werden. Das Ziel des Arztes des Nierenzentrums „Ausreichend Zeit zur Beobachtung“ um eine gute Behandlung für die Patienten zu bieten, wirkt sich negativ auf das Ziel „Geringe Aufenthaltszeit für Patienten“ aus an dem Manager, Patient und Klinikleitung interessiert sind. Bei der Klinkleitung handelt es sich um eine Work Stakeholder Group. Der Arzt des Nierenzentrums kann ebenso der Klinikleitung angehören, wodurch der Konflikt entsteht.

Für die dritte Konfliktart bezüglich negativer Auswirkungen auf persönliche Werte von Stakeholdern wurden 13 potentielle Konflikte ermittelt. Jedoch handelt es sich davon bei lediglich einem Konflikt um ein tatsächliches Problem. Dieser in Abbildung 6.6 markiert. Das Ziel „Patientenrestriktionen durchsetzen“ ist mit einer negativen Auswirkung auf den persönlichen Wert „Selbstbestimmung“ versehen. Dem Facharzt wurde dieser persönliche Wert zugeordnet. Es könnte somit zu Problemen kommen, wenn der

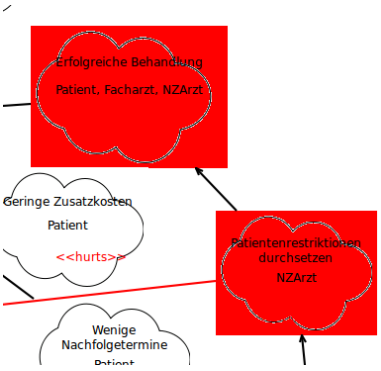


Abbildung 6.6: Konflikt

Arzt des Nierenzentrums versucht, dem Facharzt Vorgaben zu machen wie eine weitere Behandlung aussehen sollte. Die hohe Anzahl an nicht relevanten Konflikten dieser Art resultiert aus dem Erkennungsverfahren. Dieses kann den Kontext der Ziele nicht erfassen. Somit wird potentiell jedem Element, das in einer Beziehung mit dem negativ gekennzeichneten Element steht, Konfliktpotential zugesprochen. Ob es sich tatsächlich um einen Konflikt handelt, kann nur durch einen Analysten bewertet werden.

Der Konflikt bezüglich des persönlichen Werts ist ein gutes Beispiel für die erfolgreiche Nutzung der Analysetechniken. Der Vorteil besteht darin, dass eine negative Auswirkung auf einen persönlichen Wert des Elements ausreicht, um Konflikte bei anderen Elementen zu erkennen. Es muss also nicht, wie bei den Konflikten die durch negative Auswirkungen an andere Elemente gefunden werden können, ein expliziter Zusammenhang hergestellt werden.

6.4 Erfahrungen

Die Umsetzung der Fallstudie mit dem Editor zeigte, dass die Konzepte der definierten Methode funktionieren. Es ist nun erstmals möglich in einem Zielmodell persönliche Werte von Stakeholdern und somit deren persönliche Ziele zu vermerken und andere Ziele hinsichtlich persönlicher Werte zu kennzeichnen. Die Bedeutung der Stakeholder wird dadurch in den Vordergrund gerückt und sie erhalten mehr Beachtung. Dies zeigte sich nicht zuletzt bei der Konfliktanalyse. Das Verfahren zur Analyse von Alternativen konnte ebenso erfolgreich angewandt werden. Die große Stärke lag hier vor allem in der Nachvollziehbarkeit der ermittelten Werte. Auch die Möglichkeit zur Visualisierung von Teilausschnitten konnte sich bewähren und trotz der vielen Elemente im Modell für einen guten Überblick sorgen. Ein Umstand der in der Fallstudie nicht zum Tragen kam, war die weitere Nutzung der definierten Ziele im Softwareentwicklungszyklus. Durch die Nutzung von UNICASE können nun in weiteren Schritten Prozesse oder auch Use Cases definiert werden, die sich mit den definierten Zielen verknüpfen lassen. So ist die Verfolgbarkeit der Anforderungen stets gewährleistet.

Allgemein ist die Nutzung des Editors ohne Probleme möglich. Jedoch muss man beachten, dass es sich derzeit um eine erste prototypische Version handelt. So ist die Verwaltung von mehreren Bewertungen für eine Alternative derzeit schwierig. Wenn man beispielsweise im Rahmen der Fallstudie evaluieren möchte, ob man zur Erfüllung der Anforderungen eine Individualsoftware oder eine Standardsoftware einsetzten sollte, erfordert dies die komplette Neubewertung aller Alternativen. Eine Erweiterung des Editors, die verschiedene „Durchläufe“ des Modells verwalten kann, wäre hierzu wünschenswert.

Zur allgemeinen Erstellung des Zielmodells ist zu sagen, dass die alleinige Definition von Zielen anhand von Informationen in Textform schwierig ist, da nicht die Möglichkeit besteht, Stakeholdern gezielte Fragen zu stellen. Insbesondere bei der Verfeinerung von Zielen oder der Bewertung von Alternativen könnten dadurch wesentlich mehr Informationen gewonnen werden.

7 Fazit und Ausblick

7.1 Fazit

Im Rahmen dieser Arbeit konnte erfolgreich eine Methode zur Integration von persönlichen Werten in den Bereich der Zielmodellierung erarbeitet und in Form eines Plugins für UNICASE umgesetzt werden. Basierend auf der Betrachtung vorhandener Ansätze zur Zielmodellierung und der Kritik an diesen Ansätzen wurden zunächst Anforderungen für eine entsprechende Methode erarbeitet. Dabei wurden insbesondere die Möglichkeiten zur Analyse des Zielmodells auf Alternativen und Konflikte in Form einer Evaluation untersucht. Zur Untersuchung von Alternativen wurde der Ansatz von Lamsweerde [44] ausgewählt. Zur Analyse von Konflikten fand sich kein geeignetes Verfahren, da entweder eine sehr starke Formalisierung vorausgesetzt wurde oder aber der manuelle Aufwand sehr hoch war. Dennoch konnten Teilideen verschiedener Ansätze genutzt werden, um schließlich eine Methode zu definieren, welche die Nutzung von persönlichen Werten in Zielmodellen erlaubt und sie in neu definierte Analysemethoden integriert.

Die Methode bietet ein nachvollziehbares Vorgehen zur Analyse von Alternativen und kann drei Konfliktarten erkennen. Die Fallstudie zeigte die erfolgreiche Anwendung der Methode und die Verwendung der persönlichen Werte. Dadurch ist es erstmals möglich, persönliche Werte explizit in ein Zielmodell einzubringen. Somit wird den Stakeholdern im Zielmodell mehr Aufmerksamkeit bei der Modellierung zugesprochen. Ebenso erweisen sich die persönlichen Werte als erster Anhaltspunkt zur Mediation zwischen verschiedenen Stakeholdern bei Konflikten.

Durch die Integration in UNICASE konnte zudem die Verknüpfung zwischen zielorientierten und prozess-, sowie aufgabenorientierten Elementen hergestellt werden. Somit können Ziele als Referenz für Nachfolgende Elemente dienen, wodurch eine nachvollziehbare Definition aller Anforderungen erfolgen kann. Die Untersuchung vorhandener Werkzeuge zur Zielmodellierung zeigte, dass bisher nur ein Werkzeug diese Möglichkeit bietet. Objectiver ist jedoch ein kommerzielles Werkzeug, während UNICASE unter einer Open-Source-Lizenz steht.

7.2 Ausblick

Da diese Arbeit nur ein erster Schritt zu einer funktionierenden Methode sein kann, ergeben sich verschiedene Ansatzpunkte zur Weiterentwicklung. So war es im Rahmen dieser Arbeit nicht möglich, die definierte Notation auf Benutzerfreundlichkeit hin zu testen.

Dies erfordert ausführliche Tests mit einer Vielzahl an Teilnehmern. Dies ist zwar sehr aufwendig, jedoch hat die Betrachtung der Kritik an bestehenden Ansätzen gezeigt, dass oft die Notation in Frage gestellt wird. Eine ausführliche Betrachtung könnte somit dafür sorgen, dass langfristig eine höhere Akzeptanz erreicht werden kann.

Notwendigkeit zur Weiterentwicklung besteht auch beim Vorgehen zur Definition des Zielmodells. Bisher fehlen gute Hilfestellungen - Muster könnten hier Abhilfe schaffen. Gleiches gilt für die Bewertung von Elementen hinsichtlich persönlicher Werte. Die Ausarbeitung einfacher Regeln für die Zuordnung bildet möglicherweise einen Ansatzpunkt.

Die Einführung von Durchläufen für die Analyse stellt eine sinnvolle Ergänzung für die Umsetzung in UNICASE dar. Dadurch wäre die Verwaltung mehrerer Bewertungen für Elemente leichter möglich. Es handelt sich dabei praktisch um eine weitere Evaluation auf einer höheren Ebene. Durch die Nutzung von UNICASE wären außerdem weitere Evaluationen denkbar, beispielsweise hinsichtlich der Frage, ob für die definierten Ziele entsprechende Prozesse spezifiziert wurden. Dadurch wäre UNICASE zur Überprüfung des IT-Alignments geeignet.

Das Konzept zur Visualisierung von Teilausschnitten des Zielmodells könnte man des Weiteren um verschiedene Optionen erweitern. Denkbar wäre beispielsweise die Visualisierung von lediglich den Elementen, die eine UND-Relation zu den gewählten Elementen besitzen oder denen ein bestimmter Stakeholder zugeordnet wurde.

Bezüglich der Erkennung von Konflikten ist es derzeit in UNICASE nicht möglich bestimmte Einträge zu ignorieren, die sich als nicht relevant herausstellen. In der Fallstudie zeigte sich beispielsweise, dass viele irrelevante Konflikte bezüglich persönlichen Werten gefunden wurden. Das ist bei weiteren Analysen störend, da diese Einträge immer wieder auftauchen.

A Die Entitäten des Metamodells

A.1 Schicht: Entities

Goal

Das Goal Element ist eine abstrakte Entität, die eine bestimmte Eigenschaft oder einen bestimmten Zustand beschreibt den ein oder mehrere Stakeholder anstreben. In konkreter Form handelt es sich um ein Property Goal oder ein Achievement Goal. Goals können zueinander in Verbindung stehen, so dass ein Goal eines oder mehrere übergeordnete Goals hat.

Beziehungen

- Ein Goal kann durch einen oder mehrere Stakeholder angestrebt werden
- Ein Goal in Relation zu einem oder mehreren übergeordneten Goals stehen
- Ein Goal kann durch eine oder mehrere Solutions erfüllt werden

Achievement Goal

Ein Achievement Goal beschreibt einen anzustrebenden Zustand eines Systems, der durch einen oder mehrere Prozesse des Systems erreicht werden kann und Stakeholdern zugeordnet ist.

Beziehungen

- Es kann durch einen oder mehreren Prozesse eines Systems erreicht werden
- Ein oder mehrere Stakeholder verfolgen ein Achievement Goal

Property Goal

Ein Property Goal beschreibt eine gewünschte Eigenschaft eines Systems oder eines Prozesses. Bei einer solchen Eigenschaft handelt es sich typischerweise um eine bestimmte zu erreichende Qualität. Stakeholder können Property Goals zugeordnet werden.

Beziehungen

- Es bezieht sich auf ein oder mehrere Systeme oder Prozesse
- Ein oder mehrere Stakeholder verfolgen ein Property Goal

Soft Goal

Dabei handelt es sich um einen Spezialfall eines Property Goals. Es definiert ein bestimmtes Qualitätsziel im Bezug auf ein System oder einen Prozess. Die Erfüllung eines solchen Ziels ist oft nur in Teilen möglich und es fehlt eventuell an klaren Definitionen wann das Ziel erreicht ist.

Beziehungen

- Es bezieht sich auf ein oder mehrere Systeme oder Prozesse
- Ein oder mehrere Stakeholder verfolgen ein Soft Goal

Solution

Eine Solution bezeichnet eine mögliche Lösung zur Erfüllung eines oder mehrerer Goals und kann durch einen oder mehrere Stakeholder unterstützt werden. Das Vorgehen einer Solution ist durch einen oder mehrere Prozesse definiert. Die Solution bietet die Möglichkeit zur unscharfen Definition von zukünftigem Prozessverhalten zu Beginn der Zieldefinition. Es ist nicht notwendig genau zu wissen, um welche bestimmte Form von Process es sich handelt. Diese Information kann durch die spätere Zuordnung von konkreten Process-Elementen erfolgen.

Beziehungen

- Einer Solution kann einer oder mehrere Stakeholder zugeordnet werden
- Eine Solution kann zur Erfüllung eines oder mehrerer Goals beitragen
- Eine Solution kann durch mehrere Process-Elemente realisiert werden

Beziehungen

- Es bezieht sich auf ein oder mehrere Systeme oder Prozesse
- Ein oder mehrere Stakeholder verfolgen ein Soft Goal

Stakeholder

Unter einem Stakeholder versteht man eine Rolle oder eine organisatorische Einheit, die ein Interesse an bestimmen Zielen bezüglich des zu entwickelnden Systems hat. Ein Stakeholder kann, aber muss nicht an der Ausführung von Prozessen beteiligt sein.

Beziehungen

- Ein Stakeholder kann mehrere Achievement oder Property Goals verfolgen
- Er kann an der Ausführung von Business Processes beteiligt sein
- Er kann mehrere Solutions anstreben

System

Die Entität System ist abstrakt und kann in konkreter Form als Business System, Work System oder IT System auftreten. Ihm können Property Goals zu Beschreibung einer gewünschten Eigenschaft zugeordnet werden. Das Verhalten eines Systems beschreibt eine nichtleere Menge von Process Elementen.

Beziehungen

- Das Verhalten eines Systems beschreiben ein oder mehrere Processes
- Mehrere Property Goals können sich auf ein System beziehen

Process

Das Process-Element ist abstrakt und beschreibt allgemein das Verhalten eines System zur Erreichung eines oder mehrerer Achievement Goals. Dabei werden Data Elemente verwendet oder gar erzeugt. Weiterhin können einem Process auch mehrere Property Goals zugewiesen werden. Konkrete Formen eines Process sind Business Process und BPIT Activity.

Beziehungen

- Ein Process nutzt oder erzeugt mehrere Data Elemente
- Ein Process wird ausgeführt um eines oder mehrere Achievement Goals zu erreichen
- Einem Process können mehrere Property Goals zugeordnet werden
- Process-Elemente beschreiben das Verhalten eines Systems

Data

Data steht allgemein für beliebige Formen von Daten, die durch mehrere Processes genutzt oder auch erzeugt werden.

Beziehungen

- Data wird durch mehrere Processes genutzt oder erzeugt

A.2 Schicht: Business Domain

Business Stakeholder

Ein Business Stakeholder ist eine spezielle Form des Stakeholders innerhalb eines Business System. Er ist abstrakt und wird in konkreter Form durch einen Work Stakeholder oder eine Work Stakeholder Group repräsentiert. Ein Business Stakeholder ist nicht zwingend ein Nutzer des zu entwickelnden Systems. Er ist beteiligt an der direkten Ausführung eines Prozesses. Er kann nur Ziele verfolgen, die dem Business System zugeordnet sind.

Beziehungen

- Er kann ein oder mehrere Achievement oder Property Goals verfolgen, die sich auf ein Business System beziehen
- Er führt einen oder mehrere Business Processes aus
- Er wirkt in einem Business System mit

Business System

Ein Business System besteht aus einem oder mehreren Unternehmen in Form von Menschen, Ressourcen und deren organisatorische Struktur. Goals bezüglich eines Business Systems können nur Business Stakeholder formulieren. Das Verhalten eines Business System wird durch Business Processes beschrieben, es beinhaltet somit verschiedene Work Systems, die zur Verwirklichung der Business Processes dienen.

Beziehungen

- Achievement oder Property Goals können im Bezug zu einem Business System stehen
- Das Verhalten eines Business System wird durch einen oder mehrere Business Processes beschrieben
- In einem Business System wirken einer oder mehrere Business Stakeholder mit

Business Process

Ein abstrakter Business Process beschreibt einen Teil des Verhaltens eines Business System und kann in konkreter Form durch Composite Business Process Elemente und Business Process Activities detailliert beschrieben werden. Die Ausführung eines Business Process erfolgt durch Business Stakeholder und dient der Erfüllung eines Goals eines Business Stakeholder.

Beziehungen

- Ein Business Process besteht aus mehreren Business Process Activities
- Er kann durch einen oder mehrere Business Stakeholder werden
- Ein Business Process wird zur Erfüllung eines Goals eines Business Stakeholder ausgeführt

BP Activity

Eine Business Process Activity ist ein Prozess, der durch einen Work Stakeholder ausgeführt wird um ein Achievement oder Property Goal eines Business Stakeholders zu verfolgen. Sie bildet einen Teil des Verhaltens eines Work System ab und ist ebenso Teil eines Business Process. Sie besteht ausschließlich aus BPIT Activities. Außerdem kann eine Business Process Activity Gegenstand einer Work Task sein.

Beziehungen

- Eine Business Process Activity ist einem Business Process zugeordnet
- Eine Business Process Activity ist einem Work System zugeordnet
- Eine Business Process Activity besteht aus einer bis mehreren BPIT Activities
- Eine oder mehrere Business Process Activities können einer Work Task zugeordnet werden

Composite Business Process

Ein Composite Business Process ist ein Business Process, der selbst wieder aus Business Processes zusammengesetzt ist.

Beziehungen

- Ein Composite Business Process kann mehrere Business Processes beinhalten.

A.3 Schicht: Work Domain

Work Stakeholder

Ein Work Stakeholder ist eine spezielle Form eines Business Stakeholder. Es handelt sich um eine Rolle innerhalb eines Work System und um einen zukünftiger Benutzer des zu entwickelnden Systems. Work Stakeholder können einer Work Stakeholder Group zugordnet werden. Der Work Stakeholder kann einzelne Business Process Activities sowie die zugehörigen BPIT Activities und Human Steps ausführen. Daher können Work Stakeholder auch Work Tasks zugeordnet werden.

Beziehungen

- Er führt eine oder mehrere Business Process Activities aus
- Er führt eine oder mehrere BPIT Activities aus
- Ihm können eine oder mehrere Work Tasks zugeordnet werden
- Er kann eines oder mehrere Achievement oder Property Goals verfolgen
- Er kann einer Work Stakeholder Group zugeordnet sein

- Er führt einen oder mehrere Human Steps aus

Work Stakeholder Group

Eine Work Stakeholder Group ist eine spezielle Form eines Business Stakeholders. Es handelt sich dabei um eine organisatorische Einheit. Diese organisatorische Einheit kann aus mehreren Business Stakeholdern bestehen.

Beziehungen

- Eine Work Stakeholder Group besteht aus mehreren Business Stakeholdern

Work System

Das Work System ist Teil eines oder mehrerer Business Systems. Das Verhalten eines Work System ist durch Business Process Activities beschrieben.

Beziehungen

- Ein Work System ist Teil eines Business System.
- Das Verhalten eines Work System ist durch eine oder mehrere Business Process Activities beschrieben
- Achievement oder Property Goals eines Work Stakeholders können im Bezug zu einem Work System stehen

BPIT Activity

Eine BPIT Activity ist ein Prozess, der durch einen Work Stakeholder und optional das IT System durchgeführt wird zur Unterstützung einer oder mehreren Business Process Activities. Somit dient sie auch dazu den Teil eines Achievement Goals mit Bezug zu einem Work Stakeholder zu erfüllen. Das Verhalten der BPIT Activity wird in BPIT Activity Steps beschrieben.

Beziehungen

- Eine BPIT Activity ist Teil von einer oder mehreren Business Process Activities
- Eine BPIT Activity besteht aus einer nichtleeren Menge von BPIT Activity Steps
- Eine BPIT Activity wird durch einen Work Stakeholder ausgeführt

Use Case

Bei einem Use Case handelt es sich um eine Beschreibung eines Anwendungsfalls bezüglich des zu entwickelnden Systems. Er definiert die konkrete Umsetzung eines BPIT Activity Step. Zur Beschreibung des Verhaltens eines Use Cases nutzt man Human und IT System Steps.

Beziehungen

- Ein Use Case beschreibt die konkrete Realisierung von einem oder mehreren BPIT Activity Steps.
- Ein Use Case besteht aus je ein oder mehreren Human und IT System Steps.

Human Step

Es handelt sich dabei um einen Teil eines Use Cases, der lediglich Aktionen von Menschen ohne Einbeziehung eines IT Systems beschreibt. Lediglich Work Stakeholder führen Human Steps aus.

Beziehungen

- Ein Human Step ist Teil eines Use Cases

BPIT Activity Step

Ein BPIT Activity Step beschreibt einen Teil des Verhaltens einer BPIT Activity. Eine konkrete Realisierung eines BPIT Activity Step mit Hilfe des Systems kann durch einen Use Case gegeben werden.

Beziehungen

- Ein BPIT Activity Step kann zu einer oder mehreren BPIT Activities gehören
- Ein BPIT Activity Step kann durch einen Use Case konkret realisiert sein

Work Task

Eine Work Task stellt die Verantwortlichkeit von einem Work Stakeholder bezüglich einer Menge von Business Process Activities dar. Das bedeutet nicht, dass der Work Stakeholder diese Business Process Activities auch selbst ausführt.

Beziehungen

- Eine Work Task bezieht sich auf eine oder mehrere Business Process Activities
- Eine Work Task bezieht sich auf einen Work Stakeholder

A.4 Schicht: IT Domain

IT System

Das IT System besteht aus Software und Hardware Komponenten. Es bietet eine bestimmte Funktionalität für mehrere Work Systems. Das Verhalten des IT Systems wird durch IT System Steps beschrieben.

Beziehungen

- Einer oder mehrere IT System Steps beschreiben das Verhalten des IT Systems
- Achievement oder Property Goals können im Bezug zu einem IT System stehen

IT System Step

Ein IT System Step beschreibt einen Teil eines Use Cases, der sich nur durch Aktionen eines IT Systems auszeichnet. Zur Erfüllung einer Aktion kann eine IT System Function angewendet werden.

Beziehungen

- Ein IT System Step ist einem Use Case zugeordnet
- Ein IT System Step ist einem IT System zugeordnet
- Ein IT System Step kann eine IT System Function anwenden

IT System Function

Eine IT System Function ist eine konkrete Funktionalität eines IT Systems, die im Verlauf eines oder mehrerer IT System Steps angewandt wird.

Beziehungen

- Eine IT System Function wird durch einen oder mehrere IT System Steps angewandt

B Szenario zur Evaluation

Einleitung

Sie sind Requirements Engineer und haben die Aufgabe, ein neues System für das Nierenzentrum (NZ) des Universitätsklinikums Heidelberg zu entwickeln. Dabei handelt es sich um ein Internetportal, das den Wissenstransfer zwischen Klinikum und Ärzten mit einigen Praxen erleichtern soll. Eine Kollegin hat schon Kundeninterviews geführt und ihre Erkenntnisse über Kontext und Stakeholder dokumentiert. Ihre Aufgabe ist es, anhand dieser Dokumentation 1) ein Zielmodell der Stakeholder zu erstellen 2) Eventuell vorhandene Zielkonflikte aufzudecken 3) Die vorhandenen Lösungsvorschläge im Zielmodell abzubilden, durch eigene Lösungsvorschläge zu ergänzen, und anschließend eine Auswertung der möglichen Lösungsalternativen durchzuführen.

Kontext

Im Nierenzentrum werden immer besonders schwere Fälle behandelt, z. B. Nierentransplantationen oder Tumorentfernungen. Ein Patient geht normalerweise immer zuerst zu seinem Hausarzt, wenn er Beschwerden hat. Dann wird er von einem Facharzt mit eigener Praxis übernommen. Wenn ein schwerer Fall festgestellt wird, der die Kompetenz des Facharztes überschreitet, wird er ins Nierenzentrum für Untersuchungen geschickt. Dann wird er operiert, und nach seiner Entlassung wird er weiterhin vom Facharzt betreut. Er braucht aber auch regelmäßig Nachfolgetermine beim Nierenzentrum.

Probleme, die das neue System lösen soll

Transfer von Wissen über Patientengeschichte und Kontext (Facharzt –> NZ)

Der Facharzt in der Praxis weiß viel über die Krankheitsgeschichte eines Patienten, hat ihn womöglich schon jahrelang betreut. Er übermittelt aber nur einen Teil dieses Wissens über die erforderliche medizinische Dokumentation. Wenn ein Arzt aus dem Nierenzentrum mehr Information braucht als in der Dokumentation vorhanden, ist es sehr schwer, die Information zu bekommen. Erstens kann es sein, dass er nicht weiß, dass die Information überhaupt vorliegt (wenn er die Kreatininwerte aus letztem Jahr braucht, weiß

er nicht, dass so eine Untersuchung gemacht wurde, wenn sie nicht vermerkt ist). Zweitens, es dauert lange, den Kontakt zum Facharzt zu erstellen, weil dieser eine beschränkte Arbeitszeit hat (und der Arzt im Nierenzentrum möglicherweise gerade in Nachtschicht arbeitet). Drittens ist es schwer, die Information zu übermitteln, weil diese vertraulich sind. Also kann man nicht einfach die Daten an einer E-Mail anhängen.

Transfer von Wissen über den momentanen Patientenzustand (NZ –> Facharzt)

Der Facharzt hat oft eine enge Beziehung zu seinem Patient und will über seinen Zustand informiert sein. Es ist aber schwierig für ihn, über seine Patienten Informationen herauszukriegen, die gerade in Behandlung sind. Fachärzte, die NZ-Ärzte persönlich kennen, nutzen die direkte Vorwahl des behandelnden NZ-Arztes, aber alle anderen haben schlechte Aussichten.

Transfer von neuem Fachwissen (NZ –> Facharzt)

Das NZ hat eine begrenzte Kapazität und hätte es gern, dass die Fachärzte mehr von der Betreuung der Patienten übernehmen. Dafür müssen die Fachärzte aber die notwendige Kompetenz haben. Darum bemüht sich das Nierenzentrum, neue Forschungserkenntnisse an die Fachärzte zu übermitteln, um eine bessere Unterstützung für den Patient zu sichern.

Stakeholder und ihre Goals

Klinikleiter des Nierenzentrums

Der Klinikleiter in einem Universitätsklinikum ist gleichzeitig Forscher, Arzt und Manager. Als Manager muss er sicherstellen, dass die Organisation des Klinikums stimmt und dass das Klinikum möglichst effizient mit den vorhandenen Mitteln umgeht. Die Krankenkasse zahlt immer einen festen Betrag pro Behandlung, daher hat er Interesse, möglichst viele Patienten zu behandeln, aber jeden Patient so kurz wie möglich im Klinikum zu halten und früh in die Betreuung des Facharztes zu entlassen.

Damit er viele Patienten kriegt, braucht er einen guten Ruf bei den Fachärzten, denn unterschiedliche Universitätskliniken konkurrieren um Patienten - ein Facharzt aus Mannheim entscheidet selbst, ob er seinen Patient nach Heidelberg oder Karlsruhe schickt, und

dabei kann er durchaus das entfernte Klinikum wählen, wenn er da eine bessere Behandlung erwartet. Den Ruf will er durch gute Patientenvorsorge und reibungslose Abwicklung der Zusammenarbeit sicherstellen. Er denkt, dass das neue System jedem Facharzt erlauben kann, einen Account anzulegen. Mit diesem Account soll dem Facharzt ein gesicherter, schneller Kommunikationskanal zum NZ eröffnet werden. Allerdings muss man da unterschiedliche technische Implementierungen sehr genau anschauen, denn es muss eine strenge Vertraulichkeit sichergestellt werden.

Jede Art von Zusatzleistung, die das Klinikum an den Facharzt bieten kann, steigert die Chancen, dass der Facharzt Patienten überweist. Wenn es Informationen gibt, die ein Facharzt braucht, aber nicht leicht findet, will das Klinikum diese dem Facharzt zur Verfügung stellen im Portal (das können Leitlinien oder Ähnliches sein).

Der Klinikleiter des NZ will auch, dass die Fachärzte möglichst viel Wissen haben, damit sie 1. ihre Patienten schon vor der Überweisung korrekt behandeln (so spart man unnötige Behandlung im NZ), 2. besser einschätzen können, wann ein Patient ins NZ muss (so spart man unnötige Untersuchungen bei leichten Fällen und Komplikationen bei schweren Fällen) und 3. möglichst früh den Patient wieder übernehmen können. Dazu bietet das NZ ausführliche Informationen und ist bemüht, diese an den Facharzt zu bringen. Mit dem neuen Portal hätten die Ärzte eine Anlaufstelle, an der sie die Informationen gebündelt finden. Dabei kann es sich um alte, bewährte Information handeln, oder ganz frische Forschungsergebnisse.

Das Nierenzentrum bietet auch Fortbildungen für Fachärzte an. Das ist ein ganz guter Weg, Informationen an sie zu bringen. Es wäre für die Fachärzte bestimmt von Vorteil, wenn sie mit ihrem Login im Portal auch Zugriff auf die Materialien zu Kursen kriegen, die sie besucht haben. Allerdings konkurriert es da auch mit anderen Einrichtungen und muss PR Arbeit leisten, um Interesse bei den Fachärzten zu wecken, so dass sie 1. sich für den Kurs einschreiben und 2. Beim Kurs wirklich was lernen, anstatt nur zu erscheinen, ihren Schein mitzunehmen, und ohne Zuhören zu verschwinden. Dafür möchte das Nierenzentrum einerseits eine gute Beschreibung der Kurse auf der eigenen Webseite haben, andererseits Information darüber in einem Push-Verfahren an die Fachärzte schicken, zum Beispiel als eine Newsletter. Neueste Forschungsergebnisse eignen sich auch gut für den Inhalt eines Newsletters. Allerdings ist der Klinikleiter nicht sicher, ob die Fachärzte so ein Newsletter haben wollen. Im schlimmsten Fall betrachten sie es als eine weitere unsinnige Sendung in ihrer überfüllten Inbox. Er hofft, dass gute Qualität der Newsletter dabei Abhilfe schaffen kann, sowie geringer Umfang, andererseits möchte er so viel Information wie möglich darin unterbringen.

Als Arzt ist der Klinikleiter daran interessiert, dass die Patienten im Klinikum möglichst gut behandelt werden. Er hat nicht mehr die Zeit, Patienten zu behandeln, außer einige besonders schwere Fälle in seinem Spezialgebiet. Er besteht aber darauf, dass das Personal seine Arbeit gewissenhaft erledigt. Es ist besonders schwer für ihn, das für Patienten sicherzustellen, die aus Bettenmangel im Gebäude eines anderen Klinikums liegen, aber eigentlich Nierenpatienten sind.

Als Forscher will der Klinikleiter, dass hochqualitative Forschungsergebnisse im Nierenzentrum erschaffen werden. Er möchte, dass die Fachärzte insbesondere wissenschaftlich interessante Fälle an ihn überweisen. Dafür müssen die Fachärzte informiert sein, in welcher Richtung das Nierenzentrum forscht, und abschätzen können, welche Patienten dazu passen würden. Außerdem kann ein Facharzt durchaus einen Einfluss darauf haben, ob sich ein Patient für eine Studie einschreibt. Dafür muss der Facharzt aber genau das Ziel der Studie kennen, und überzeugt sein, dass sie dem Patient helfen kann. Diese Informationen können gut über das Portal übermittelt werden.

Nierenzentrumarzt

Der NZ-Arzt soll in erster Linie Patienten betreuen. Er hat eine extrem hohe Arbeitsauslastung. Alles, was zusätzliche Arbeit für ihn bringen könnte, ist für ihn problematisch. Er will sich Zeit für seine Patienten nehmen und hasst jegliche administrative Aufgaben, auch wenn ihm bewusst ist, dass sie notwendig sind. Es ist für ihn gut, wenn der Facharzt mehr von der Patientenbetreuung übernehmen kann. Für ihn ist es aber etwas problematisch, wenn die Patienten nur kurz im Krankenhaus sind - er hätte lieber weniger Patienten, die er dafür ausführlicher betreuen könnte. Da das nicht möglich ist, ist er auf enge Zusammenarbeit mit dem Facharzt angewiesen. Er will ausführliche medizinische Dokumentation, wenn der Patient zum ersten Mal zu ihm kommt. Er will auch später leicht nach Informationen nachfragen können. Nachdem der Patient entlassen wurde, will der NZ-Arzt, dass der Patient seine Anweisungen genau befolgt. Er hat aber zu wenig Kontakt zum Patient, deswegen muss es der Facharzt durchsetzen. Wenn der Facharzt selbst nicht den unmittelbaren Nutzen z. B. einer strengen Diät sieht, wird er auch nicht darauf achten, dass sie der Patient einhält.

Der NZ-Arzt ist auch ein Forscher. Er führt eigene Forschung durch, leitet Studien. Er braucht genug Patienten mit bestimmten Erkrankungen dafür - also umso mehr Patienten das Klinikum behandelt, desto besser für seine Forschung, weil das die Wahrscheinlichkeit steigt, dass genug passende Patienten zur Verfügung stehen. Wenn der Facharzt die Patienten motiviert hat, ist das auch für ihn gut.

Er ist daran interessiert, dass Fachärzte möglichst viel Wissen haben, hat aber nicht die Zeit, sein eigenes Wissen in eine Form aufzubereiten, die für Fachärzte geeignet ist. Wenn er seine Arbeit vorstellen soll, benutzt er seine Publikationen. Wenn er auf Weiterbildungsseminare vorträgt, benutzt er die Vorträge, die er für andere Forscher auf Konferenzen vorbereitet hat.

Facharzt mit eigener Praxis

Der Facharzt für Urologie hat eine geringere Arbeitslast als die NZ-Ärzte. Er betreut die Patienten länger, und hat eine stärkere Beziehung zu ihnen. Das Heilen steht für ihn im

Vordergrund, er ist nicht an Forschung interessiert. Er will seine Patienten im besten verfügbaren Nierenklinik unterbringen. Dabei spielen Distanz und Ruf der Klinik eine Rolle. Ärzte aus der Region haben schon gute Kontakte zum Nierenzentrum in Heidelberg, und kennen die Abläufe und ihre Kollegen im NZ. Ein Facharzt aus einem entfernten Ort wird sehr genau überlegen, bevor er seinen Patient nach Heidelberg schickt, auch wenn er wissen sollte, dass es in Heidelberg Spezialisten für die exakte Krankheit des Patienten gibt - nicht nur ist die Fahrt (inklusive Folgetermine) eine Belastung für Patienten, er ist auch unsicher, wie alles ablaufen wird. Es ist aber auch relativ unwahrscheinlich, dass er wissen wird, dass Heidelberg in dieser Krankheit mehr Erfahrung hat als andere Kliniken.

Der Facharzt möchte jedem Patient helfen können, hat aber nach seinem Studium nicht mehr viel Zeit (und oft zu wenig Motivation), neueste Entwicklungen aus seinem Gebiet zu verfolgen. Er würde bei einem ungewöhnlichen Fall eher sein altes Lehrbuch aus dem Studium konsultieren, als in Bibliotheken und Katalogen zu recherchieren, ob es Leitlinien gibt. Als Praktiker ist er eigentlich an eine Sache interessiert: welche Therapie wende ich an, damit dieser Patient sich verbessert, also Handlungsempfehlungen. Studien will er gar nicht lesen. Sie sind voll mit Statistik, die er nicht länger versteht, und enthalten keine Handlungsempfehlungen. Deswegen sind Fortbildungen für ihn auch oft langweilig - er sieht, was die NZ-Ärzte geforscht haben, aber er kann es nicht verwenden.

Es ist für den Facharzt wichtig, immer über sein Patient Bescheid zu wissen, auch wenn dieser im Nierenzentrum liegt.

Patient

Der Patient will vor allem geheilt werden. Er möchte möglichst wenig Zeit im Krankenhaus verbringen. Er will möglichst keine zusätzlichen Kosten, aber er würde schon Behandlungen bezahlen, die keine Kassenleistung sind, wenn er überzeugt ist, dass diese ihm helfen. Er will von seinen Ärzten Information auf sehr einfachem Niveau. Die meisten Patienten sind schon interessiert, was mit ihnen los ist, wollen aber darüber hinaus möglichst wenig wissen - zum Beispiel welche Tablette was macht. Sie fühlen sich durch zu viele Medikamente und Restriktionen oft überfordert, fürchten Nebenwirkungen, und lassen Medikamente gerne aus, wenn ihre Symptome verschwinden, auch wenn der Arzt sie für längere Zeit verordnet hat. Sie wollen möglichst wenig Änderungen in ihrem Leben durch die Behandlung - Diäten, Medikamente und lange Fahrten für die Folgetermine im Nierenzentrum gehören dazu.

Persönliche Werte

Die Werte von repräsentativen Mengen von Stakeholdern wurden erhoben. Der Klinikleiter hat die Werte Selbstbestimmung und Macht. Bei den Nierenzentrumärzten sind die

häufigsten Werte Leistung und Benevolenz. Die Fachärzte mit eigener Praxis haben vorwiegend Benevolenz und Selbstbestimmung. Die Werte von Patienten wurden nicht erhoben, da sie keine einheitliche Population darstellen.

Literaturverzeichnis

[1] Shalom H. Schwartz, Gila Melech, Arielle Lehmann, Steven Burgess, Mari Harris, and Vicki Owens. Extending the cross-cultural validity of the theory of basic human values with a different method of measurement. *Journal of Cross-Cultural Psychology*, 32(5):519–542, 2001.

[2] Eric S. K. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering*, RE '97, pages 226–, Washington, DC, USA, 1997. IEEE Computer Society.

[3] Axel van Lamsweerde. Requirements engineering in the year 00: A research perspective, 2000.

[4] C. Ponsard, P. Massonet, A. Rifaut, J. F. Molderez, A. Van Lamsweerde, and H. Tran Van. Early verification and validation of mission critical systems, 2004.

[5] Rumyana Proynova, Barbara Paech, Sven H. Koch, Andreas Wicht, and Thomas Wetter. Investigating the influence of personal values on requirements for health care information systems. In *Proceeding of the 3rd workshop on Software engineering in health care*, SEHC '11, pages 48–55, New York, NY, USA, 2011. ACM.

[6] Steven J. Bleistein. Strategic alignment in requirements analysis for organizational it: An integrated approach. In *In: 20th ACM Symposium on Applied Computing (SACÕ05). ACM*, 2005.

[7] Technische Universität München. What is unicase? `https://teambruegge.informatik.tu-muenchen.de/groups/unicase/wiki/bdc81/What_is_unicase_.html`, 2011. [Online; Stand 02. September 2011].

[8] I. Alexander and L. Beus-Dukic. *Discovering Requirements: How to Specify Products and Services*. Wiley, 2009.

[9] *Goal-oriented requirements engineering: a guided tour*, Toronto, 2001. IEEE Computer Society.

[10] Daniel Amyot, Jennifer Horkoff, Daniel Gross, and Gunter Mussbacher. A lightweight grl profile for i* modeling. In *Proceedings of the ER 2009 Workshops (CoMoL, ETheCoM, FP-UML, MOST-ONISW, QoIS, RIGiM, SeCoGIS) on Advances in Conceptual Modeling - Challenging Perspectives*, ER '09, pages 254–264, Berlin, Heidelberg, 2009. Springer-Verlag.

[11] Anne Dardenne, Axel van Lamsweerde, and Stephen Fickas. Goal-directed requirements acquisition. In *SCIENCE OF COMPUTER PROGRAMMING*, pages 3–50, 1993.

[12] S. H. Schwartz and W. Bilsky. Toward a universal psychological structure of human values. *Journal of Personality and Social Psychology*, 53(3):550–562, 1987.

[13] Meg J. Rohan and Mark P. Zanna. *Values and Ideologies*, pages 458–478. Blackwell handbook of social psychology. Blackwell, 2002.

[14] Emmanuel Letier. *Reasoning about Agents in Goal-Oriented Requirements Engineering*. PhD thesis.

[15] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. The Temporal Logic of Reactive and Concurrent Systems. Springer-Verlag, 1992.

[16] Axel van Lamsweerde, Emmanual Letier, and Robert Darimont. Managing conflicts in goal-driven requirements engineering. *IEEE Trans. Softw. Eng.*, 24:908–926, November 1998.

[17] J. Mylopoulos, L. Chung, and B. Nixon. Representing and using nonfunctional requirements: A process-oriented approach. *IEEE Transactions on Software Engineering*, 18:483–497, 1992.

[18] John Mylopoulos, Lawrence Chung, and Eric Yu. From object-oriented to goal-oriented requirements analysis. *Commun. ACM*, 42:31–37, January 1999.

[19] Lin Liu, Eric Yu, and John Mylopoulos. Security and privacy requirements analysis within a social setting. In *Proceedings of the 11th IEEE International Conference on Requirements Engineering*, pages 151–, Washington, DC, USA, 2003. IEEE Computer Society.

[20] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. Tropos: An Agent-Oriented Software Development Methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.

[21] Paolo Giorgini, Fabio Massacci, John Mylopoulos, Nicola Zannone, Paolo Giorgini, Fabio Massacci, Nicola Zannone, and John Mylopoulos. Modeling security requirements through ownership, permission and delegation. In *In Proc. of RE'05*, pages 167–176. IEEE Press, 2005.

[22] Interational Telecommunication Union (ITU). Z.151 : User requirements notation (URN) - Language definition. `http://www.itu.int/`, 2008.

[23] Frank Zickert. Evaluation of the goal-oriented requirements engineering method kaos. *AMCIS 2010 Proceedings*, 2010.

[24] Axel van Lamsweerde. Goal-oriented requirements enginering: A roundtrip from research to practice. *Requirements Engineering, IEEE International Conference on*, 0:4–7, 2004.

[25] Evangelia Kavakli and Pericles Loucopoulos. Goal modeling in requirements engineering: Analysis and critique of current methods. In *Information Modeling Methods and Methodologies*, pages 102–124. Idea Group, 2005.

[26] Raimundas Matulevičius. Improving the syntax and semantics of goal modelling languages. In *Proceedings of the 3rd International i* Workshop*, volume 322 of *CEUR Workshop Proceedings*, pages 75–78, Recife, Brazil, 2008. CEUR-WS.org.

[27] Daniel Moody, Patrick Heymans, and Raimundas Matulevičius. Visual syntax does matter: improving the cognitive effectiveness of the i* visual notation. *Requirements Engineering*, 15:141–175, 2010.

[28] Raimundas Matulevičius and Patrick Heymans. Visually effective goal models using kaos. In *Advances in Conceptual Modeling – Foundations and Applications*, volume 4802 of *Lecture Notes in Computer Science*, pages 265–275. Springer Berlin / Heidelberg, 2007.

[29] Object Management Group (OMG). Business Process Model and Notation (BPMN), Version 2.0. `http://www.omg.org/`, 2010.

[30] G. Koliadis, A. Vranesevic, M. Bhuiyan, A. Krishna, and A. Ghose. Combining i* and bpmn for business process model lifecycle management. In *Proceedings of BPM 2006 International Workshops, Vienna, Austria, September 4-7, 2006*, volume 4103 of *Lecture Notes in Computer Science*, pages 416–427, 2006.

[31] Ken Decreus and Geert Poels. Mapping semantically enriched formal tropos to business process models. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 371–376, New York, NY, USA, 2009. ACM.

[32] Christopher J. Pavlovski and Joe Zou. Non-functional requirements in business process modeling. In *Proceedings of the fifth Asia-Pacific conference on Conceptual Modelling - Volume 79*, APCCM '08, pages 103–112, Darlinghurst, Australia, Australia, 2008. Australian Computer Society, Inc.

[33] Object Management Group (OMG). Unified Modeling Language, Superstructure Version 2.3. `http://www.omg.org/`, 2010.

[34] Victor F. A. Santander and Jaelson Castro. Deriving use cases from organizational modeling. In *Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering*, RE '02, pages 32–42, Washington, DC, USA, 2002. IEEE Computer Society.

[35] Alistair Cockburn. *Writing Effective Use Cases*. Addison-Wesley Professional, 2000.

[36] Soren Lauesen. *User Interface Design: A Software Engineering Perspective*. Addison Wesley, 2005.

[37] Raimundas Matulevičius, Patrick Heymans, and Guttorm Sindre. Comparing goal-modelling tools with the re-tool evaluation approach. *Information Technology And Control*, 35A, No. 3:276–284, 2006.

[38] Lin Liu and Eric Yu. Designing information systems in social context: a goal and scenario modelling approach. *Inf. Syst.*, 29:187–203, April 2004.

[39] Haihua Xie, Lin Liu, and Jingwei Yang. i*-prefer: optimizing requirements elicitation process based on actor preferences. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, pages 347–354, New York, NY, USA, 2009. ACM.

[40] Kazuma Yamamoto and Motoshi Saeki. Using attributed goal graphs for software component selection: an application of goal-oriented analysis to decision making. In *Tutorials, posters, panels and industrial contributions at the 26th international conference on Conceptual modeling - Volume 83*, ER '07, pages 215–220, Darlinghurst, Australia, Australia, 2007. Australian Computer Society, Inc.

[41] Haruhiko Kaiya, Hisayuki Horai, and Motoshi Saeki. Agora: Attributed goal-oriented requirements analysis method. In *Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering*, RE '02, pages 13–22, Washington, DC, USA, 2002. IEEE Computer Society.

[42] Sotirios Liaskos, Sheila A. McIlraith, Shirin Sohrabi, and John Mylopoulos. Integrating preferences into goal models for requirements engineering. In *Proceedings of the 2010 18th IEEE International Requirements Engineering Conference*, RE '10, pages 135–144, Washington, DC, USA, 2010. IEEE Computer Society.

[43] Emmanuel Letier and Axel van Lamsweerde. Reasoning about partial goal satisfaction for requirements and design engineering. *SIGSOFT Softw. Eng. Notes*, 29:53–62, October 2004.

[44] Axel Lamsweerde. Conceptual modeling: Foundations and applications. chapter Reasoning About Alternative Requirements Options, pages 380–397. Springer-Verlag, Berlin, Heidelberg, 2009.

[45] Vic Chung. Considering role-based conflicts of interest in analyzing and designing e-health systems with goal-oriented methodologies. In *Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services*, PST '06, pages 78:1–78:4, New York, NY, USA, 2006. ACM.

[46] Paolo Giorgini, Fabio Massacci, John Mylopoulos, and Nicola Zannone. Detecting conflicts of interest. In *Proceedings of the 14th IEEE International Requirements Engineering Conference*, pages 308–311, Washington, DC, USA, 2006. IEEE Computer Society.

[47] Department of Computer Science University of Toronto. Openome, an open-source requirements engineering tool. https://se.cs.toronto.edu/trac/ome/, 2011. [Online; Stand 05. Juni 2011].

[48] Department of Computer Science University of Ottawa. jucmnav - eclipse plugin for the user requirements notation. http://jucmnav.softwareengineering.ca/jucmnav/, 2011. [Online; Stand 05. Juni 2011].

[49] Gunter Mussbacher, Sepideh Ghanavati, and Daniel Amyot. Modeling and analysis of urn goals and scenarios with jucmnav. *Requirements Engineering, IEEE International Conference on*, 0:383–384, 2009.

[50] Respect-IT. Engineer your requirements with objectiver. http://www.objectiver.com/, 2011. [Online; Stand 05. Juni 2011].

[51] A. Rifaut T. Delor, R. Darimont. Software quality starts with the modelling of goal-oriented requirements. In *Proceedings of the 16th International Conference on Software and Systems Engineering and their Applications*, Paris, December 2-4, 2003.

[52] Nan Ma, Tracy Hall, and Trevor Barker. Building a narrative based requirements engineering mediation model. *Software Process Improvement*, 16:1–12, 2008.

[53] Eclipse Foundation. Graphical modeling project (gmp). http://www.eclipse.org/modeling/gmp/, 2011. [Online; Stand 08. September 2011].

[54] ISO/IEC. *ISO/IEC 9126. Software engineering – Product quality*. ISO/IEC, 2001.

Eidesstattliche Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig und unter ausschließlicher Verwendung der angegebenen Literatur und Hilfsmittel erstellt zu haben.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Heidelberg, 14. November 2011

_____

Markus Fischer