Eignungseinstufung von Vorgehensmodellen

Alexander Ziegler

Dissertation
zur Erlangung des Grades eines Doktors der Ingenieurswissenschaften
-- Dr. Ing. --

Vorgelegt im Fachbereich 3 (Mathematik und Informatik)
Universität Bremen

06.02.2009

Datum des Promotionskolloquiums: 06.02.2009

Gutachter
Prof. Dr. Bernd Krieg Brückner (Universität Bremen & DFKI Lab Bremen)
Dr. habil Ernest Wallmüller (Qualität und Informatik, Geroldswil, Schweiz)

Danksagungen

Diese Arbeit konnte nur in einem entsprechenden Umfeld entstehen. Prof. Dr. Bernd Krieg-Brückner danke ich für sein Vertrauen und seinen Mut für unkonventionelle Konstruktionen. Bei Dr. habil. Ernest Wallmüller bedanke ich mich für wertvolle Anregungen und die Übernahme des Zweitgutachtens.
Dr. Katharina Spies, Florian Deissenböck, Prof. Dr. Clemens Ballarin, Prof. Dr. Gerd Beneken, Dr. Leonid Kof, Tilman Seifert, Patrick Keil, Thomas Kuhn, Dr. Stefan Wagner und speziell Prof. Dr. Dr. h.c. Manfred Broy schulde ich Dank für viele Diskussionen und kritische Anmerkungen, sowie die Möglichkeit, in einem professionellen wissenschaftlichem Umfeld zusammenarbeiten zu können.
Bei Klaus Gollin, Dietrich Olfs, Ingo Rah bedanke ich mich für ihre Unterstützung und kritische Anmerkungen.
Meiner Mutter Lydia Hane, meinem Stiefvater Peter Hane, meiner Großmutter Joana Neumeier, meinen Schwiegereltern Günter und Judit Vetterling danke ich für Ihre Unterstützung.
Meiner Frau Monika Vetterling habe ich für Diskussionen, kritische Anmerkungen, Unterstützung und ihre unendliche Geduld zu danken.

Zusammenfassung

Software ist ein bedeutender Wirtschaftsfaktor. Softwareentwicklung ist komplex und teuer. Um die Komplexität in Softwareentwicklungsprojekten besser beherrschen zu können, wurden Standardprozeßbeschreibungen entwickelt, die üblicherweise als „Vorgehensmodell“ bezeichnet werden. Diese stehen für Softwareentwicklungsprojekte zur Verfügung. Weil die Auswahl eines geeigneten Vorgehensmodells für ein konkretes Projekt schwierig ist, werden Vorgehensmodelle seltener eingesetzt, als nötig wäre.

In der vorliegenden Arbeit wird ein Eignungseinstufungsverfahren für Vorgehensmodelle (EVGM) entwickelt, das die spezifischen Problemschwerpunkte im jeweils gegebenen Projekt gezielt berücksichtigt. Die Problemschwerpunkte sind aus empirischen Studien und entsprechender Fachliteratur abgeleitet. Es wird eine semiformale Modellierungssprache beschrieben, die erfolgsbestimmende Merkmale von Projekten und Vorgehensmodellen, sowie deren wechselseitige Beeinflussungsbeziehungen bereitstellt. Die Merkmale sind durch eine Ontologie hergeleitet und in eine Systematik eingeordnet, für die verdeutlicht wird, daß sie die wesentlichen Aspekte von Softwareentwicklungsprojekten und von Vorgehensmodellen überdeckt. Die Modellierungssprache basiert auf dem System Dynamics Ansatz von Jay W. Forrester, sowie auf einem Derivat von Peter Gomez und Gilbert Probst. Mit ihr können Ursache-Wirkungs-Diagramme entwickelt werden, welche die wichtige Einflüsse und deren Wechselwirkungen im Projekt abbilden.

Durch diese vernetzte Modellierung können alle wesentlichen Probleme eines Projektes im Zusammenhang untersucht und geeignet durch ein Vorgehensmodell adressiert werden.
Das EVGM gibt ein modulares, methodisches Schritt-für-Schritt-Verfahren vor, das seine Anwender bei der Auswahl eines Vorgehensmodells unterstützt. Diese Anwender benötigen Detailkenntnisse über Software Engineering und ihr spezifisches Projekt, aber nicht über Vorgehensmodelle. Projektspezifische Veränderungen oder Erweiterungen der Modellelemente durch die Benutzer werden durch das EVGM unterstützt. Die Wirkweise des EVGM wurde durch Fallstudien überprüft. Sie wird durch eine Fallstudie mit einem fiktiven Projekt veranschaulicht. Das EVGM wurde bereits in mehreren industriellen Prozeßverbesserungsprojekten erfolgreich zur Projektanalyse verwendet. Die Benutzung des EVGM wird durch ein Software Werkzeug unterstützt.

Inhaltsverzeichnis

1 Einleitung

„For every complex problem,
there is an answer that is
short, simple, and wrong“
H.L.Mencken, zitiert nach
[McC96]

Die Abwicklung von Tätigkeiten, beispielsweise im alltäglichen Geschäftsleben, der Energieversorgung, im Gesundheitswesen oder Personen- und Güterverkehr wird in immer stärkerem Ausmaß durch Software unterstützt. Dadurch gewinnt die Softwareentwicklung eine große Bedeutung, erhält aber auch eine hohe Verantwortung. Der Einsatz fehlerhafter Software kann zu schwerwiegenden ökologischen oder finanziellen Auswirkungen führen. Auch Menschen können zu Schaden kommen, beispielsweise bei Softwarefehlern in der Steuerung eines Kernreaktors. Der Einsatz ungeeigneter Software verschlechtert die Effizienz der Tätigkeiten, die sie unterstützen soll.
Die Erfolgsquote bei Softwareprojekten ist vergleichsweise niedrig. Zu viele Projekte verfehlen ihre Qualitäts- und Funktionsziele oder verletzen Zeit- und Budgetrestriktionen, wie [EVA00] und [Sta99] belegen. Softwareentwicklungsprojekte sind vergleichsweise komplexe Vorhaben [Chr92]. Daher können sie durch unsystematische Vorgehensweisen bei der Entwicklung nur begrenzt erfolgreich sein. Softwareentwicklungsprojekte bedürfen der Unterstützung durch systematische und definierte Vorgehensweisen, um Software produzieren zu können, welche die erforderliche Qualität aufweist (vgl. [Chr92], [Hör+06] beziehungsweise [Wal07]). Durch methodisches Vorgehen in Softwareprojekten kann die geforderte Qualität mit akzeptablen Kosten und in der vorgegebenen Zeit erzielt werden. In der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04] werden „Unzureichende Softwareentwicklungsmethoden“ als Risikofaktor in Softwareentwicklungsprojekten dargestellt.
Systematische, definierte Vorgehensweisen für Softwareentwicklungsprojekte sind in Form von Vorgehensmodellen verfügbar. Projektbetreiber können sie als Vorlage für ihre Entwicklungsprozesse verwenden. Somit können Vorgehensmodelle einen Beitrag zu einer erfolgreicheren Softwareentwicklung leisten. Trotzdem werden Softwareprojekte laut [EVA00] zu oft ohne ein Vorgehensmodell durchgeführt.
Vor dem Einsatz eines Vorgehensmodells steht die Entscheidung, welches für das aktuelle Projekt am besten geeignet wäre. Der Einsatz eines weniger geeigneten Vorgehensmodells ist nicht förderlich für den Projekterfolg, da es die Lösung der spezifischen Probleme in einem Projekt nicht optimal unterstützt. Vor der Entscheidung für ein Vorgehensmodell ist deshalb die Untersuchung seiner Eignung für ein Projekt erforderlich. Diese Untersuchung erfordert in der Regel eine detaillierte Kenntnis, welche Vorgehensmodelle in welchem Maß für welche Projekte sinnvoll einsetzbar sind. Ein geeignetes Vorgehensmodell muß durch seine Eigenschaften und Bestandteile so beschaffen sein, daß es die Lösung der wesentlichen Probleme im Projekt bestmöglich unterstützt. Vorgehensmodelle sind durch die Vielzahl ihrer Eigenschaften und Bestandteile komplexe Lösungsansätze für Projektprobleme. Die Anzahl der Einflußfaktoren, die wesentlich für einen Projekterfolg sind, sowie deren mangelnde Beständigkeit machen auch Softwareentwicklungsprojekte komplex. Bei einem Vorgehensmodell, welches geeignet

für ein Projekt ist, muß daher eine komplexe Problemlösung auf eine komplexe Problemstellung passen. Diese Untersuchung ist schwierig, wie auch [Dal+05] bestätigt.
Es ist jedoch für Softwareentwickler und Projektleiter nicht mit wirtschaftlich vertretbarem Aufwand möglich, sich neben ihren Hauptaufgaben über die existierenden Vorgehensmodelle und ihre Eigenschaften einen Überblick zu verschaffen. Dies trägt dazu bei, daß häufig auf die Verwendung eines Vorgehensmodells verzichtet wird.
Durch den vermehrten und systematischen Einsatz von Vorgehensmodellen ist eine erfolgreichere Softwareentwicklung erreichbar [Chr92]. Eine Möglichkeit, systematisch und mit vertretbarem Aufwand zu einer fundierten Entscheidung für ein geeignetes Vorgehensmodell zu kommen, schafft bessere Voraussetzungen für die Auswahl und den Einsatz eines geeigneten Vorgehensmodells. Dies muß für Mitarbeiter in Softwareentwicklungsprojekten auch ohne professionellen Hintergrund in Bezug auf Vorgehensmodelle möglich sein. Eine systematische Vorgehensweise macht das komplizierte Unterfangen, ein geeignetes Vorgehensmodell für ein Projekt auszuwählen, für seine Anwender beherrschbar. Durch ein Verfahren, das die Diagnose für ein Projekt unterstützt, kann die adäquate Auswahl von Vorgehensmodellen, sowie ihre Einsatzhäufigkeit erhöht werden.

In der vorliegenden Arbeit wird ein Verfahren zur Eignungseinstufung von Vorgehensmodellen (EVGM) beschrieben. Es unterstützt die Mitarbeiter in Softwareentwicklungsprojekten bei der Entscheidung für ein geeignetes Vorgehensmodell. Mit diesem Verfahren können Projektbetreiber Vorgehensmodelle systematisch hinsichtlich ihrer Eignung für die spezifischen kritischen Probleme in einem Projekt vergleichen und bewerten. Durch ein Verfahren zur Eignungseinstufung von Vorgehensmodellen wird es möglich, Softwareprojekte besser durch erprobte Entwicklungsprozesse zu unterstützen und so einen Beitrag zu ihrem Erfolg zu leisten.

Im folgenden wird dargestellt, welche Ziele die Arbeit hat (Abschnitt 1.1). Der State of the Art beim Einsatz von Vorgehensmodellen wird skizziert (Abschnitt 1.2). Dann wird ein Überblick gegeben, für welche Probleme in Projekten Vorgehensmodelle Lösungen bieten können, um so deren Nutzen für die Softwareentwicklung zu belegen (Abschnitt 1.3). Die vorliegende Arbeit wird in Abschnitt 1.4 mit Hinblick auf ähnliche Ansätze mit dem Thema Vergleich und Bewertung von Vorgehensmodellen positioniert und abgegrenzt. Ein kurzer Überblick bietet eine Vorschau auf den Ablauf des EVGM (Abschnitt 1.5). Die Zusammenhänge zwischen den einzelnen Themenblöcken der vorliegenden Arbeit werden verdeutlicht (Abschnitt 1.6).

Inhalt des Kapitels

1.1 Ziele der Arbeit

Das Eignungseinstufungsverfahren für Vorgehensmodelle (EVGM) unterstützt seine Anwender umfassend und methodisch bei der Entscheidung, welches Vorgehensmodell für ein konkretes Projekt in einer softwareentwickelnden Organisation geeignet ist.
Es betrachtet ausschliesslich reine Softwareentwicklungsprojekte.
Das EVGM leitet seine Anwender an, systematisch die wesentlichen erfolgsbestimmenden Merkmale im Projekt zu analysieren. Diese werden in einem Projektprofil zusammengefasst. Das EVGM fördert dabei die Berücksichtigung der speziellen Eigenschaften des betreffenden Projektes. Aus den wesentlichen Einflußfaktoren im Projekt werden mit dem EVGM Anforderungen an ein geeignetes Vorgehensmodell abgeleitet. Diese Anforderungen werden vom EVGM vorgeschlagen. Aufbauend auf den abgeleiteten Anforderungen an Vorgehensmodelle unterstützt das EVGM den systematischen Vergleich, welches konkrete Vorgehensmodell diese am besten erfüllt. Basierend auf diesem Vergleich können seine Anwender eine Entscheidung bezüglich eines Vorgehensmodells treffen. Die zuvor abgeleiteten Anforderungen an ein geeignetes Vorgehensmodell dienen als Hinweise für die projektspezifische Anpassung dieses Vorgehensmodells. Das EVGM leistet Vorarbeit für die Einführung des Vorgehensmodells. Durch das EVGM wird der Entscheidungsprozeß für ein Vorgehensmodell nachvollziehbar dokumentiert und somit überprüfbar. Das EVGM bietet eine Begriffswelt in Form einer Ontologie und ein Beschreibungsschema in Form einer Systematik zur Charakterisierung und Bewertung von Softwareprojekten und Vorgehensmodellen. Es schafft so eine einheitliche Kommunikationsbasis für seine Anwender. Das EVGM unterstützt seine Anwender durch aufbereitetes Fachwissen über Softwareprojekte und Vorgehensmodelle. Es ist anpassbar und erweiterbar.

Das EVGM ist selbst wie ein Vorgehensmodell aufgebaut. Es besteht aus Rollen, Arbeitsprodukten und Aktivitäten [Chr92], die diese miteinander verbinden.

Zielgruppe für das EVGM. Das EVGM ist für Softwareentwickler und Manager in industriellen, softwareentwickelnden Organisationen vorgesehen. Diese müssen Wissen und Erfahrung über Softwareentwicklungsprojekte, sowie Kenntnisse bezüglich des betreffenden Projektes besitzen. Die Zielgruppe des EVGM muß keinen professionellen Hintergrund in Bezug auf Vorgehensmodelle besitzen.
Das EVGM ist für Laien in Bezug auf Softwareentwicklungsprojekte nicht ohne Weiteres einsetzbar. Es erfordert bei seiner Anwendung Beurteilungen und Entscheidungen bezüglich des betreffenden Projektes. Diese basieren auf dem Wissen und der Erfahrung seiner Anwender in der Softwareentwicklung und bezüglich des betreffenden Projektes.
Zur Zielgruppe des EVGM gehören auch Forschende mit den Interessenschwerpunkten Softwareentwicklungsprojekte und Vorgehensmodelle.

Geeignete Einsatzszenarios für das EVGM. Das EVGM kann für Einzelprojekt- und strategische Unternehmensentscheidungen verwendet werden. Optimalerweise wird es auf Projekte angewendet, die sich in der Phase der Voruntersuchung befinden, aber noch nicht gestartet wurden. Wenn ein Projekt bereits definiert ist, kann es mit Hilfe des EVGM analysiert werden um herauszufinden, welches Vorgehensmodell seine Durchführung am besten unterstützen kann. Nach der Auswahl eines Vorgehensmodells durch die Anwendung des EVGM muß dieses im Rahmen eines Projektes zur Verbesserung von Softwareentwicklungsprozessen in die betreffende Organisation eingeführt werden.

Das EVGM fokussiert die Eignungseinstufung von Vorgehensmodellen und kann in ein umfassendes Projekt zur Verbesserung von Softwareentwicklungsprozessen eingebettet sein. In dessen Rahmen werden die weiterführenden organisatorischen Belange und die tatsächliche Einführung des Vorgehensmodells abgedeckt, die im EVGM nicht enthalten sind.
Schnittstellen zu nicht veränderlichen Ist-Prozessen in der betrachteten Organisation werden im EVGM nicht berücksichtigt. Dies muß durch Anpassung des Vorgehensmodells und im Rahmen seiner Einführung in die Organisation gelöst werden, welche nicht Gegenstand der vorliegenden Arbeit sind.

1.2 Heutige Situation beim Einsatz von Vorgehensmodellen

Wie stellt sich die heutige Situation beim Einsatz von Vorgehensmodellen in Softwareentwicklungsprojekten dar? Um bezüglich dieser Frage die Sicht des Autors durch unabhängige Erkenntnisse zu ergänzen um so ein ein aktuelles, repräsentatives Bild der Situation zu erhalten, wurden Experteninterviews mit Personen aus der deutschsprachigen Softwareprozeßverbesserungs-Industrie [Fic08], [Her08], [Kne08], [Koc08], [Maa08], [Mül08], [Mut08], [Pab08], [Sch08], [Van08], [Wal06] durchgeführt. Diese haben in ihrer Eigenschaft als Assessor oder Appraiser, beziehungsweise als Geschäftsführer von Beratungsunternehmen für Softwareprozeßverbesserung die Möglichkeit, sich ein breites Bild der Situation in Bezug auf den Einsatz von Vorgehensmodellen zu machen. Der Fragebogen, mit dessen Hilfe die Interviews durchgeführt wurden, findet sich im Anhang in „Fragen zur Erhebung des State of the Art“.
Vorgehensmodelle werden entsprechend der Interviewpartner nicht flächendeckend und oft nur bruchstückhaft eingesetzt. Sehr oft werden selbstentwickelte Entwicklungsprozesse verwendet oder gegebenenfalls nach SPICE [SQI06] oder CMMI [SEI06a] eigene Softwareentwicklungsprozesse definiert. Die Interviepartner haben in der Praxis offenbar überwiegend den Rational Unified Process ® [RUP06], das V-Modell ® XT [VMX06] beziehungsweise [VMXT08], Scrum [Sch02] und Extreme Programming [Bec00] vorgefunden. Dies bestätigt auch nachträglich die Auswahl der Vorgehensmodelle in der vorliegenden Arbeit, die weit vor den Interviews und auf anderem Wege durchgeführt wurde.
Hierzu ist zu erwähnen, daß das V-Modell XT des Bundes für Projekte der öffentlichen Hand ohnehin zwingend vorgeschrieben ist. Bezüglich der sogenannten agilen Ansätze [Agi06] muß betont werden, daß sie durch ihre wenig restriktiv anmutende Art und die geringe Anzahl der zu erstellenden Arbeitsprodukte ein geringeres Integrationsrisiko suggerieren können und weiterhin durch vielerlei entsprechende Publikationen seit Jahren aggressiv propagiert werden. All dies stand ihrer Verbreitung sicher nicht im Wege. Der Rational Unified Process schließlich wird offenbar häufig als Rahmen für das objektorientierte Entwicklungsparadigma, aber auch als Beiwerk der Werkzeugkette der entsprechenden Herstellerfirma eingesetzt.
Wenn Vorgehensmodelle in Unternehmen eingesetzt werden, stellen die umfassenden Ansätze der Rational Unified Process und das V-Modell XT oft den Organisationsstandard dar. Die agilen Ansätze wie Scrum und Extreme Programming werden auch projektweise eingesetzt.
Konservativere Branchen wie die sogenannte Old Economy, also Verteidigung, Finanzdienstleister oder Automotive, sowie Projektbetreiber sicherheitsrelevanter Softwareerzeugnisse setzen tendenziell die etablierten Vorgehensmodelle RUP und V Modell XT ein, die eher progressiv wahrgenommenen Branchen wie beispielsweise die Telekom-

munikationsindustrie eher die agilen Ansätze.
Der Nutzen von Vorgehensmodellen als bewährte Ansätze und Beitrag für bedarfsgerechte Entwicklungsprozesse wird auch von die Interviewpartner im überwiegenden Ausmaß als hoch angesehen (vergleiche auch Kapitel 1.3 beziehungsweise 5.1.2). Sie ist tendenziell der Eigenentwicklung von Softwareprozessen vorzuziehen, die spezialisiertes Personal erfordert, fehleranfällig, teuer, komplex, aufwendig, und daher vergleichsweise ineffizient und ineffektiv ist. Der Nutzen des Einsatzes von Vorgehensmodellen wird von den Interviewpartnern allerdings unter der Voraussetzung gesehen, daß sie unter zu Hilfenahme von spezifischen Know-how bedarfsgerecht und systematisch ausgewählt und angepaßt werden. Gerade dies aber erfolgt in der Praxis offenbar kaum.
Fast ausnahmslos erfolgt die Auswahl eines Vorgehensmodells zufallsgetrieben oder allenfalls ansatzweise systematisch. Diese Herangehensweise kann mit Hinblick auf die Komplexität und Tragweite einer solchen Entscheidung (siehe auch Kapitel 1.3 beziehungsweise 5.1.3) keinesfalls als bedarfsgerecht angesehen werden.
Im Bezug auf den Nutzen von Vorgehensmodellen muß allerdings kritisch angemerkt werden, daß sie eines der klassischen und signifikanten Risiken, nämlich der unbekannten Technologien bezogen auf Entwicklungswerkzeuge, -Plattformen und -paradigmen (siehe auch Kap 2.4.3.1) nicht im hinreichenden Maße adressieren, um Projektbetreiber bei der Handhabung der hieraus entstehenden Projektprobleme adäquat zu unterstützen [Wal08]. Auch eine systematische Analyse kritischer Erfolgsfaktoren für ein Softwareentwicklungsprojekt, so wie sie im Rahmen der vorliegenden Arbeit beschrieben wird, ist häufig nicht beschrieben [Wal08], allerdings bietet das V-Modell XT an diesem Punkt Unterstützung. Ein weiterer kritischer Lösungsschritt in Softwareentwicklungsprojekten sind Softwaredesign und -architektur [Wal08]. Diese werden in den agilen Methoden praktisch per Definition ausgeklammert. Auch von den in der vorliegenden Arbeit berücksichtigten Vorgehensmodellen werden Design und Architektur nur im Rational Unified Process [RUP06] umfassend unterstützt.
Für die inhaltlichen Bezüge innerhalb der vorliegenden Arbeit wurde beim V-Modell XT die Version 1.2 [VMXT06] herangezogen und nur zum Schluß mit der Version 1.2.1.1 [VMXT08] auf Konsistenz überprüft. Für die Bezüge zum Rational Unified Process wurde die Version 2003.06.15 verwendet und zwischenzeitlich geänderte Inhalte am Schluß mit Hilfe von [Ess07] aktualisiert.

1.3 Motivation für ein Eignungseinstufungsverfahren

Vor der Motivation für das Eignungseinstufungsverfahren für Vorgehensmodelle wird im folgenden verdeutlicht, welchen Nutzen Vorgehensmodelle für welche Probleme in Projekten bewirken und welche Unterschiede es hierbei gibt.

Vorgehensmodelle lösen Probleme in Softwareentwicklungsprojekten. In Softwareentwicklungsprojekten sind oft die fachlichen, technischen und organisatorischen Verantwortlichkeiten der Projektmitarbeiter nicht genau genug festgelegt. Es fehlt häufig ein gemeinsames Verständnis darüber, welche Tätigkeiten wann und wie durchzuführen sind und wie ihre Arbeitsprodukte beschaffen sein müssen. Hieraus resultieren reduzierte Möglichkeiten der Projektkontrolle und -steuerung. Es existieren ebenfalls unterschiedliche Auffassungen von der Bedeutung von Begriffen, weil hierfür keine Standards etabliert sind. Dies führt zu Mißverständnissen und Ineffizienz im Projektablauf.
Vorgehensmodelle definieren Rollen, um Verantwortungsbereiche für Personen festzu-

legen. Sie stellen dar, welche Aktivitäten im Projekt wann von wem durchgeführt werden müssen. Vorgehensmodelle beschreiben, welche Arbeitsprodukte im Projekt entstehen müssen und welche Eigenschaften diese aufweisen müssen [Chr92]. Sie legen die Bedeutung von Begriffen fest und tragen so zu einer Verminderung von Mißverständnissen und einer erhöhten Effizienz bei. Dies wirkt sich förderlich für den Projekterfolg aus.
In Softwareentwicklungsprojekten kann das Wissen über Vorgehensmodelle und methodische Softwareentwicklung im Großen nicht als gegeben vorausgesetzt werden.
Vorgehensmodelle beinhalten das Erfahrungs- und Methodenwissen ihrer Autoren über die aus deren Sicht geeignete Durchführung von Softwareentwicklungsprojekten. Sie tragen so zu einer Kompetenzsteigerung im Projekt bei, was die Erfolgschancen von Projekten erhöht.
In vielen Softwareprojekten ergeben sich häufig Änderungen von Zielen und Anforderungen. Unkontrollierte Änderungen hinterlassen unter Umständen widersprüchliche Stände verschiedener Arbeitsprodukte. Vorgehensmodelle beschreiben unterschiedlich flexible Herangehensweisen, um mit großer Veränderungshäufigkeit umzugehen.
[Chr92] betont, daß die Qualität von Softwareprodukten von der Qualität des Entwicklungsprozesses abhängt. Auch [Wal01] unterstreicht die Bedeutung von Vorgehensmodellen für die Qualität der Softwareentwicklung.

Nutzen eines Eignungseinstufungsverfahrens für Vorgehensmodelle. Ein Eignungseinstufungsverfahren unterstützt die systematische Auswahl von Vorgehensmodellen für spezifische Projekte. Da Vorgehensmodelle sehr unterschiedlich sind und daher für jeweils sehr unterschiedliche Projekte geeignet sind, ist eine systematische Auswahl notwendig. Verschiedene Vorgehensmodelle tragen zur Lösung unterschiedlicher Probleme in Projekten bei. Diese Probleme sind beispielsweise aus der Projektgröße resultierende fehlende Überschaubarkeit oder ständigen Veränderungen, die zu mangelnder Planbarkeit führen. Da Projektgröße und Flexibilität nicht beide im gleichen Maße durch Vorgehensmodelle unterstützt werden können, sind für unterschiedliche Problemschwerpunkte unterschiedlich entworfene Vorgehensmodelle geeignet. Dies erfordert ein Verfahren zur Eignungseinstufung von Vorgehensmodellen.

Die umfassende Diskussion der oben genannten Punkte würde hier den Lesefluß beeinträchtigen und findet sich daher im Hintergrund-Kapitel 5.1.

1.4 Vergleich mit inhaltlich verwandten Arbeiten

Das EVGM leistet einen Beitrag zur Weiterentwicklung des bestehenden Standes des Wissens in den Themengebieten der Software Prozeßverbesserung und der Vorgehensmodelle. Es unterstützt die Untersuchung von Projekten, um geeignete Vorgehensmodelle auszuwählen. Um den diesbezüglichen Stand des Wissens zu charakterisieren, ist es sinnvoll, existierende Ansätze zur Untersuchung von Projekten, sowie zum Vergleich und zur Evaluation von Vorgehensmodellen zu untersuchen. Ein Vergleich mit methodisch verwandten Arbeiten erfolgt hingegen in Kapitel 5.2.

Abgrenzung Reifegradmodelle und klassische Software Prozeß Verbesserung. Als methodische Ansätze zur Untersuchung und Bewertung von Projekten sind die Reifegradmodelle ISO 15504 (SPICE) [SQI06], [Tha98], [Hör+06], Automotive SPICE [Mue+07] oder CMM [Hum90] beziehungsweise CMMI ® [SEI06a], [Kne03] und [Chr

+07] verbreitet. Reifegradmodelle legen ihr Hauptaugenmerk eher auf die Existenz und Etabliertheit von Prozessen in einer Organisation. Sie formulieren Anforderungen an Entwicklungsprozesse in Unternehmen und leiten die Untersuchung der Verwendung von sogenannten Best Practices zur Zielerreichung von Softwareentwicklungsprozessen an. Entsprechend der Zielerreichung der Prozesse einer Organisation oder eines Projektes können diese in ein kontextunabhängig definiertes Reifegradschema eingeordnet werden [Chr+07].
Das EVGM hingegen unterstützt eine problemorientierte und projektspezifische Faktorenanalyse. Im Gegensatz zu Reifegradmodellen sind der Ansatzpunkt des EVGM die projektspezifischen Probleme und nicht die bereits eingeführten Prozesse zu deren Lösung. Es untersucht ein Projekt oder eine Organisation daher problemorientierter als Reifegradmodelle. Das EVGM operiert bei der Lösungsfindung mit umfassenden konsistenten Maßnahmenpaketen in Form von Vorgehensmodellen. Es hat insgesamt einen unmittelbarer an konkreten Problemen ausgerichteten, auf wesentliche Problemschwerpunkte fokussierten Ansatz, der durch die Modellierung auch dynamische Aspekte abdeckt. Es ist kein Verbesserungsverfahren wie es beispielsweise IDEAL ^{SM} [SEI06], [McF96] und kein Begutachtungsverfahren wie beispielsweise SCAMPI [SEI06b]. Das EVGM ist auch nicht geeignet, in ein Prozeßverbesserungsverfahren wie beispielsweise IDEAL eingebettet zu werden, da es auf der Problemebene in Projekten ansetzt, während die üblichen SPI-Ansätze mit der Prozeßanalyse, also auf der Lösungsebene anfangen. Die Unterschiede zwischen SPI und EVGM zeigt die Tabelle 1 noch einmal auf einen Blick.

Aspekt	*Reifegradmodelle und SPI*	*EVGM*
Operand	Reifegradmodelle formulieren Anforderungen an Entwicklungsprozesse	VGM sind Vorlage für Entwicklungsprozesse
Ansatzpunkt	Universell, Unternehmens- und projektunabhängig unabhängig definierte Best Practices als Lösungen	projektspezifisch existierende Probleme
Betrachtung dynamischer Facetten	Dynamischen Wechselwirkungen werden nicht explizit berücksichtigt	Dynamische Wechselwirkungen werden explizit berücksichtigt
Scope	Kompletter SPI-Zyklus, beginnend bei Prozeßanalyse und enden bei Überprüfung der Nachhaltigkeit der Prozeßverbesserung	Problemanalyse und Lösungsvorschlag durch VGM, keine Implementierung der Prozesse

Tabelle 1 Unterschiede SPI und EVGM

Frühere Ansätze zum Vergleich von Vorgehensmodellen. [Boe04], [Lar04] und [Fil05] vergleichen jeweils einige konkrete Vorgehensmodelle miteinander. Dies geschieht allerdings nicht mit Hilfe anwendungsbezogener, problemorientierter Kriterien, so daß deren Ergebnisse in der vorliegenden Arbeit nur bedingt weiterverwendbar sind.
[Dal+05] vergleichen Vorgehensmodelle mit Hilfe der Kategorien („Sequential“, „Incremental“, „Evolutionary“ und „Agile Approaches“). Der schlußendliche Schritt, basierend hierauf ein konkretes Vorgehensmodell auszuwählen, bleibt dem Projektbetreiber überlassen. Da hierfür keine eindeutigen Entsprechungen und Zuordnungen von Kategorien zu konkreten Vorgehensmodellen beschrieben sind, kann dieser Schritt nicht ohne spezielle Kenntnisse ausgeführt werden, was in der vorliegenden Arbeit mit Hinblick auf die in 1.1 genannten Zielgruppen der Arbeit nicht als bedarfsgerecht angesehen wird.
[Oca+05] beschreiben den Vergleich von Vorgehensmodellen auf Metamodellebene. Der Ansatz ist generisch und erwähnt konkrete Vergleichskriterien nur exemplarisch.

Auch hier ist kein Anwendungsbezug und keine Verbindung zu konkreten Vorgehensmodellen beschrieben.
[Ste97] vergleicht objektorientierte Analysemethoden und bewertet ihre Eignung für „Datenlastige“, „Algorithmenlastige“, “Echtzeitfähige“ und „Ablauforientierte Systeme“. Diese Kategorisierung von Anwendungsgebieten ist einerseits nicht überschneidungsfrei und andererseits nicht umfassend und differenziert genug, um unter den in der vorliegenden Arbeit angenommenen Rahmenbedingungen bedarfsgerecht zu sein. Sie richtet sich auch nicht nach den empirisch belegbaren Problemschwerpunkten aus und enthält keine Betrachtung von Wechselwirkungen.
[Hul+02] vergleichen auf der Basis von Best Practices einige Vorgehensmodelle. Sie behandeln aber keine Projektmerkmale und haben damit keinen unmittelbaren Problembezug.

Frühere Ansätze zur Eignungseinstufung von Vorgehensmodellen. [Per+96] beschreiben einen Ansatz, der ein Rahmenwerk mit einer generischen Vorgehensweise bietet. Es werden einige „Project Variables“ und „Context Attributes“ vorgeschlagen, aber nicht hergeleitet und systematisiert. Die Existenz von Beziehungen und Interaktionen zwischen diesen wird angesprochen, aber nicht beschrieben. Die Ausführung des Ansatzes wird nicht konkret beschrieben, daher ist er nicht direkt anwendbar. Unter den für die vorliegenden Arbeit angenommenen Rahmenbedingungen ist der Ansatz von [Per+96] nicht bedarfsgerecht, weil er durch die Zielgruppe nicht ausgeführt werden kann.
[Ale91] beschreibt systematisierte, aber nicht hergeleitete „Project Criteria“. Beziehungen zwischen diesen werden nicht dargestellt. Auch hier wird nicht mit konkreten Vorgehensmodellen, sondern Vorgehensmodell Kategorien („Level 1-3 Processes“ gestaffelt nach Detaillierungsgrad der Beschreibungen, beziehungsweise „Conventional“, „Incremental“, „Evolutionary“, „Waterfall“, „Hybrid Prototyping“, „Operational Specification Software Process Model“) gearbeitet. Die Zuordnung der „Project Criteria“ zu Vorgehensmodell-Maßnahmen wird tabellarisch beschrieben und binär bewertet, aber nicht hergeleitet und begründet. Der eigentliche Auswahlschritt von Vorgehensmodellkategorie zum konkreten Vorgehensmodell wird auch hier dem Projektbetreiber überlassen.
[Boe90] diskutiert eine unsystematisierte, nicht hergeleitete Liste von isoliert betrachteten Projektkriterien. Er führt eine Zuordnung zu Vorgehensmodellkategorien durch, die er aber nicht begründet.
[Kar93] trennt nicht Vorgehensmodell- und Projektkriterien. Die Kriterien sind nicht hergeleitet und werden isoliert voneinander betrachtet. Sie sind nicht auf empirisch bekannte Probleme ausgerichtet und ihre Bewertungen und Zuordnungen werden nicht begründet.
[Sch02a] leitet in Zusammenhang mit dem Forsoft I Projekt [For02] ein Verfahren zur Bewertung von Vorgehensmodellen her. Er geht aber nicht auf die Wechselwirkungen zwischen den Einflußgrößen von Softwareprojekten ein. Sein Ansatz mündet in der Zuordnung von Vorgehensmodellkategorien. Die Auswahl von existierenden Vorgehensmodellen wird nicht durchgeführt.

1.5 Überblick über den Ablauf des EVGM

Motivation des EVGM durch einführendes Beispiel. Vor der Darstellung des Ablaufes des EVGM wird sein Nutzen durch ein Beispiel verdeutlicht. Nachfolgend wird beispielhaft ein erdachtes Projekt skizziert. Es werden mit einem nicht methodisch

durchgeführten Ansatz fiktive Einflußfaktoren analysiert. Daraufhin wird verdeutlicht, daß ein unsystematischer Ad-hoc-Ansatz der Komplexität der Problemstellung nicht gerecht würde. Somit wird gezeigt, daß zur Eignungseinstufung von Vorgehensmodellen ein umfassender methodischer Ansatz sinnvoll ist.

Im gegebenen Projekt seien durch eine Ad hoc Analyse zusammen mit dem Projektbetreiber die folgenden Sachverhalte identifizierbar:

Es handelt sich um eine In-house Entwicklung eines Softwareproduktes, die komplett durch ein kompetentes, kleines Team durchgeführt werden soll. Als Stakeholder fungiert ein dezidiert verantwortlicher Produktmanager. Seine Verfügbarkeit für fachliche Rückfragen der Entwickler ist zugesagt. Das Projekt steht unter einem hohen Zeit- und Budgetdruck, da die Konkurrenz mit vergleichbaren Produkten ebenfalls eine Markteinführung anstrebt. Das Produkt hat einen hohen Innovationsgrad, wodurch dem verantwortlichen Produktmanager viele Anforderungen noch nicht klar sind. Die Anforderungen können sich daher auch fortwährend ändern.

Sogar wenn beim Projektbetreiber als bekannt vorausgesetzt würde, daß Ansätze aus dem Bereich der Agilen Methoden wie beispielsweise Scrum [Sch02] oder Extreme Programming [Bec00] dafür vorgesehen sind, in Projekten mit instabilen Anforderungen eingesetzt zu werden, weist ein rein intuitiver Zugang zur Auswahl eines Vorgehensmodells einige Defizite auf:

1. Eine systematische Untersuchung von Merkmalen und deren Wechselwirkungen ist mit einem intuitiven Ad hoc Ansatz nicht gegeben, dadurch ist nicht überprüfbar ob die Problemanalyse komplett ist.
2. Eine systematische Betrachtung von Vorgehensmodellen mit ihren Stärken und Schwächen ist mit einem intuitiven Ansatz nicht gegeben, dadurch kann die Entscheidung für ein Vorgehensmodell nicht optimiert werden.
3. Eine explizite Zuordnung von Problemklassen und Lösungen durch Vorgehensmodelle ist nicht gegeben, dadurch ist die Lösungsfindung und Entscheidung für ein Vorgehensmodell für Dritte nicht nachvollziehbar. Auch ein Lerneffekt wird dadurch behindert.
4. Eine methodische Schritt-für-Schritt-Vorgehensweise ist durch einen intuitiven Ansatz nicht gegeben, dadurch geht der Überblick über projektförderlich und projektbehindernd zu bewertende Einflußfaktoren, sowie deren Wechselwirkungen verloren. Der jeweils entstehende Gesamtzustand durch den Einsatz eines Vorgehensmodells ist intuitiv nicht mehr überschaubar und somit nicht vergleichbar. Dies behindert das Erzielen einer optimalen Lösung durch die Auswahl des geeignetsten Vorgehensmodells.

Aus diesen Punkten kann gefolgert werden, daß der Vergleich und die Auswahl eines Vorgehensmodells für ein Projekt eines systematischeren Lösungsansatzes bedarf, so wie ihn das EVGM bietet.

Ablauf des EVGM. Im EVGM werden Projekte und Vorgehensmodelle anhand ihrer wesentlichen Einflußfaktoren, dargestellt als Merkmale, beschrieben. Diese Darstellung wird im EVGM durch die Wechselwirkungen zwischen den Merkmalen detailliert. Die Merkmale von Projekten und Vorgehensmodellen werden in Ursache-Wirkungs-Diagrammen (siehe Kapitel 4.4 „Ursache-Wirkungs-Diagramme als Notation des EVGM“)

modelliert. Hierbei wird bewertet, ob der projektspezifische Zustand der Merkmale förderlich oder hinderlich für den Projekterfolg ist. Die Wechselwirkungen zwischen den Merkmalen werden mit Hilfe von Beeinflussungsbeziehungen ausgedrückt.
Der Ablauf des EVGM wird zunächst allgemein dargestellt (Abbildung 1, UML Aktivitätsdiagramme vgl. [Jec+04]). Ein Beispiel mit reduziertem Umfang veranschaulicht den prinzipiellen Ablauf der Modellierung im EVGM (Abbildung 2 bis Abbildung 5).

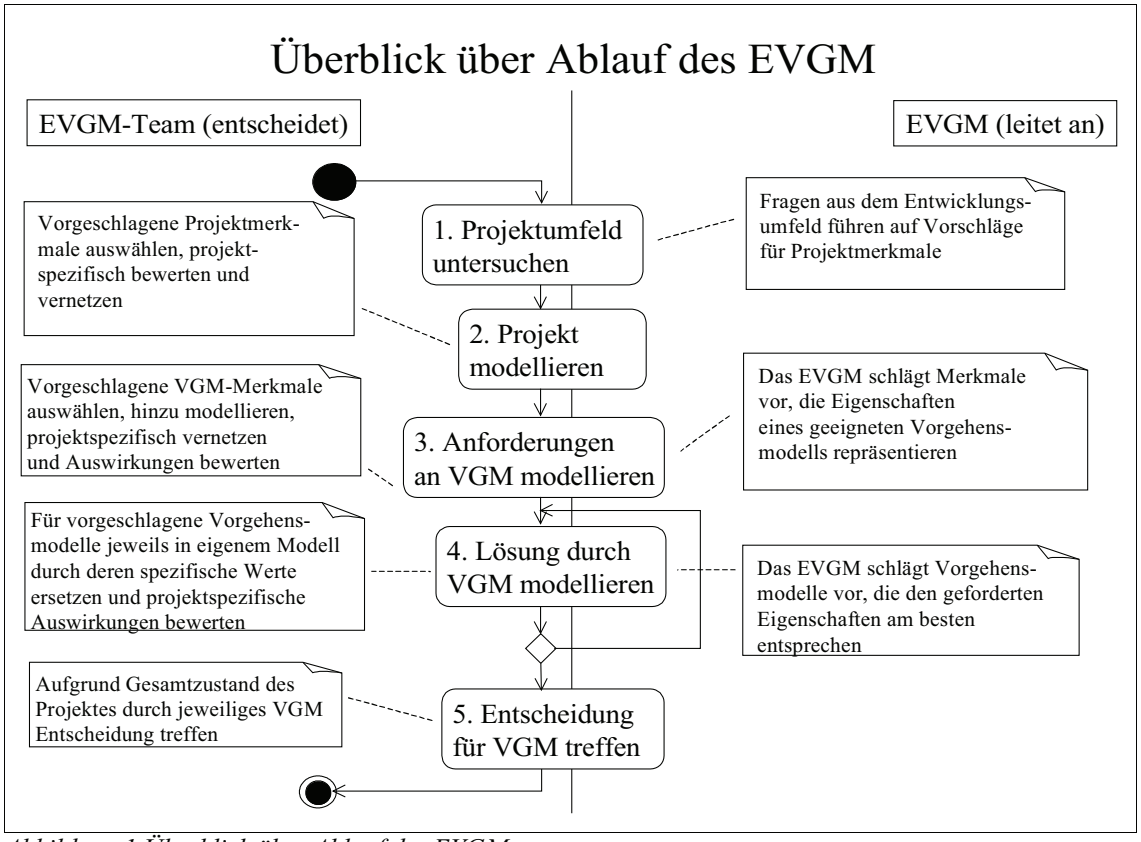

Abbildung 1 Überblick über Ablauf des EVGM

1. Im ersten Schritt des Verfahrens werden vom EVGM Projektmerkmale vorgeschlagen (Abbildung 1, Aktivität 1 „Projektumfeld untersuchen“).
2. Die für das anstehenden Projekt wesentlichen Merkmale werden von den EVGM-Benutzern ausgewählt und hinsichtlich ihrer Zustände bewertet (Abbildung 1, Aktivität 2 „Projekt modellieren“). Sie werden in einem Ursache-Wirkungs-Diagramm modelliert. Vom EVGM beschriebene Beeinflussungsbeziehungen zwischen ihnen werden hinzugefügt, wie in Abbildung 2 unten dargestellt. Abbildung 2 zeigt ein Ursache-Wirkungs-Diagramm mit reduziertem Umfang. Es enthält die Merkmale „Stabilität und Transparenz der fachlichen Anforderungen“ (*EPD-StaTrans-FachlicheAnforderungen*, Herleitung siehe Anhang, „Ontologie zur Herleitung der Merkmale“*),* sowie „Bedarfsdeckung Projektbudget“ (*EPP-BedDeck-Kap-Projektbudget)*. Die Zustände von beiden wurden vom EVGM-Team als hinderlich für den Erfolg des anstehenden Projektes bewertet („-“). Die Merkmale sind durch eine Beeinflussungsbeziehung verbunden, die besagt daß eine bessere Stabilität und Transparenz der fachlichen Anforderungen ein kosteneffizienteres Arbeiten ermöglicht und damit die Bedarfsdeckung des Projektbudgets positiv beeinflusst („≈“). Diese Beeinflussungsbeziehung wird vom EVGM mit „≈≈“ vorgeschlagen, hier aber vom Team spezifisch angepasst.

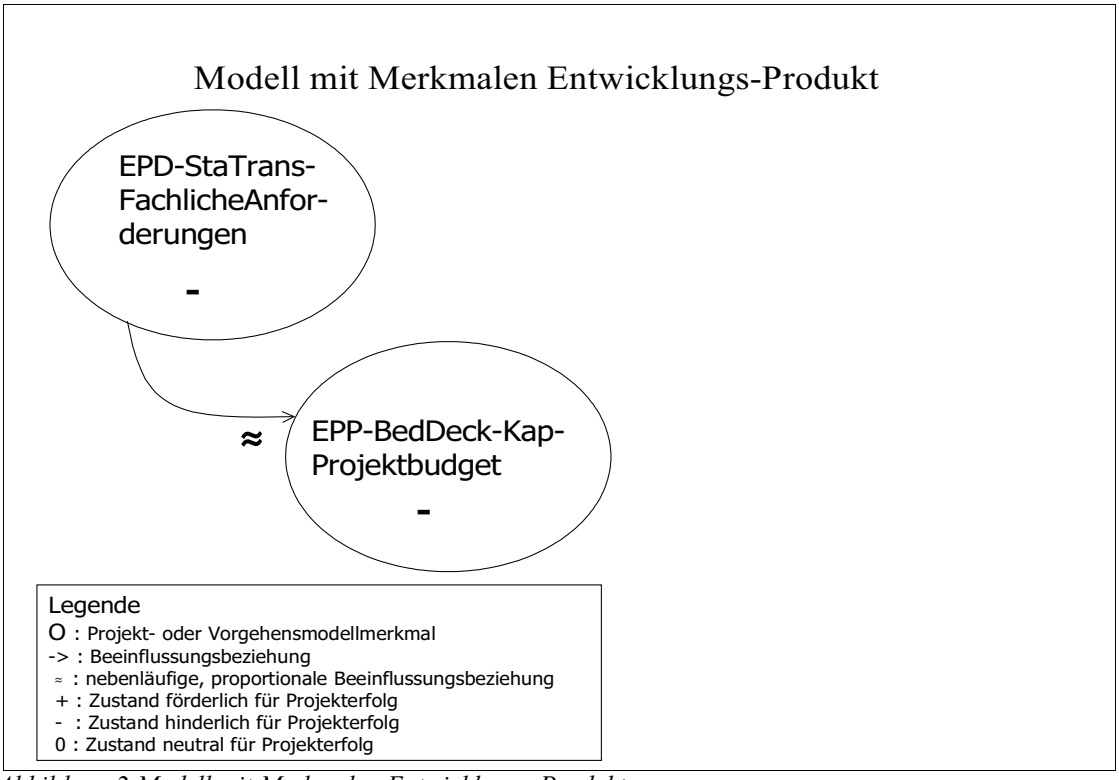

Abbildung 2 Modell mit Merkmalen Entwicklungs-Produkt

3. Das EVGM leitet für die als projektgefährdend bewerteten Projektmerkmale Anforderungen an die Eigenschaften ab, die ein für dieses Projekt ideales Vorgehensmodell aufweisen muß (Abbildung 1, Aktivität 3 „Anforderungen an VGM modellieren"). Diese Anforderungen an Vorgehensmodelle werden von den Benutzern des EVGM dem Ursache-Wirkungs-Diagramm aus Abbildung 2 hinzugefügt und durch Beeinflussungsbeziehungen mit den Merkmalen des Projektes verbunden. Diese Beeinflussungsbeziehungen werden vom EVGM vorgegeben. Durch die Beeinflussung der Prozeßanforderungsmerkmale verbessern sich die Zustände der problematischen Projektmerkmale, wie die Abbildung 3 unten verdeutlicht. Hier wird starke („++") „Unterstützung des Requirements Engineering" (*EV-UnterstProjTheBer-Requirements-Engineering)* gefordert, welche die „Stabilität und Transparenz der fachlichen Anforderungen" (*EPD-StaTrans-FachlicheAnforderungen)* positiv („**≈**") beeinflusst. Das EVGM-Team bewertet daraufhin den Zustand der „Stabilität und Transparenz der fachlichen Anforderungen" (*EPD-StaTrans-FachlicheAnforderungen)* als neutral („**0**"). Es entscheidet, daß sich diese Zustandsverbesserung aufgrund der positiven Beeinflussung bis zur „Bedarfsdeckung des Projektbudgets" (*EPP-BedDeck-Kap-Projektbudget)* auswirkt, die daraufhin ebenfalls mit neutral („**0**") eingeschätzt wird.

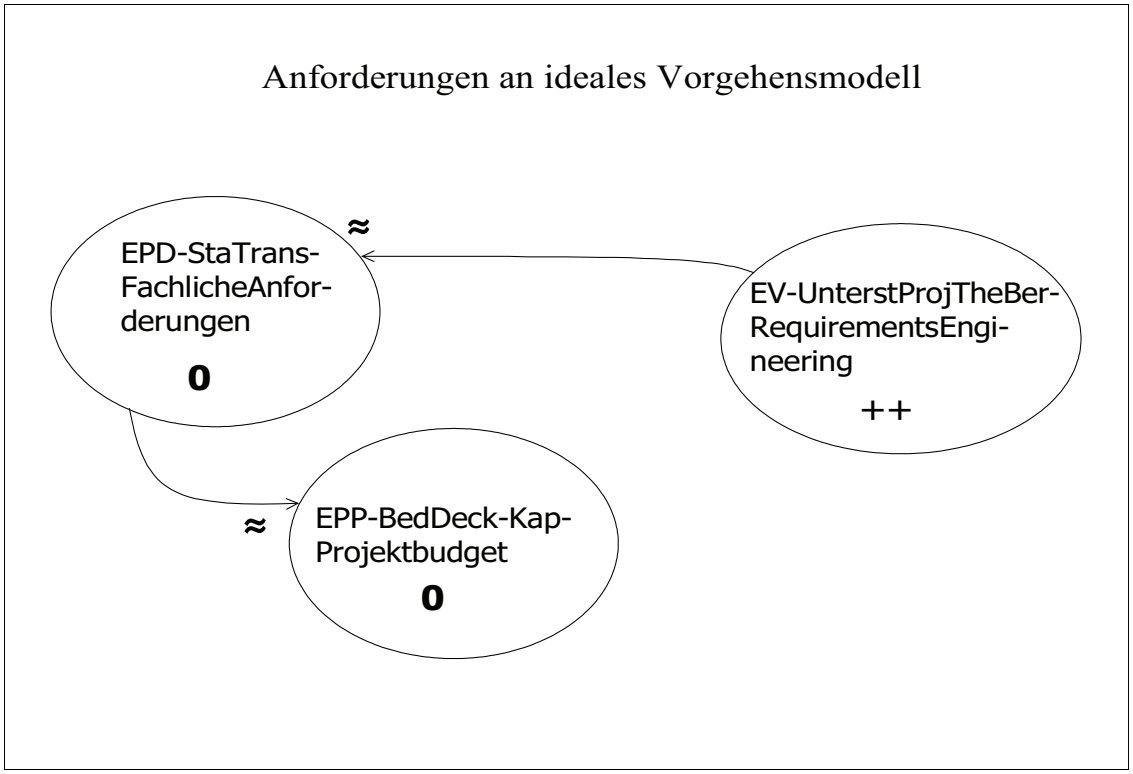

Abbildung 3 Anforderungen an Vorgehensmodell

4. Das EVGM berücksichtigt exemplarisch die vier konkreten Vorgehensmodelle Extreme Programming (XP), Rational Unified Process (RUP), Scrum und V-Modell XT, die jeweils unterschiedliche thematische Ausrichtungen besitzen. Die Bewertung der Merkmale dieser Vorgehensmodelle sind durch das EVGM vorgegeben. Mit Hilfe der Anforderungen an ein adäquates Vorgehensmodell wird geprüft, welches der zur Auswahl stehenden Vorgehensmodelle für das anstehende Projekt geeignet erscheint. Jeweils pro geeignetem Vorgehensmodell wird ein eigenes Ursache-Wirkungs-Diagramm modelliert (Abbildung 1, Aktivität 4 „Lösung durch VGM modellieren“, in der Abbildung 4 für XP und Abbildung 5 für RUP), in dem die Vorgehensmodell-Anforderungen durch die Merkmale des jeweiligen Vorgehensmodells ersetzt und modelliert werden. Das Projektmerkmal „Bedarfsdeckung des Projektbudgets“ (*EPP-BedDeck-Kap-Projektbudget)* verbleibt nach Entscheidung der Benutzer beim Einsatz von XP auf „**-**“, während es beim Einsatz von RUP auf „**0**“ kommt.

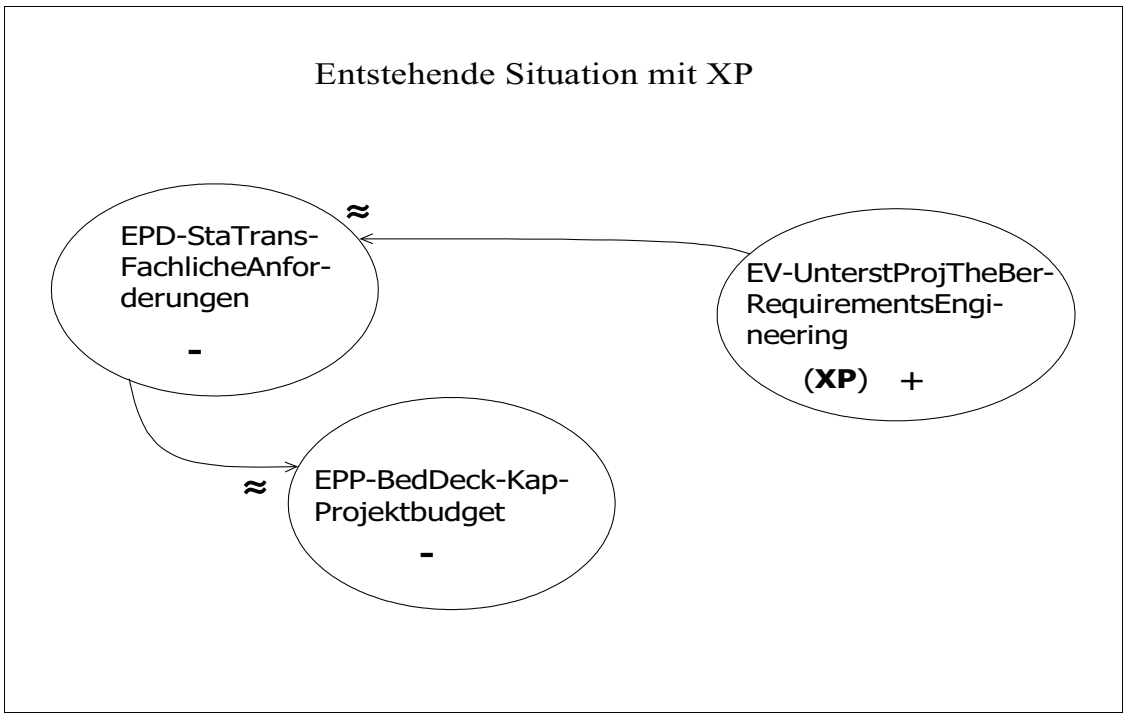

Abbildung 4 Entstehende Situation mit XP

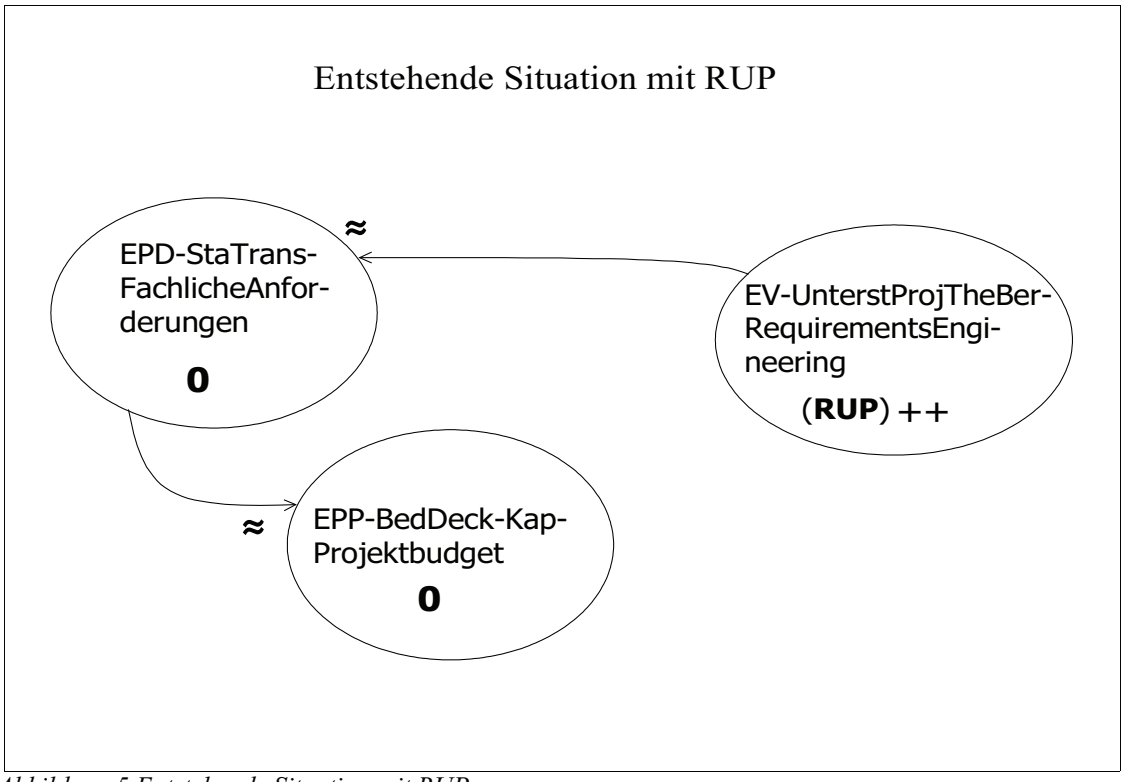

Abbildung 5 Entstehende Situation mit RUP

5. Anhand der verbesserten Zustände der ursprünglich erfolgsgefährdend bewerteten Projektmerkmale, die in den jeweils vorgehensmodellspezifischen Ursache-Wir-

kungs-Diagrammen entstehen, kann die Entscheidung für ein Vorgehensmodell getroffen werden (Abbildung 1, Aktivität 5 „Entscheidung für VGM treffen“). Im Beispiel oben würde die Wahl auf den Rational Unified Process (Abbildung 5) fallen.

1.6 Struktur der Arbeit

Das Eignungseinstufungsverfahren für Vorgehensmodelle (EVGM) unterstützt die Entscheidung für ein Vorgehensmodell für ein konkretes Projekt. Dies geschieht anhand einer Modellierungsmethode, welche die signifikanten Probleme im Projekt in Zusammenhang mit möglichen, durch Vorgehensmodelle angebotenen Lösungen darstellt und modelliert.

Kapitelinhalte und -zusammenhänge. Im Kapitel 2 „Charakterisierung von Projekten“ werden die Merkmale hergeleitet, systematisiert und definiert, die ein Projekt charakterisieren. Aus den Projektmerkmalen werden bei der Anwendung des EVGM die projektspezifisch wichtigsten ausgewählt und für die Modellierung des spezifischen Projektes verwendet. Dies wird in Kapitel 4 „Das Modellierungsverfahren im EVGM“ beschrieben. Für alle Merkmale werden ihre Wechselwirkungen mit anderen Merkmalen dargestellt.
Im Kapitel 3 „Charakterisierung von Vorgehensmodellen“ werden die Merkmale hergeleitet, systematisiert und definiert, welche die allgemeine und konkrete Wirkweise Vorgehensmodellen in Projekten beschreiben. Aus diesen Merkmalen werden bei der Anwendung des EVGM die projektspezifisch geeignetsten ausgewählt und für die Modellierung der spezifischen Lösung oder Verbesserung durch ein Vorgehensmodell verwendet. Dies wird in Kapitel 4 „Das Modellierungsverfahren im EVGM“ beschrieben. Die im EVGM berücksichtigten Vorgehensmodelle werden anhand der Vorgehensmodellmerkmale bewertet.
Das Kapitel 4 beschreibt die Modellierungsmethode des EVGM. Mit dieser Modellierungmethode werden die Merkmale aus Kapitel 2 und 3 ausgewählt und modelliert, um für das spezifische Projekt eine möglichst geeignete Lösung durch ein Vorgehensmodell zu finden. Das EVGM wird am Schluß des Kapitels 4 anhand einer Fallstudie für ein fiktives Projekt veranschaulicht.
Tiefergehende Fundierungen der in den Kapiteln 2, 3 und 4 getroffenen Aussagen finden sich in Kapitel 5 und im Anhang. Diese würden den Lesefluß, die Verständlichkeit und die Übersichtlichkeit der Kapitel im Hauptteil vermindern und werden darum im Anschluß an diese dargestellt. Im Kapitel 6 „Zusammenfassung, kritische Betrachtung, Ausblick“ werden die wesentlichen Erkenntnisse und Aussagen der Arbeit zusammengefaßt und weiterführende Perspektiven der Thematik und des EVGM aufgezeigt.

Leseempfehlung. Um den Ablauf des EVGM schnell kennen zu lernen empfiehlt es sich also, mit Kapitel 4 zu beginnen. Zuvor können im Bedarfsfall die Zusammenfassungen am Beginn von Kapitel 2 „Charakterisierung von Projekten“ und 3 „Charakterisierung von Vorgehensmodellen“ gelesen werden, um einen Überblick über die Projekt- und Vorgehensmodellmerkmale zu bekommen, die das in Kapitel 4 „Das Modellierungsverfahren im EVGM“ dargestellte Verfahren benutzt.
In den Kapiteln 2 und 3 kann nachgeschlagen werden, wenn die detaillierte Herleitung, Definition und Motivation einzelner Projekt- oder Vorgehensmodellmerkmale von Interesse ist. Das Kapitel 5 „Hintergrund und vertiefende Fundierung des EVGM“

schliesslich ist nur für den Leser interessant, für den die Grundlagen der vorliegenden Arbeit wichtig sind. Die Themenübersicht der Hauptkapitel wird nachfolgend in Abbildung 6 dargestellt.
Weitere Herleitungen, die für das Verständnis der Grundlagen für die vorliegende Arbeit nützliche sind, finden sich im Anhang.

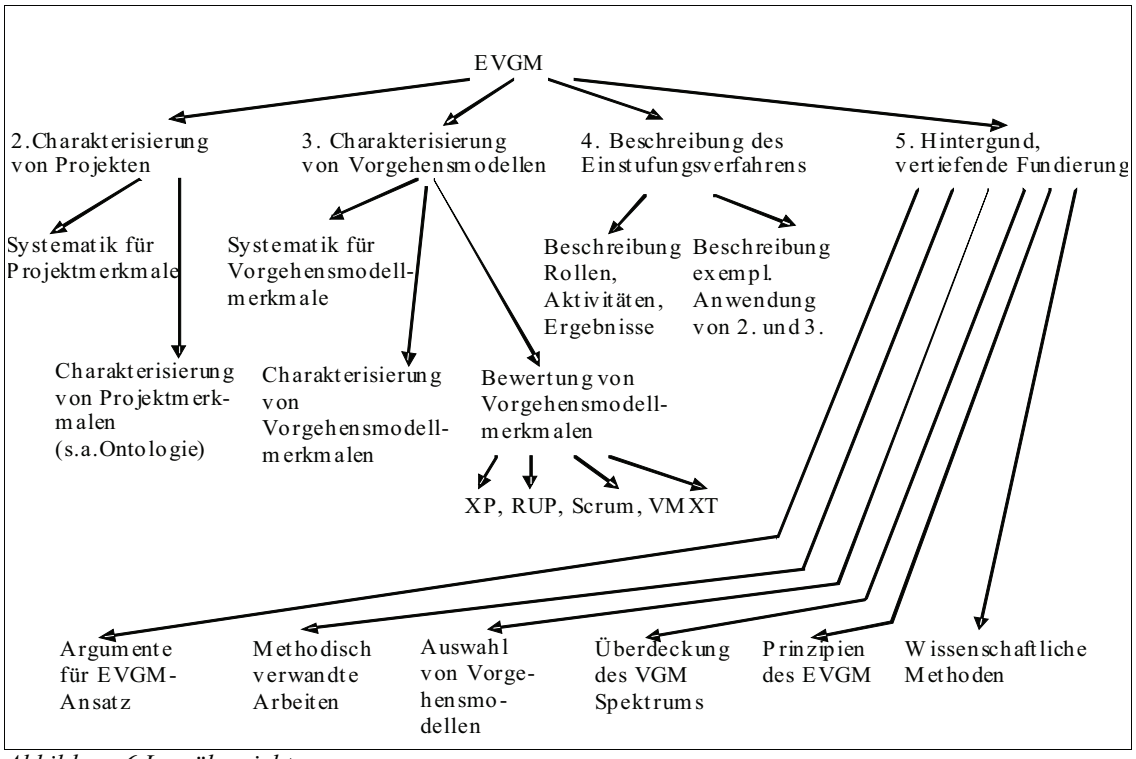

Abbildung 6 Leseübersicht

2 Charakterisierung von Projekten

In diesem Kapitel werden die Projektmerkmale erarbeitet, die in der Modellierungsmethode in Kapitel 4 zur Darstellung der projektspezifisch wesentlichen Einflußgrößen verwendet werden.
Das Kapitel beginnt mit einer Zusammenfassung seines Inhalts. Diese Zusammenfassung gibt dem Leser einen Überblick über die Projektmerkmale. Die ontologischen Bezeichner sind hierbei in Klammern angegeben. Die Zusammenfassung ersetzt jedoch nicht die detaillierten Herleitungen und Begründungen in den nachfolgenden Abschnitten. Jedes Projektmerkmal wird dort anhand von Quellen belegt, hinsichtlich ihrer Relevanz motiviert, sowie definiert. Die Projektmerkmale sind durch eine Ontologie hergeleitet und in eine Systematik eingeordnet. Diese verdeutlicht, welche Aspekte von Projekten abgedeckt werden. Die umfassende Herleitung der Systematik, sowie der Ontologie kann im Anhang, Abschnitt „Die Systematik“ nachgeschlagen werden.

Zusammenfassung. Für die Charakterisierung von Projekten beschreibt das EVGM die folgenden Merkmale (Bezeichner sind *kursiv* und in Klammern dargestellt, die Kürzel entstehen durch die ontologische Herleitung, welche im Anhang, Abschnitt „Die Systematik“ verdeutlicht wird).
Das Merkmal Einfachheit durch Umfang Funktionen (*EPD-Einf-DurchUmfangFunktionen)* beschreibt die anwachsende Schwierigkeit des Projektes durch eine steigende Anzahl fachlicher Funktionen und deren wechselseitiger Abhängigkeiten. Es steht in Zusammenhang mit der Stabilität und Transparenz fachlicher Anforderungen (*EPD-StaTrans-FachlicheAnforderungen).* Durch sie wird die Beständigkeit und Erkennbarkeit der geforderten Eigenschaften der Software beschrieben, die aus der Problemdomäne des Projektes resultieren. Diese beiden Merkmale decken die fachlichen Aspekte ab und werden auf der technischen Seite durch die beiden nächsten Merkmale ergänzt. Die Einfachheit durch den Umfang technischer Anforderungen und Qualitätskriterien (*EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien)* stellt die steigende Schwierigkeit des Projektes dar, welche ein Anwachsen der technischen Restriktionen und Qualitätsanforderungen mit sich bringt. Je mehr technische Restriktionen bei der Lösung des fachlichen Problems berücksichtigt werden müssen, desto höher steigt die Kompliziertheit des Codes. Die Stabilität und Transparenz technischer Anforderungen und Qualitätskriterien (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien)* repräsentiert hingegen nicht deren Umfang, sondern deren Beständigkeit bezüglich unerwünschter Änderungen, sowie deren vollständige Erkennbarkeit. Technische Anforderungen und Qualitätskriterien haben oft Querschnittscharakter innerhalb eines Softwaresystems. Deshalb wirken sich Änderungen an ihnen durch besonders weitreichende Seiteneffekte aus. Zu den Merkmalen des Entwicklungsproduktes gehören neben seinen fachlichen und technischen Eigenschaften auch für seine Entwicklung benötigte Ressourcen. Über die bereits angeführten, durch Personen repräsentierten Ressourcen hinaus sind daher auch personenunabhängige Aspekte zu betrachten. Die Bedarfsdeckung Kalendarische Zeit (*EPP-BedDeck-Kap-KalendarischeZeit)* beschreibt das Verhältnis zwischen der für das Projekt benötigten und der verfügbaren Zeit. Sie wird beispielsweise durch einen unveränderlichen Liefertermin verschlechtert. Die Bedarfsdeckung Projektbudget (*EPP-BedDeck-Kap-Projektbudget)* stellt dagegen das Verhältnis zwischen erforderlichen und vorhandenen finanziellen Mitteln für das Projekt dar. Dem Projektbudget sind bei einem Festpreisprojekt feste Grenzen gesetzt.

Ergänzend zu den eben dargestellten Merkmalen, welche die wesentlichen Aspekte des Entwicklungs-Produktes herausgreifenden, werden die für den Projekterfolg bedeutsamen Merkmale des Entwicklungs-Teams dargestellt. Die Bedarfsdeckung fachlicher Fähigkeiten (*EPD-BedDeck-Fähigk-Fachlich)* repräsentiert das Verhältnis von im Projekt erforderlichen und verfügbaren Kenntnissen über die fachliche Domäne, in der das Projekt angesiedelt ist. Diese Kenntnisse sind erforderlich, um beispielsweise die fachlichen Anforderungen an das Projektergebnis zu erheben. Im Vergleich dazu steht die Bedarfsdeckung Software Engineering Fähigkeiten (*EPSWE-BedDeck-Fähigk-SoftwareEngineering)* für das Verhältnis zwischen dem erforderlichen und vorhandenen Wissen über Methoden der Softwareentwicklung für das Projekt. Es wird benötigt, um die technischen Arbeiten im Projekt durchführen zu können. Das Verhältnis von im Projekt benötigten und tatsächlich gegebenen Kenntnissen zum Einsatz der technischen Methoden und zur Verwaltung von deren Arbeitsprodukten beschreibt hingegen die Bedarfsdeckung Software Management Fähigkeiten (*EPSWE-Deck-Fähigk-SoftwareManagement)*. Durch sie wird eine Abwicklung der technischen Arbeiten des Projektes im Rahmen einer geordneten Vorgehensweise möglich. Durch die Bedarfsdeckung Fähigkeiten kaufmännisches und organisatorisches Projektmanagement (*EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement)* wird das Verhältnis zwischen gefordertem und vorhandenem Können, um die wirtschaftlichen und administrativen Facetten des Projektes abzudecken, repräsentiert. Dieses dient dazu, ein Projekt zu planen, zu kalkulieren, zu organisieren und zu kontrollieren. Die mit wachsender Anzahl der Projektbeteiligten steigende Anforderung an deren Koordination, sowie die durch sie sinkende Produktivität wird mit Einfachheit durch Teamgröße (*EPP-Einf-DurchTeamgröße)* abgebildet. Sie ist nicht zu verwechseln mit der Bedarfsdeckung personelle Kapazitäten (*EPP-BedDeck-Kap-Personell).* Diese steht für das Verhältnis von nötigen und verfügbaren Personen und Personentagen für das Projekt. Während bei der *EPP-Einf-DurchTeamgröße* Organisations- und Effizienzaspekte im Vordergrund stehen, fokussiert die *EPP-BedDeck-Kap-Personell* Aspekte der Ressourcenkalkulation.
Neben rationalen Aspekten werden auch weiche Faktoren betrachtet, deren Berücksichtigung durch die Menschen in Softwareprojekten geboten ist. Eine davon ist die Bedarfsdeckung Motivation im Team (*EPM-BedDeck-MotivationImTeam).* Sie stellt das Verhältnis zwischen gebotenem und gegebenem Willen des Teams, das Projekt zu Erfolg zu führen und die Projektziele über Individualziele zu stellen. Ausreichend motiviert stellen die Projektmitarbeiter ihr gesamtes Können und Wissen uneingeschränkt in den Dienst des Projektes. Die Bedarfsdeckung Kommunikation im Team (*EPM-BedDeck-KommunikationImTeam)* repräsentiert das Verhältnis zwischen benötigtem und erfolgtem Austausch von Informationen im Projekt. In Projekten existiert beziehungsweise entsteht unterschiedliches individuelles Können und Wissen. Daher ist es erforderlich, diese jeweils da zugänglich zu machen, wo sie aktuell benötigt werden. Hierfür ist neben der *EPM-BedDeck-KommunikationImTeam* auch die Bedarfsdeckung Führungsfähigkeiten (*EPM-BedDeck-Führungsfähigkeiten)* dienlich. Sie stellt das Verhältnis von benötigtem und vorhandenem Können zur Betreuung der Projektmitarbeiter in menschlicher Hinsicht dar, wodurch die kooperative Zusammenarbeit der Spezialisten verschiedener Projektthemenbereiche unterstützt wird.

Detailliertere Ausführungen zu den eben benannten Projektmerkmalen finden sich in den nachfolgenden Abschnitten.
Nachfolgende Abbildung 7 stellt Projektmerkmale noch einmal im Überblick dar. Die

Kategorisierung nach „Bedarfsdeckung“, Stabilität und Transparenz“, sowie „Einfachheit ist dabei bereits ein Vorgriff auf Kapitel 2.1.2 „Definieren von Merkmalen“

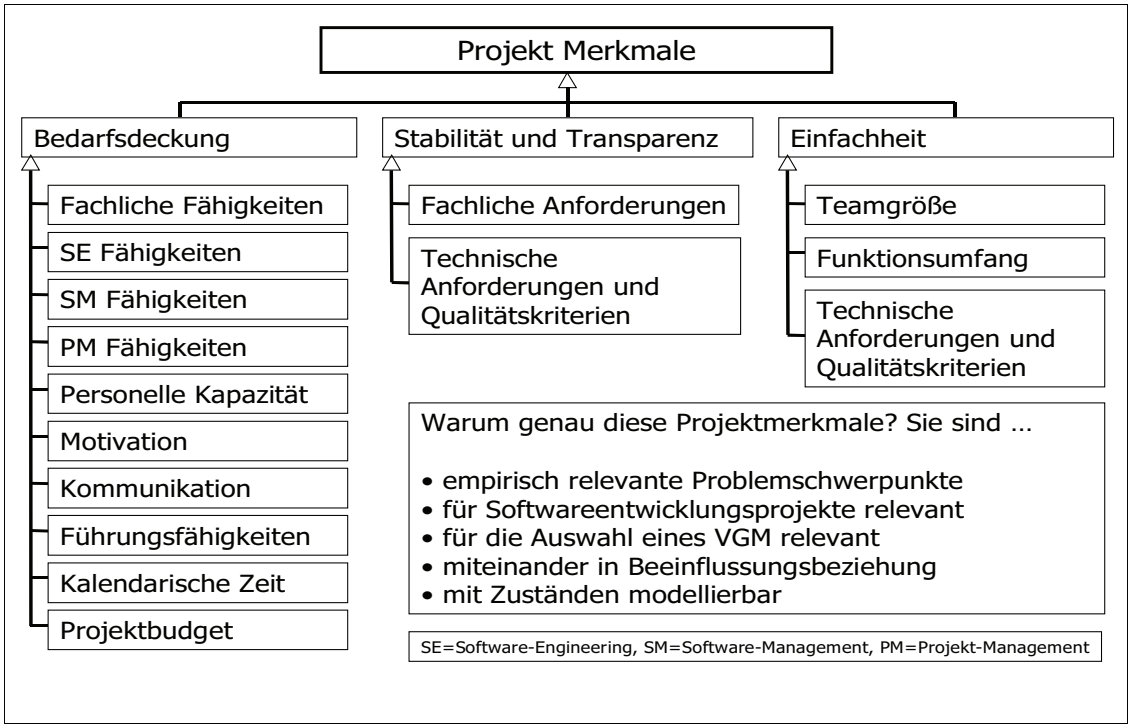

Abbildung 7 Projektmerkmale

Inhalt des Kapitels

2.1 Vorbemerkungen

2.1.1 Die Systematik für Projektmerkmale

Die im folgenden vorgeschlagene Systematik zur Beschreibung von Projekten besteht aus den beiden Dimensionen “Risikobereiche“ und „Projektthemenbereiche“. Innerhalb dieser Dimensionen erfolgt eine Unterteilung nach Bereichen. Die Dimensionen und Bereiche zur Beschreibung von Projekten werden nachfolgend in Abbildung 8 überblicksartig dargestellt und dann beschrieben.

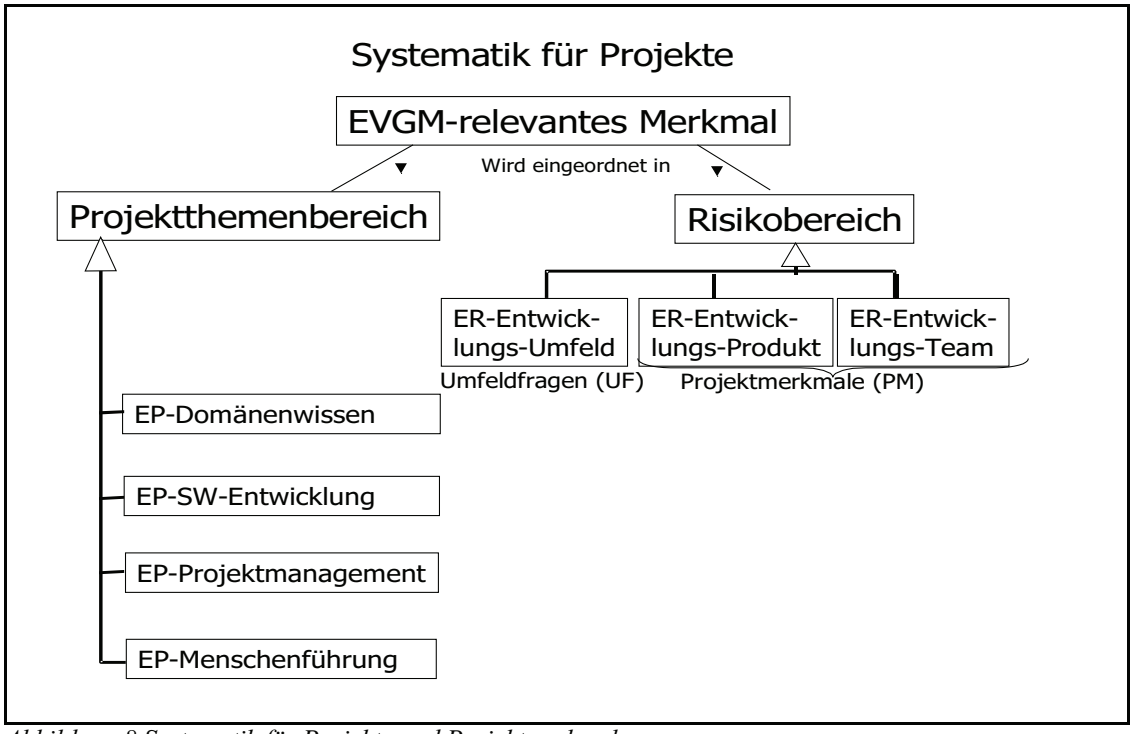

Abbildung 8 Systematik für Projekte und Projektmerkmale

2.1.1.1 Die Risikobereiche

Entwicklungs-Team umfasst für den Projekterfolg wesentliche Eigenschaften der Projektbeteiligten und des Projektteams.
Entwicklungs-Produkt umfasst die für den Projekterfolg wesentlichen Eigenschaften und Aspekte des Projektergebnisses.
Die Risikobereiche Entwicklungs-Team und Entwicklungs-Produkt dienen der systematischen Einordnung von Merkmalen, die wesentlich für den Zustand eines Projektes sind.
Entwicklungs-Umfeld umfasst die für den Projekterfolg wesentlichen Eigenschaften, die nicht unmittelbar aus dem Software Projekt resultieren. Es wird verwendet, um Fragen zu Einflußfaktoren aus dem Projektumfeld systematisch einzuordnen, die Benutzern des EVGM helfen, Projektmerkmale auszuwählen und mit Zuständen zu versehen.

Der vierte Risikobereich heißt **Entwicklungs-Prozeß**, der die für den Projekterfolg wesentlichen Eigenschaften von Vorgehensmodellen umfasst. Dieser Teil der Systematik wird erst in Kapitel 3 „Charakterisierung von Vorgehensmodellen“ umfassend beschrieben. Der Risikobereich Entwicklungs-Prozeß adressiert alle anderen Risikobereiche, weil für jedes dargestellte Problem ein Lösungsansatz durch Vorgehensmodelle angeboten werden soll.
Die Risikobereiche werden in der Ontologie (siehe Anhang, Abschnitt „Ontologie zur Herleitung der Merkmale“) mit Einstufung Risiko (abgekürzt ER) bezeichnet.

2.1.1.2 Die Projektthemenbereiche

Domänenwissen. Alle Eigenschaften des Projektes, welche durch die zu bewältigende, inhaltliche Aufgabenstellung bedingt sind. Beispiele sind erforderliche fachliche Kenntnisse oder gegebener fachlicher Anspruch in einem Projekt.

2.1 Vorbemerkungen

Softwareentwicklung. Alle Eigenschaften des Projektes, welche durch die softwaretechnische und die Software Management Bearbeitung der inhaltlichen Aufgabenstellung bedingt sind. Sie teilt sich in die Themenbereiche Software Engineering und Software Management auf. Software Engineering besteht aus Requirements Engineering, Analyse, Design, Implementierung und Qualitätssicherung. Software Management besteht aus Requirements Management, Change Management, Konfigurationsmanagement (siehe jeweils „Glossar“ und „Ontologie zur Herleitung der Merkmale“ im Anhang).
Projektmanagement. Alle Eigenschaften des Projektes, welche durch die organisatorische und kaufmännische Begleitung der softwaretechnischen und Software Management Bearbeitung des Projektes bedingt sind. Sie besteht aus dem kaufmännischen und dem organisatorischen Projektmanagement (siehe Kapitel 2.3.3.1).
Menschenführung. Alle Eigenschaften des Projektes, welche durch menschliche und zwischenmenschliche Faktoren bedingt sind, welche die Bearbeitung der inhaltlichen Aufgabenstellung begleiten. Sie besteht aus der Motivation, Kommunikation und den Führungsfähigkeiten (siehe Kapitel 2.3.4.1-2.3.4.3).

In der Ontologie werden die Projektthemenbereiche mit Einstufung Projekt (abgekürzt EP) bezeichnet, Domänenwissen wird mit „D“, Softwareentwicklung mit „S“, Projektmanagement mit „P“ und Menschenführung mit „M“ abgekürzt (siehe Anhang, Abschnitt „Ontologie zur Herleitung der Merkmale“).
Die komplette Systematik für die Überdeckung von Projektthemenbereichen und Risikobereichen wird nachfolgend anhand der Abbildung 9 veranschaulicht. Sie zeigt beide Dimensionen mit allen ihren Aspekten im Zusammenhang.

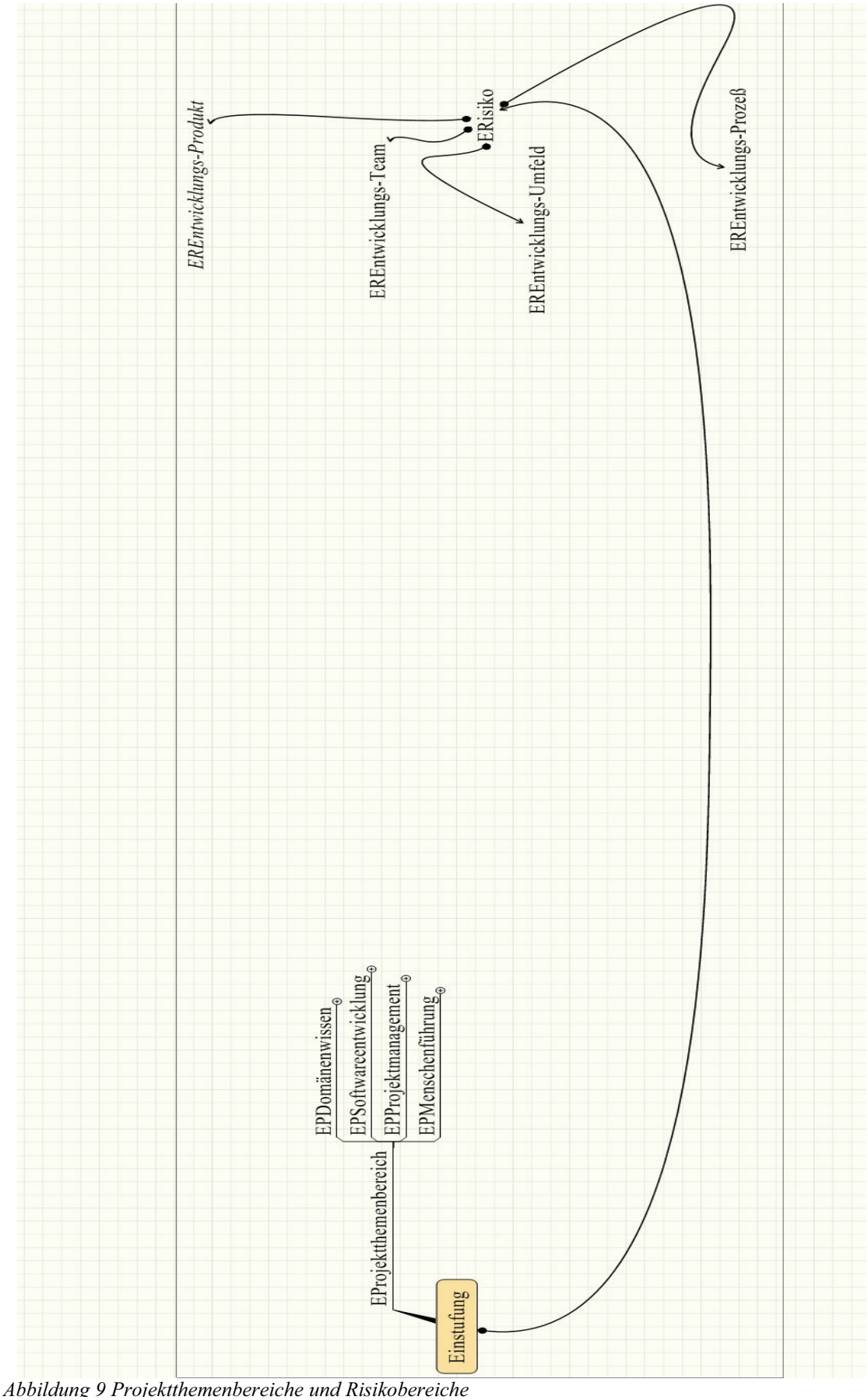

Abbildung 9 Projektthemenbereiche und Risikobereiche

2.1.2 Definieren von Merkmalen

Projektmerkmale werden im EVGM mit Werten von „--“ bis „++“ bewertet (siehe Kapitel 4) und durch Beeinflussungsbeziehungen miteinander vernetzt.

Modellierbarkeit von Merkmalen. Projektmerkmale stellen die Zustände von wesentlichen Einflußgrößen im Projekt dar. Sie werden in Ursache-Wirkungsdiagrammen modelliert und mit einem Wert versehen. Deshalb müssen ihre Namen hierfür geeignet gewählt sein.
Zustandsinformationen über ein Projekt können die **Bedarfsdeckung** von wichtigen Projektressourcen, oder die **Einfachheit**, **Stabilität** und **Transparenz** von wichtigen Projekteinflußfaktoren sein, welche durch die Merkmale dargestellt werden. Merkmale sind demzufolge immer als Bedarfsdeckung einer Ressource, beziehungsweise als Einfachheit oder Stabilität und Transparenz eines Einflußfaktors formuliert.
Definition: Bedarfsdeckung steht hierbei für das Verhältnis zwischen Verfügbarkeit und Bedarf einer Ressource. Bedarfsdeckung ist ein aussagekräftigeres Attribut für die Modellierung von Ressourcen als beispielsweise „Umfang“, weil dieser nicht die Verhältnismäßigkeit zwischen Verfügbarkeit und Bedarf ausdrückt. Beispielsweise hat eine Erweiterung des Funktionsumfanges nicht notwendigerweise eine Auswirkung auf das Budget, auf das Verhältnis zwischen dessen Bedarf und Verfügbarkeit schon. Dies kann durch die Bedarfsdeckung besser ausgedrückt werden. Bedarfsdeckung kann sich auf quantifizierbare Kapazitäten, sowie auf menschliche Fähigkeiten beziehen.
Falls die Bedarfsdeckung negativ ist, entsteht ein Engpaß und somit ein Problem für das Projekt. Ein Bedarf kann durch Erfordernisse des Entwicklungs-Produktes entstehen und kann durch die Fähigkeiten des Entwicklungs-Teams oder die Unterstützung des Entwicklungs-Prozesses gedeckt werden.
Beispiel für Bedarfsdeckung: In manchen Projekten werden ausreichend Geld und personelle Kapazitäten zur Verfügung gestellt, aber sie stehen unter starkem Zeitdruck wie das Toll Collect Projekt. Auch die Jahrzweitausend- oder die Euroumstellungen hatten kritische Endtermine. Diese sind auch in der Automobilindustrie üblich, auch um den für die Außenwirkung bedeutsamen „Start of Production“ (SOP) einer Fahrzeugserie nicht verschieben zu müssen.
In anderen Projekten werden immense personelle Kapazitäten aufgeboten, um einen bereits mehrfach verschobenen Liefertermin zu halten wie im Projekt TS90 eines Transportunternehmens, in dem der Autor der vorliegenden Arbeit selbst tätig war. Andere Projekte sind terminlich unkritisch, werden aber aufgrund der niedrigen Dringlichkeit mit sehr geringen personellen Kapazitäten ausgestattet, leiden also unter einem Engpaß in dieser Hinsicht.
Während des DotCom-Booms um die Jahrtausendwende war es nur unter schwierigsten Umständen möglich, befähigtes Personal für ein Projekt zu bekommen und hierfür mußten erhebliche Kosten in Kauf genommen werden, eine Ressourcensituation, die sich Anfang des aktuellen Jahrtausends komplett gedreht hat.
Definition: Einfachheit steht für die Beherrschbarkeit eines Einflußfaktors. Wenn ein Einflußfaktor unbeherrschbar kompliziert ist und dadurch Probleme für das Projekt aufwirft, wird die Einfachheit negativ bewertet. Einfachheit kann durch Erfordernisse des Entwicklungs-Produktes verloren gehen und kann durch die Fähigkeiten des Entwicklungs-Teams oder die Unterstützung des Entwicklungs-Prozesses kompensiert werden.
Beispiel für Einfachheit: Bei steigendem Umfang der geforderten fachlichen Funktionen und deren wechselseitiger Abhängigkeiten geht Einfachheit verloren.

Definition: Stabilität und Transparenz stehen für die Beständigkeit eines Einflußfaktors gegen unerwünschte Veränderungen, sowie dessen Erkenn- und Durchschaubarkeit. Transparenz bedeutet insbesondere die Verfügbarkeit von wesentlichen Informationen für alle von ihr betroffenen Projektteilnehmer. Stabilität und Transparenz stehen im engen Zusammenhang, da ständige Veränderungen Neues bringen, was vorher nicht erkennbar war. Neues aber verändert das Bestehende. Unbekannte oder sich fortwährend ändernde fachliche Anforderungen an das Entwicklungs-Produkt verschlechtern den Zustand von deren Stabilität und Transparenz. Dies kann durch die Fähigkeiten des Entwicklungs-Teams oder die Unterstützung des Entwicklungs-Prozesses kompensiert werden.
Beispiel für Stabilität und Transparenz: Bei sich ständig ändernden fachlichen Anforderungen geht Stabilität und Transparenz verloren.

Es werden in der vorliegenden Arbeit ausschließlich Projektmerkmale berücksichtigt, deren Relevanz für Softwareprojekte und für die Eignungseinstufung von Vorgehensmodellen nachgewiesen werden kann. Einen Überblick über die Herleitung der Projektmerkmale, die in diesem Kapitel eingeführt werden, bietet die folgende Abbildung 10.

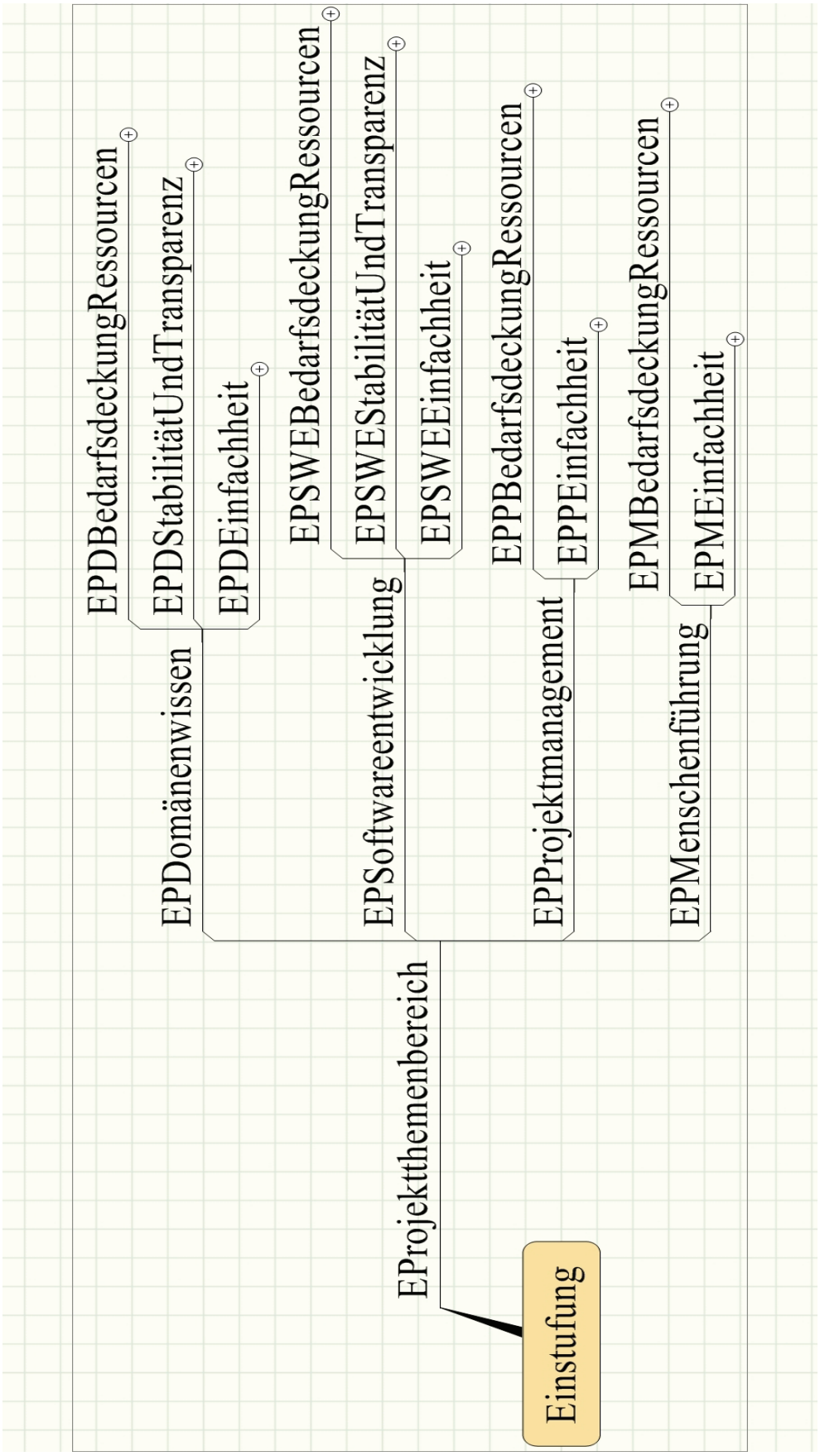

Abbildung 10 Typen von Projektmerkmalen

Beschreiben von Projektmerkmalen. Jedes vom EVGM vorgeschlagene Projektmerkmal wird in den folgenden Abschnitten jeweils zuerst definiert. Dann wird es anhand seiner Bedeutung für Softwareprojekte, für die Eignungseinstufung von Vorgehensmodellen, sowie anhand seiner Beeinflussungsbeziehungen mit anderen Merkmalen motiviert. Hierbei werden Vorgriffe auf noch nicht definierte Merkmale erforderlich.
Nur als wesentlich belegbare Beeinflussungsbeziehungen werden vom EVGM vorgeschlagen. Ihre Bedeutung wird in der vorliegenden Arbeit plausibel gemacht. Die Beeinflussungsbeziehungen werden im EVGM danach unterschieden, ob ihre Auswirkung als ungefähr proportional oder eher überproportional eingeschätzt wird. Keine der dargestellten Beeinflussungsbeziehung ist zwingend, sondern wahrscheinlich und hat daher Vorschlagscharakter. Beeinflussungsbeziehungen stellen Effekte dar, die ohne weitere Fremdeinwirkung entstehen und ablaufen. Wie die Merkmale auch müssen die Beeinflussungsbeziehungen bei der Anwendung des EVGM auf ihre projektspezifische Relevanz überprüft werden.
Die Struktur für die Beschreibung der Projektmerkmale wird mit der folgenden Abbildung 11 verdeutlicht.

EVGM - Beschreibung der Projektmerkmale

Projektmerkmale (PM)

- Definition PM
- Motivation PM
  - Bedeutung für Software Engineering
  - Bedeutung für EVGM - unterschiedliche Adressierung durch VGM
  - Beeinflussungsbeziehungen zu anderen PM
- Herleitung PM durch Belege aus der Literatur

jeweils mit Beispielen und eingeordnet in Systematik, die nach Risikobereichen und Projektthemenbereichen unterteilt ist

Abbildung 11 Beschreibung der Projektmerkmale

Als Gliederungsstruktur für die Erarbeitung der Merkmale und Fragen zu Einflußfaktoren aus dem Projektumfeld dient die bereits in Kapitel 2.1.1 dargestellte Systematik über Entwicklungs-Team, Entwicklungs-Produkt und Entwicklungs-Umfeld, sowie über Domänenwissen, Softwareentwicklung, Projektmanagement und Menschenführung. Jedes Merkmal ist dadurch in die Systematik eingeordnet.

Anmerkung zur Darstellung. Die gesamte Darstellung der Projektmerkmale erfolgt in der Art eines Handbuches, weil bei der Anwendung des EVGM darin nachgeschlagen werden muß. Die Bezeichner der Merkmale von Projekten und Vorgehensmodellen werden bei ihrer Nennung im Fließtext *kursiv* geschrieben.

2.2 Merkmale des Risikobereichs Entwicklungs-Produkt

Entsprechend der Systematik aus Kapitel 2.1.1 werden nachfolgend die Merkmale innerhalb der Projektthemenbereiche Domänenwissen, Softwareentwicklung, Projektmanagement und Menschenführung dargestellt.

2.2.1 Projektthemenbereich Domänenwissen

Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Produkt“ und des Projektthemenbereichs „Domänenwissen“ zeigt der folgende Ontologieausschnitt in Abbildung 12.

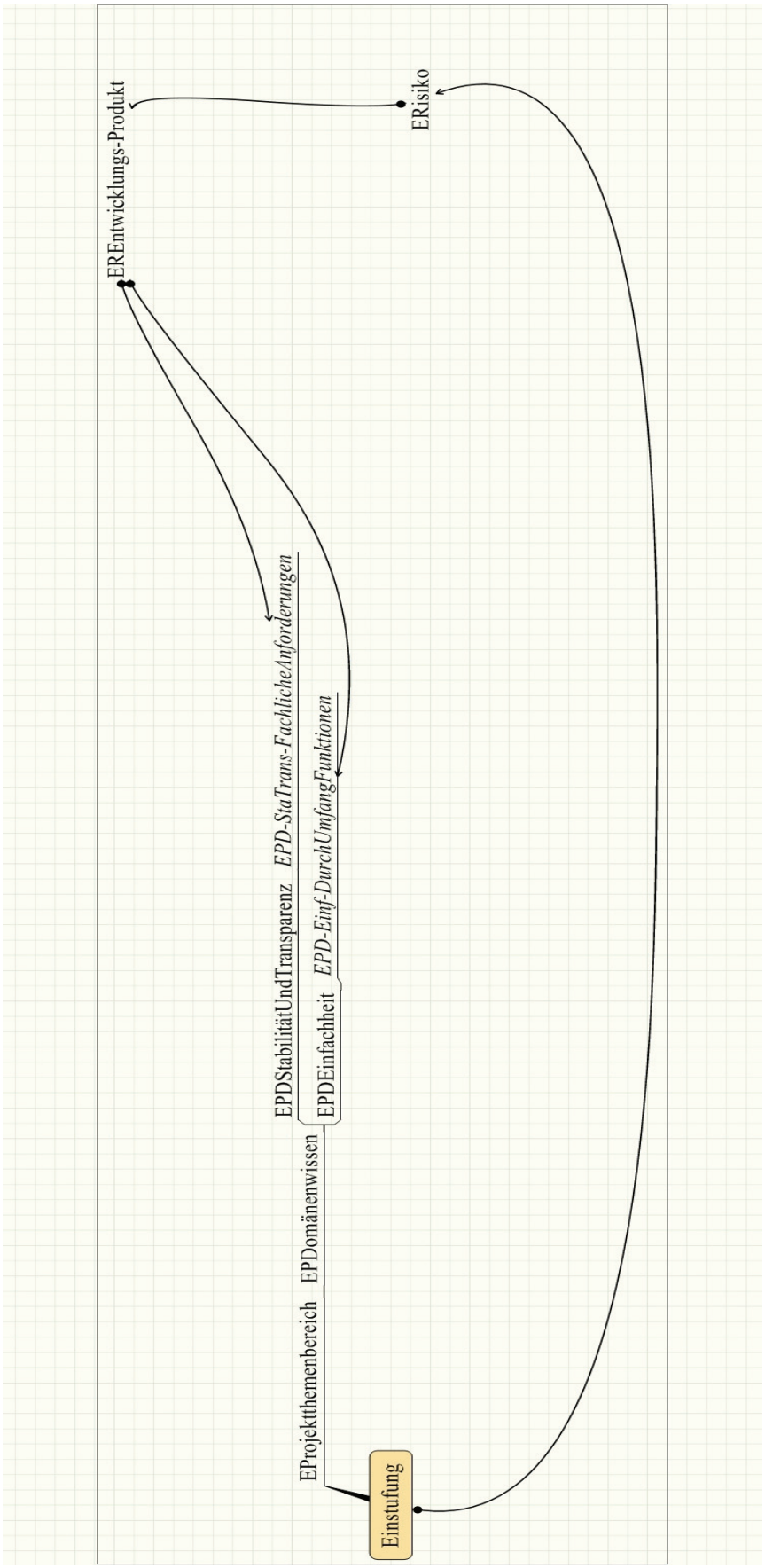

Abbildung 12 Merkmale Risikobereich Entwicklungs-Produkt und Projektthemenber. Dom.Wi.

2.2.1.1 Umfang Funktionen

Definition des Merkmals *EPD-Einf-DurchUmfangFunktionen:*
Unter dem Funktionsumfang soll die Menge an Anforderungen bezüglich fachlicher Funktionen verstanden werden. Die Einfachheit steht für die Abhängigkeiten zwischen den Funktionen und die Seiteneffekte bei Änderungen.
Beispiel: Ein System mit drei über Schnittstellen verbundenen Funktionsblöcken weist maximal drei Abhängigkeitsbeziehungen auf, eines mit vier Funktionsblöcken mindestens drei und maximal sechs Abhängigkeiten.

Motivation des Merkmals *EPD-Einf-DurchUmfangFunktionen:*
Bedeutung für Softwareprojekte. Mit steigendem Funktionsumfang steigen die Abhängigkeiten zwischen den einzelnen Funktionalitäten (siehe Beispiel oben) und die Wahrscheinlichkeit von Änderungen an den Anforderungen. Diese Änderungen können wegen der Abhängigkeiten zu großen Seiteneffekten führen. Hierbei müssen unter Umständen bereits fertiggestellte Arbeitsprodukte wieder verändert werden, wodurch die bereits in sie investierten Ressourcen zumindest teilweise vergeudet waren. Dadurch werden Zeit, Geld und personelle Kapazität für Arbeiten investiert, die hinterher nicht wertschöpfend in das Projektergebnis einfliessen. Dies trägt zur Ineffizienz der eingesetzten Projektressourcen Zeit, Geld und personelle Kapazitäten bei und kann, wenn diese nur begrenzt zur Verfügung stehen, zum Mißerfolg des Projektes führen.
Beispiel: Da die Gesamtfunktionalität eines Softwaresystems immer Abhängigkeiten an den Schnittstellen der einzelnen Bestandteile aufweist, wachsen diese mit der Anzahl der Bestandteile ebenfalls. Wenn ein kaufmännisches Softwaresystem aus den Bestandteilen Warenwirtschaft, Finanzbuchhaltung und Customer Relationship Management besteht, kommunizieren diese über Schnittstellen miteinander und nutzen gemeinsame Daten, wie die Stammdaten von Kunden beziehungsweise Geschäftspartnern. Die dadurch entstehenden Abhängigkeiten werden durch ein hinzufügen eines Bestandteils Kostenrechnung erhöht, da diese ebenfalls auf Geschäftspartnerstammdaten zugreifen. Durch diese Abhängigkeiten entstehen aber Seiteneffekte bei Änderungen, die Zusatzaufwände erzeugen. Dies vermindert die Effizienz des Ressourceneinsatzes von Projektbudget, kalendarischer Zeit und personeller Kapazität und gefährdet somit den Projekterfolg.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. In Vorgehensmodellen wird mitunter versucht, einen großen Funktionsumfang mit großen Arbeitsproduktmodellen zu adressieren, in denen auch Anforderungen bezüglich fachlicher Funktionen festgehalten werden. Auch mit speziellen Rollen und Aktivitäten der Disziplin Requirements Engineering und Management versucht man, einen großen Funktionsumfang beherrschbar zu halten.
Beispiel: Vorgehensmodelle bieten sehr unterschiedliche Unterstützung zum Umgang mit großen Funktionsumfängen. Das V-Modell XT [VMX06] unterstützt mit dem Vorgehensbaustein *Anforderungsfestlegung*, den Rollen *Anforderungsanalytiker* und *Anwender*, den Arbeitsprodukten *Anforderungen (Lastenheft)* und *Anforderungsbewertung*, sowie den Aktivitäten *Anforderungen festlegen* und *Anforderungsbewertung erstellen* umfassend den Umgang mit einem großen Funktionsumfang. Extreme Programming [Bec00] bietet hingegen *Story Cards* zur Verwaltung von fachlichen Funktionen und verweist sonst darauf, Projekte mit zu großem Funktionsumfang in kleinere Teilprojekte zu zerlegen (siehe Kapitel 3.3). Das ist allerdings nicht immer sinnvoll und möglich. Wenn die Abhängigkeiten zwischen den Funktionalitäten nicht ausreichend re-

duziert werden können, ist keine hinreichend unabhängige Unterteilung der Funktionalität möglich.

Beeinflussungsbeziehungen mit anderen Merkmalen
Ein steigender Funktionsumfang kann mehr Mitarbeiter zu seiner Entwicklung erfordern, was die bei gleichbleibender personeller Kapazität deren Bedarfsdeckung verschlechtert. Die Beeinflussung von *EPD-Einf-DurchUmfangFunktionen* auf *EPP-BedDeck-Kap-Personell* proportional wird eingeschätzt und daher mit „≈“ vorgeschlagen.
Steigender Funktionsumfang bewirkt höhere Produktkosten, was sich bei gleichbleibender Ausstattung mit finanziellen Mitteln negativ auf die Bedarfsdeckung des Budgets auswirkt. Die Beeinflussung von *EPD-Einf-DurchUmfangFunktionen* auf *EPP-BedDeck-Kap-Projektbudget* wird proportional eingeschätzt und mit „≈“ vorgeschlagen.
Ein steigender Funktionsumfang kann sich bei gleichbleibender Mitarbeiterzahl verschlechternd auf die Terminsituation auswirken, weil es länger dauert, ihn zu entwickeln. Auch diese Beeinflussung von *EPD-Einf-DurchUmfangFunktionen* auf *EPP-BedDeck-Kap-KalendarischeZeit* wird proportional eingeschätzt und mit „≈“ vorgeschlagen.
Steigender Funktionsumfang wirkt er sich auch negativ auf die Stabilität und Transparenz der fachlichen Anforderungen aus, da sich durch ihn Modulabhängigkeiten und Änderungswahrscheinlichkeiten erhöhen. Auch die Beeinflussung von *EPD-Einf-DurchUmfangFunktionen* auf *EPD-StaTrans-FachlicheAnforderungen* wird proportional eingeschätzt und mit „≈“ vorgeschlagen.
Durch steigenden Funktionsumfang erhöhen sich die Fähigkeitsanforderungen im Domänenwissen, im Requirements Engineering und Management, sowie im Software Engineering und Management. Dadurch wird deren Bedarfsdeckung schlechter, wenn nicht verfügbare Fähigkeiten aufgestockt werden. Auch die Beeinflussung von *EPD-Einf-DurchUmfangFunktionen* auf *EPD-BedDeck-Fähigk-Fachlich*, auf *EPSWE-BedDeck-Fähigk-SoftwareEngineering* und der *EPSWE-Deck-Fähigk-SoftwareManagement* werden proportional eingeschätzt und werden mit „≈“ vorgeschlagen.
Die *EPD-Einf-DurchUmfangFunktionen* kann durch den Themenbereich Requirements Engineering beeinflusst werden, die eine strukturierte Darstellung desselben unterstützen. Dies erhöht die Anforderungen an die *EV-UnterstProjTheBer-RequirementsEngineering* (siehe Kapitel 3.2).

Belege aus der Literatur
[Gaul04] führt in seiner Ausarbeitung über Projektkenngrößen, Risiko- und -Erfolgsfaktoren den „Projektumfang“ an. Hierbei nennt er den Funktionsumfang des Projektproduktes und die Größe des Entwicklungsteams als Facetten. Auch Tom DeMarco benennt in [DeM03] „Inflation der Anforderungen“, „ausufernde Anforderungen“ und „Spezifikationskollaps“ im Sinne von ausufernden Funktionalitätsanforderungen als Kernrisiko in Projekten.

2.2.1.2 Stabilität und Transparenz fachlicher Anforderungen

Definition des Merkmals *EPD-StaTrans-FachlicheAnforderungen*:
Die Stabilität und Transparenz fachlicher Anforderungen beschreibt, welche fachlichen Eigenschaften durch eine Software abgedeckt werden soll. Fachliche Anforderungen dienen im Vergleich zu technischen Anforderungen direkt und eigenständig dem Systemzweck. Stabilität steht für die Beständigkeit gegen unerwünschte Veränderungen,

Transparenz für Erkennbarkeit und Durchschaubarkeit.
Beispiel: Die Anforderung nach einer Funktionalität zum Verbuchen von kaufmännischen Vorgängen ist direkt dem Zweck eines entsprechenden Systems dienlich. Im Gegensatz hierzu steht eine begleitende Antwortzeitanforderung, welche aber ohne die Verbuchungsfunktionalität, auf welche sie sich bezieht, nicht sinnvoll wäre. Falls die Anforderung nach Verbuchungsfunktionalität sich ständig ändert, ist sie instabil, wenn sie dem Entwicklungsteam nicht bekannt ist, intransparent.

Motivation des Merkmals *EPD-StaTrans-FachlicheAnforderungen*:
Bedeutung für Softwareprojekte. Instabile Anforderungen sind oft beobachtbar, wenn fachliche Zusammenhänge bei der Anforderungsanalyse nicht konsequent durchdacht, im Zusammenhang betrachtet und ohne Berücksichtigung von absehbaren zukünftigen Veränderungen von Rahmenbedingungen untersucht werden. Intransparenz entsteht oft, wenn fachliche Ansprechpartner nur Spezialfälle ihrer Arbeit, nicht aber den Normalfall beschreiben, beispielsweise, weil ihnen dieser als selbstverständlich erscheint.
Fachliche Anforderungen, die dem Projektteam nicht bekannt sind, können nicht erfüllt werden, was den Nutzen des Projektes und damit seinen Erfolg gefährdet. Intransparente Anforderungen führen fast sicher zu einem Projektfehlschlag. Aufgrund der Anzahl gegebener Lösungsmöglichkeiten ist es nicht wahrscheinlich, daß zufällig ein bedarfsgerechtes Projektergebnis entwickelt wird.
Instabile fachlichen Anforderungen sind zum Zeitpunkt der Fertigstellung ihrer Implementierung unter Umständen bereits wieder durch eine Anforderungsänderung überholt. Dann deckt die implementierte Funktionalität sie nicht ab und muß ebenfalls überarbeitet werden, um den geänderten Anforderungen zu genügen. Daher vermindert die Änderung bereits umgesetzter Anforderungen die Effizienz und Effektivität der eingesetzten Ressourcen personelle Kapazität, Geld und Zeit, da bereits erstellter Code unter Umständen neu entwickelt werden muß und die in ihn investierten Aufwände nicht in die Wertschöpfung des Projektes einfliessen. Dies gefährdet den Nutzen des Projektes und damit seinen Erfolg.
Ständig geänderte Anforderungen erschweren die Projektplanung. Anforderungen haben Einfluß auf die Aufwandsschätzungen für die Implementierung. Daher müssen diese bei instabilen oder intransparenten Anforderungen fortwährend überarbeitet werden.
Für die Gewinnung von fachlichen Anforderungen muß das Wissen der fachlichen Benutzer erschlossen werden.
Beispiel: Das Erkennen der Anforderung eines Buchgroßhändlers, daß seine Datenbasis auch Artikel verwalten können muß, die nicht durch eine ISBN identifizierbar sind, kann zur Verdeutlichung herangezogen werden. Erst durch das transparent werden dieser Anforderung wird offensichtlich, daß die ISBN nicht als Schlüsselkriterium für die Datenbank verwendet werden kann. Da sie den Wert „null“ (im Sinne eines undefinierten Zustandes) annehmen kann, ist sie nicht eindeutig identifizierend.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen Stabilität und Transparenz von fachlichen Anforderungen im unterschiedlichen Maße. Disziplinen wie Requirements Engineering, Requirements Management unterstützen die methodische und gründliche Erhebung und Formulierung von Anforderungen wie das V-Modell XT [VMX06]. Die gesamte Landschaft der Agilen Methoden [AGI06] stellt das flexible Reagieren auf geänderte Anforderungen mit dem Prinzip *Embrace Change* in den Vordergrund. Mit iterativen, inkrementellen oder evo-

lutionären Ansätzen wird versucht, mit minimalisierten Arbeitsproduktmodellen möglichst flexibel gegen Veränderungen von Anforderungen zu sein. Oft sind diese minimalisierten Arbeitsproduktmodelle auf Code und Testfälle reduziert.
Beispiel: In Extreme Programming [Bec00] kann der *on-site Customer* befragt werden, um den Entwicklern noch unbekannte fachliche Anforderungen aufzudecken. Bei neuen oder geänderten Anforderungen müssen nur Code und Testfälle geändert werden.
Hingegen können etablierte und umfassende Vorgehensmodelle wie das V-Modell XT [VMX06] aufgrund ihres iterativen und teilweise inkrementellen Ansatzes Instabilität von Anforderungen, Zielen und Rahmenbedingungen in Maßen kompensieren, würden aber aufgrund ihrer umfassenden Arbeitsproduktmodelle bei evolutionären Änderungen durch die erforderlichen Aktualisierungen von Konzepten, Dokumenten und Modellen das Projekt stark verlangsamen (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Eine niedrige Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* bedingt mehrfache Erstellung beziehungsweise Überarbeitung von Projekt-Arbeitsprodukten. Sie vermindert so die Effizienz der eingesetzten Projektressourcen Personal, Budget und Zeit. Das verschlechtert die Bedarfsdeckung der personellen Kapazitäten (*EPP-BedDeck-Kap-Personell),* die Bedarfsdeckung des Projektbudgets (*EPP-BedDeck-Kap-Projektbudget)* und die Bedarfsdeckung der kalendarischen Zeit (*EPP-BedDeck-Kap-KalendarischeZeit),* wenn nicht entsprechende Kapazitätserhöhungen stattfinden. Dieser Einfluß wird überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Schlechte Stabilität und Transparenz fachlicher Anforderungen bewirkt erhöhten Bedarf an Requirements Engineering Fähigkeiten, verschlechtert also den Zustand von *EPS-WE-BedDeck-Fähigk-SoftwareEngineering.* Dieser Einfluß wird proportional eingeschätzt und mit „≈“ vorgeschlagen.
Wenn aufgrund schlechter Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* keine Funktionalität ausgeliefert und in die Nutzung überführt wird, bleibt den Projektmitarbeitern ein Erfolgserlebnis verwehrt, was demotivierend wirken kann. Die Motivation im Team (*EPM-BedDeck-MotivationImTeam)* verschlechtert sich also ebenfalls. Dieser Einfluß wird überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Die *EPD-StaTrans-FachlicheAnforderungen* kann durch die Unterstützung des Requirements Engineering in einem Vorgehensmodell (*EV-UnterstProjTheBer-RequirementsEngineering)* verbessert werden. Auch ein flexibler iterativer und inkrementeller Ansatz durch ein Vorgehensmodell (*EV-Flexib-DurchIterativitätUndInkrementalität)* adressiert fehlende *EPD-StaTrans-FachlicheAnforderungen*, weil jedes iterativ entwickelte Inkrement einen kleinen Teil der Gesamtanforderungen herausgreift, womit sich beim noch nicht implementierten Rest Anforderungsänderungen nur in den Anforderungsspezifikationen auswirken. Die Unterstützung des Software Engineering (*EV-UnterstProjTheBer-SoftwareEngineering)* können beispielsweise über Prototypen zur Erkundung von unbekannten oder unsicheren Anforderungen beitragen. Er erhöht die Anforderungen an die Unterstützung durch das Software Management (*EV-UnterstProjTheBer-SoftwareManagement)*, speziell bezogen auf das Requirements- und Change Management (siehe Kapitel 3.2).

Belege aus der Literatur

Stabilität der fachlichen Anforderungen. Änderungen fachlicher Anforderungen werden in der Literatur als Risikofaktoren erwähnt. Beispiele sind „zu viele Änderungen an Anforderungen“ bei Jalote, zitiert nach [Gaul04], „Ständiger Strom an neuen Anforderungen“ bei Boehm, zitiert nach [Gaul04], „Nachträgliche Änderungen an den Anforderungen“ [Kel94], „Ständige Funktionsänderungen und -erweiterungen des vorgesehenen Systems“ [Gru01], „sich ständig ändernde Ziele und Anforderungen“ [Rup01], „Ständige Änderungen der Anforderungen“ bei [Boe98] zitiert nach [Wal04]und „Fehlende Änderungskontrolle“ bei McConnel zitiert nach [Gaul04] „changing requirements“ [Hum02].

Transparenz der fachlichen Anforderungen. Ein weiteres oft genanntes Projektrisiko sind intransparente Anforderungen. Die Literatur führt hier „Unklare Anforderungsanalyse“ bei Computerwoche zitiert nach [Gaul04], „Fehlen von Prioritäten“, bei Daily Telegraph zitiert nach [Gaul04], „unklare Anforderungen“ bei Jalote zitiert nach [Gaul04], „Unklare Anforderungen“ [Ver00], „Geschäftsprozesse“ [Gaul04] im Sinne der Unkenntnis der Entwickler über die Geschäftsprozesse, welche die zu entwickelnde Software unterstützen soll. [Rup01] benennt als Projektrisikofaktor „schlechte Qualität der Anforderungen“ im Sinne von mehrdeutigen und damit interpretierbaren und unpräzisen Beschreibungen, welche die Anforderungen nicht hinreichend gründlich und transparent darstellen. Bestätigt wird dies durch unkoordinierte und widersprüchliche „Anforderungen“ bei [Hal98] zitiert nach [Wal04] sowie nicht hinreichend definiert und stabile „Anforderungen“ [Wal04] und [Wal01] als Projektrisiko. Dies führt auf „Entwicklung falscher Benutzerschnittstellen“ bei Boehm zitiert nach [Gaul04], „Entwicklung von Zusatzfunktionalität“ im Sinne von „Golden Plates“ bei Boehm zitiert nach [Gaul04], sowie „Anforderungen und entwickelte Funktionen stimmen nicht überein“ beziehungsweise „Benutzerschnittstellen stimmen nicht mit Bedürfnissen überein“ bei [Boe98] zitiert nach [Wal04] und „Entwicklung falscher Funktionalitäten und Eigenschaften“ bei Boehm zitiert nach [Gaul04]. Der Chaos Report legt als Projekterfolgsfaktor „Clear Statement of Requirement“ [Sta94] nahe.

2.2.2 Projektthemenbereich Softwareentwicklung

Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Produkt“ und des Projektthemenbereichs „Softwareentwicklung“ zeigt der folgende Ontologieausschnitt in Abbildung 13.

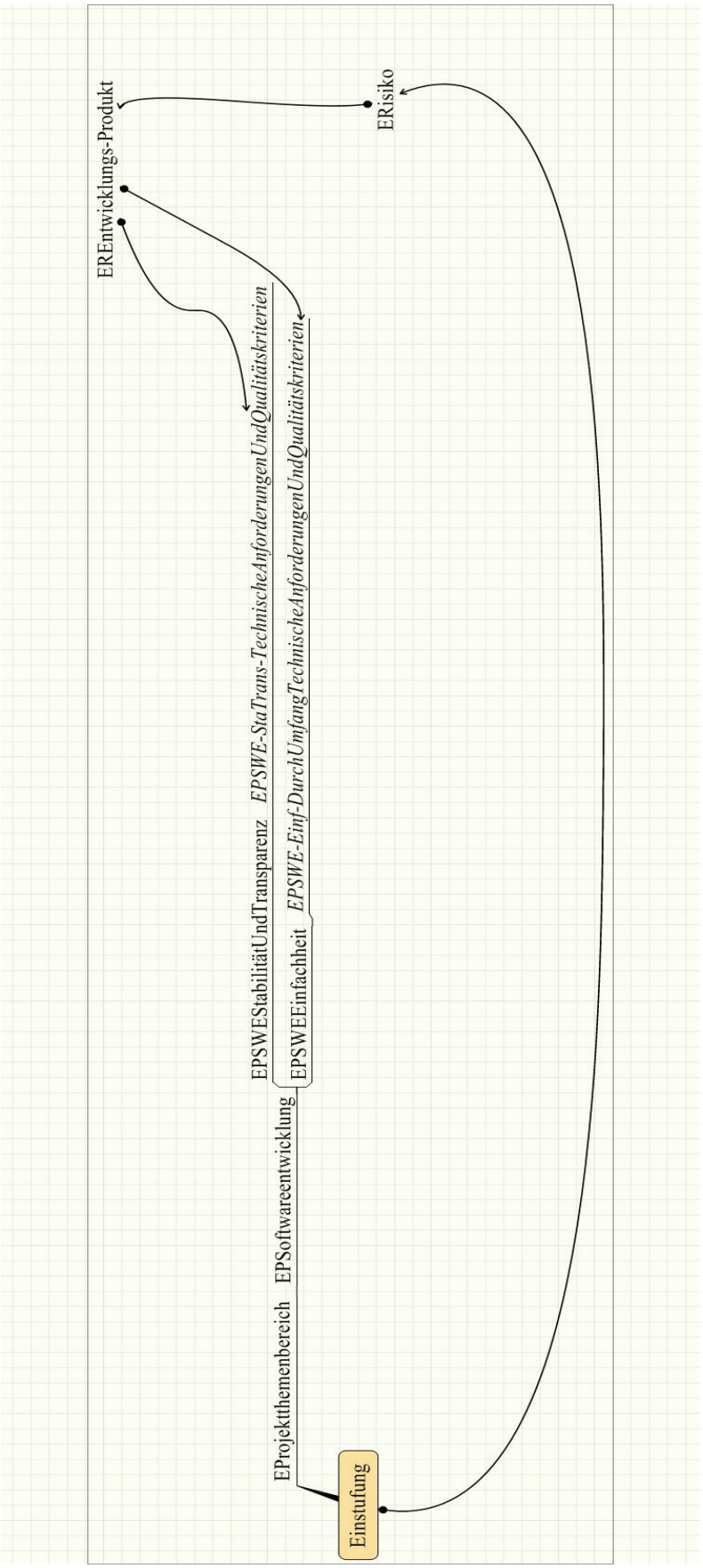

Abbildung 13 Merkmale Risikobereich Entwicklungs-Produkt und Projektthemenber.Softwareentw.

2.2.2.1 Umfang technische Anforderungen und Qualitätskriterien

Im Gegensatz zur *EPD-Einf-DurchUmfangFunktionen*, welche die fachlichen Funktionen betrifft, werden nun technische Aspekte betrachtet. Das Projektmerkmal *EPSWE-Einf-Durch-UmfangTechnischeAnforderungenUndQualitätskriterien* umfasst „Effizienz“, „Funktionalität“, „Übertragbarkeit“, „Änderbarkeit“, „Zuverlässigkeit“ und Benutzbarkeit“ entsprechend den Definitionen in der DIN ISO 9126 zitiert nach [Bal98], wie nachfolgend dargestellt wird. Detaildefinitionen können im Glossar im Anhang nachgelesen werden.

Definition von Effizienz: „Verhältnis zwischen dem Leistungsniveau der Software und dem Umfang der eingesetzten Betriebsmittel unter festgelegten Bedingungen.“ [Bal98].
Definition von Funktionalität: „Vorhandensein von Funktionen mit festgelegten Eigenschaften. Diese Funktionen erfüllen die definierten Anforderungen.“ [Bal98].
Definition von Übertragbarkeit: „Eignung der Software, von einer Umgebung in eine andere übertragen zu werden. Umgebung kann organisatorische Umgebung, Hardware-, oder Softwareumgebung einschliessen.“ [Bal98].
Definition von Änderbarkeit: Aufwand, der zur Durchführung vorgegebener Änderungen notwendig ist. Änderungen können Korrekturen, Verbesserungen oder Anpassungen an Änderungen der Umgebung, der Anforderungen und der funktionalen Spezifikationen einschließen.“ [Bal98].
Definition von Zuverlässigkeit: „Fähigkeit der Software, ihr Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum zu bewahren.“ [Bal98].
Definition von Benutzbarkeit: „Aufwand, der zur Benutzung erforderlich ist, und individuelle Beurteilung der Benutzung durch eine festgelegte oder vorausgesetzte Benutzergruppe.“ [Bal98].
Ergänzung dieser Definitionen. Als „innere Benutzbarkeit“ kann man die **Wiederverwendbarkeit** auffassen. Diese Eigenschaft kann beschrieben werden als Anpassungsaufwand für ein Softwaremodul, dieses entsprechend seiner intendierten Funktionalität in einem anderen als dem vorgesehenen Umfeld nutzbar zu machen. Beispielsweise Bibliotheksroutinen sind dafür gedacht, ohne Anpassungsaufwand wiederverwendet zu werden.
Zusammenfassung. Wenn ein Projektergebnis nicht die erforderlichen Qualitätseigenschaften aufweist, hat es für für den Auftraggeber einen geringeren Nutzen und vermindert so die Effizienz der eingesetzten Projektressourcen kalendarische Zeit und Projektbudget. Dadurch gefährdet es den Projekterfolg.

Definition des Merkmals *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUnd-Qualitätskriterien*:
Hierunter wird die Gesamtheit der oben definierten Qualitätskriterien verstanden, an denen die zu erstellende Software gemessen wird und die bei der Erfüllung der fachlichen Anforderungen nicht verletzt werden dürfen. Im Bedarfsfall kann bei der Anwendung des EVGM projektspezifisch nach einzelnen, besonders signifikanten Qualitätskriterien diversifiziert werden. Qualitätskriterien führen in Verbindung mit Qualitätszielen auf konkrete technische Anforderungen. Einfachheit steht hier für den bestehenden Schwierigkeitsgrad der Aufgabe, den mitunter widersprüchlichen Qualitätskriterien zu genügen.
Beispiel: Interoperabilität als Qualitätskriterium führt auf eine konkrete technische Anforderung, welche spezifiziert, wie und auf welchen Plattformen Interoperabilität gefor-

dert wird und das zu entwickelnde System lauffähig sein muß. Das Qualitätsziel, auf Microsoft Windows XP™ beziehungsweise Sun Solaris™ lauffähig zu sein, konkretisiert diese Plattformen, was einer technischen Anforderung entspricht.

Motivation des Merkmals *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUnd-Qualitätskriterien:*

Bedeutung für Softwareprojekte. Qualitätskriterien und die aus ihnen resultierenden technischen Anforderungen an ein Projektergebnis haben starken Einfluß auf ein Softwareentwicklungsprojekt. Da man Qualität nicht im Nachhinein in ein Softwaresystem hinein testen kann, muß sie während des gesamten Projektverlaufes definiert, geplant, umgesetzt, kontrolliert und nachgesteuert werden. Während man fachliche funktionale Anforderungen potentiell inkrementell umsetzen und ein Produkt so schrittweise erweitern kann, ist dies bei inneren Qualitätskriterien wie beispielsweise Sicherheit und Stabilität nicht ohne weiteres möglich. Wenn diese nicht von Beginn der Entwicklung an bekannt sind und ihnen hinreichend Rechnung getragen wird, kann es sehr aufwendig und schwierig werden, sie im Nachhinein noch zu berücksichtigen. Da sie sich meist nicht auf dezidierte und isoliert betrachtbare Bestandteile eines Softwaresystems, sondern aufgrund ihres Querschnittscharakters meist auf das gesamte System beziehen, sind große Änderungsumfänge und Seiteneffekte zu erwarten. Weiterhin steigen die Kosten pro Codezeile mit jeder Restriktion überproportional an. Zum Beispiel sind bei hohen Performance-Anforderungen neben der Logik der algorithmischen Lösung für die fachliche Funktionalität zusätzlich Laufzeitrestriktionen zu beachten. Das macht die Entwicklung der fachlichen Funktion komplizierter, als wenn keine Laufzeitrestriktionen zu beachten sind. Dies führt zu einem ineffizienteren Einsatz der Ressourcen Zeit, Geld und personelle Kapazität führt und gefährdet so den Projekterfolg.

Beispiel: „Übertragbarkeit“ [Bal98] muß ein Projektteam schon während der Analyse der technischen Anforderungen berücksichtigen. Sie hat weiterhin einen signifikanten Einfluß auf das Design, weil sie eine Schichtenarchitektur für das Softwaresystem erfordert.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle adressieren Produktqualitätsanforderungen unterschiedlich. Dies geschieht unter anderem mit Disziplinen wie Qualitätssicherung, Qualitätsmanagement, Requirements Engineering und Requirements Management. Manche Vorgehensmodelle erwähnen nur Akzeptanztests mit den Benutzern wie Scrum [Sch02], manche entwickeln Testfälle vor dem Produktivcode wie XP [Bec00]. Ganz allgemein bieten Scrum und XP keine systematische, durch einen ordentlichen Prozeß aufgebaute Qualitätssicherung und Qualitätsmanagement [Boe04]. Insbesondere werden dort keine expliziten Maßnahmen konstruktiver Qualitätssicherung [Bal98], sondern nur Funktions- bzw. White Box Tests beschrieben. Innere Qualitätskriterien von Software wie beispielsweise Änderbarkeit werden kaum betrachtet, beispielsweise durch „Refactoring“ [Fow99]. Die Komplexität der Qualitätssicherung wird überwiegend den Fähigkeiten der Entwickler und Stakeholder überlassen. Diese beiden Ansätze haben kein umfassendes systematisches Herangehen an das Qualitätsthema, wie beispielsweise das V-Modell XT [VMX06] mit expliziten Rollen, Arbeitsprodukten und Aktivitäten des Vorgehensbausteins *Qualitätssicherung*.

Beispiel: Die definierten Qualitätskriterien aus der DIN ISO 9126 zitiert nach [Bal98] werden durch Vorgehensmodelle adressiert wie folgt:

Funktionalität mit den Aspekten Richtigkeit, Angemessenheit, Interoperabilität,

Ordnungsmäßigkeit [sic!] und Sicherheit werden durch das V-Modell XT durch die Vorgehensbausteine *Qualitätssicherung, Anforderungsfestlegung, Systemerstellung, HW-Entwicklung, SW-Entwicklung, Lieferung und Abnahme* [VMX06] unterstützt. Während der *Anforderungsfestlegung* wird die richtige Funktionalität spezifiziert, während der *Systemerstellung, HW-Entwicklung* und *SW-Entwicklung* erstellt, durch die *Qualitätssicherung* überprüft und bei *Lieferung und Abnahme* durch den Auftraggeber bestätigt. Hieran wird ersichtlich, daß sich die Berücksichtigung der Produktqualität quer durch den gesamten Entwicklungsprozeß zieht. Die Agilen Ansätze sind hier sehr viel sparsamer. Extreme Programming [Bec00] bietet *Test first*, Scrum [Sch02] fordert *Sprint Reviews* zur Überprüfung des Funktionalitäts-Qualitätskriteriums. Diese adressieren nur bedingt die Interoperabilität, Ordnungsmäßigkeit [sic!], Sicherheit, für die beispielsweise im V-Modell XT [VMX06] im Rahmen des Tailoring Mechanismus ein Projektmerkmal „Safety und Security" untersucht wird, um Erkenntnisse über die Sicherheitsrelevanz des zu erstellenden Softwaresystems zu gewinnen.

Zuverlässigkeit mit den Aspekten Reife, Fehlertoleranz und Wiederherstellbarkeit wird im V-Modell XT [VMX06] durch die Vorgehensbausteine *Qualitätssicherung*, *Anforderungsfestlegung*, *Systemerstellung*, *HW-Entwicklung, SW-Entwicklung, Lieferung und Abnahme* [VMX06] adressiert. Agile Ansätze wie Extreme Programming [Bec00] fordern *Test first,* Scrum [Sch02] beschreibt Akzeptanztests.
Speziell die Kritikalität eines Softwaresystems im Sinne der Safety ist ein oft genanntes Projektrisiko [Gau02]. Die Methodenfamilie Crystal verwendet sie als Kriterium für die Methodenauswahl [Coc05].
Der Aspekt Kritikalität ist wichtig, weil durch sie die Erfordernis steigt, die Wahrscheinlichkeit von Ausfall und Fehlfunktionen von Softwaresystemen durch umfassende Qualitätsmanagement- und Sicherungsmaßnahmen zu minimieren.
Das Projektkriterium der Kritikalität wird von unterschiedlichen Vorgehensmodellen unterschiedlich adressiert. Wiederum stellt das V-Modell XT [VMX06] anhand von Rollen, Aktivitäten und Arbeitsprodukten der Vorgehensbausteine *Qualitätssicherung, Anforderungsfestlegung, Systemerstellung, HW-Entwicklung, SW-Entwicklung, Lieferung und Abnahme* Unterstützung für eine Entwicklung kritischer Systeme bereit, während beispielsweise für Extreme Programming vom Einsatz für solche Projekte explizit abgeraten wird [Bec00] (siehe Kapitel 3.3).

Benutzbarkeit mit den Facetten Verständlichkeit, Erlernbarkeit, Bedienbarkeit wird durch das V-Modell XT unterstützt durch die Vorgehensbausteine *Benutzbarkeit und Ergonomie, Qualitätssicherung, Anforderungsfestlegung, SW-Entwicklung, Lieferung und Abnahme* [VMX06]. Extreme Programming [Bec00] verweist hier wiederum auf *Test first*, Scrum [Sch02] auf *Sprint Reviews* (siehe Kapitel 3.3). Die Wiederverwendbarkeit, betrachtet als innere Benutzbarkeit" (siehe oben) wird nur von Select Perspective [All98] umfassend unterstützt, welches in der vorliegenden Arbeit nicht berücksichtigt wurde (siehe auch Kapitel 5.3 „Die Auswahl der Vorgehensmodelle").

Effizienz mit den Teilaspekten Zeitverhalten und Verbrauchsverhalten unterstützt das V-Modell XT durch die Vorgehensbausteine *Qualitätssicherung*, *Anforderungsfestlegung*, *Systemerstellung*, *HW-Entwicklung, SW-Entwicklung, Lieferung und Abnahme* [VMX06]. Die Agilen Vorgehensweisen wie Extreme Programming [Bec00] arbeiten mit *Test first*, Scrum [Sch02] mit *Sprint Reviews* (siehe Kapitel 3.3). Sie adressieren nicht explizit das Verbrauchsverhalten.

Änderbarkeit mit den Facetten Analysierbarkeit, Modifizierbarkeit, Stabilität, Prüfbarkeit werden im V-Modell XT unterstützt mit den Vorgehensbausteinen *Quali-*

tätssicherung, Anforderungsfestlegung, Systemerstellung, HW-Entwicklung, SW-Entwicklung, Lieferung und Abnahme [VMX06]. Extreme Programming [Bec00] bietet hinsichtlich der Analysierbarkeit und Modifizierbarkeit *Refactoring* [Fow99], hinsichtlich der Stabilität *Test first* [Bec00], die Prüfbarkeit wird nicht unterstützt. Scrum [Sch02] adressiert das Qualitätskriterium der Änderbarkeit nicht explizit, fordert es durch seinen flexiblen Entwicklungsansatz aber implizit (siehe Kapitel 3.3).

Übertragbarkeit mit den Aspekten Anpaßbarkeit, Installierbarkeit, Konformität, Austauschbarkeit wird durch das V-Modell XT mit den Vorgehensbausteinen *Qualitätssicherung, Anforderungsfestlegung, Systemerstellung, HW-Entwicklung, SW-Entwicklung, Lieferung und Abnahme* [VMX06] unterstützt. Die Agilen Ansätze Extreme Programming [Bec00] und Scrum [Sch02] lassen dieses Qualitätskriterium unbeachtet.

Beeinflussungsbeziehungen mit anderen Merkmalen

Hohe Qualitätsanforderungen (*EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien)* erfordern den erhöhten Einsatz von Entwicklungsressourcen, um Aktivitäten der konstruktiven und analytischen Qualitätssicherung (siehe [Bal98]) ausführen zu können. Dies verschlechtert bei gleichbleibenden Kapazitäten die Bedarfsdeckung personeller Kapazitäten (*EPP-BedDeck-Kap-Personell)*, die Budgetsituation (*EPP-BedDeck-Kap-Projektbudget)* und die die Terminsituation (*EPP-BedDeck-Kap-KalendarischeZeit)*. Diese Einflüsse werden proportional eingeschätzt und mit „≈“ vorgeschlagen.

Die Erfüllung von hohen Qualitätsanforderungen durch die genannten Qualitätssicherungsmaßnahmen benötigt auch qualifiziertere Mitarbeiter, daher verschlechtert sie auch die *EPSWE-BedDeck-Fähigk-SoftwareEngineering.* Dies wird durch die erhöhte Anforderung an das Requirements Engineering der technischen Anforderungen noch bestärkt. Dieser Einfluß wird überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.

Die *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien* kann durch ein Arbeitsproduktmodell unterstützt werden. Dies kann im Rahmen der Unterstützung durch das Requirements Engineering (*EV-UnterstProjTheBer-Requirements-Engineering)* und der Unterstützung durch das Software Engineering (*EV-UnterstProj-TheBer-SoftwareEngineering)* erfolgen (siehe Kapitel 3.2).

Belege aus der Literatur

„Qualität“ wird von [Kel94] als wichtige Projektkenngröße gesehen. Auch weitere Fachliteratur zu Projektrisikofaktoren bezeichnen „Vernachlässigung der Qualitätssicherung“ bei McConnel zitiert nach [Gaul02], „Dilettantische Tests und fehlende Qualitätskontrolle“ [Gru01], „Fehlende Qualitätssicherung“ [Ver00], „Nichterfüllung von Qualitätsanforderungen“ [Wic00], „poor quality work“ [Hum02] als erfolgskritisch für Projekte.

Effizienz - Belege aus der Literatur. „Performancemängel“ bei Boehm zitiert nach [Gaul02], „Verfehlen der Anforderungen an Performance“ bei Jalote zitiert nach [Gaul02], benennen den konkreten Qualitätsaspekt der „Effizienz“ [Bal98]. Auch [Wal01] betrachtet die Komplexität der Anforderungen als kritischen Faktor für Softwareprojekte, wobei die technischen Anforderungen nicht ausgeschlossen werden können.

Wenn beispielsweise eine Anwendung nicht die erforderlichen Antwort- und Durchlaufzeiten erbringen kann, unterstützt sie ihre Benutzer in der Arbeit nicht optimal, weil durch die erhöhten Wartezeiten der Arbeitsfluß behindert wird und Gedankengänge des Benutzers verloren gehen.

Funktionalität - Belege aus der Literatur. [Kel94] führt bei ihrer Darstellung wichtiger Projektkennzeichen auch „Funktionalität“ im Sinne von Funktionstreue an. Auch diese ist als etabliertes Qualitätskriterium „Funktionalität“, speziell „Richtigkeit“ [Bal98] aus der Literatur ersichtlich. Wenn Software nicht die wichtigsten der geforderten Funktionen unterstützt, ist sie für die Anwender keine Arbeitserleichterung und erbringt nicht den ihr zugedachten Nutzen. Dann sind aber die in das Projekt investierten Ressourcen vergeudet und somit der Projekterfolg in Gefahr.
Auch weitere Risikofaktoren aus der Literatur wie „Vernachlässigung von Analyse und Design“ bei McConnel zitiert nach [Gaul02] weisen in die Richtung der Produktqualität, da Analyse und Design kein Selbstzweck sind, sondern der Implementierung eines bedarfsgerechten und strukturierten Softwareproduktes vorarbeiten. Analyse und Design sind beispielsweise hilfreich, den Qualitätskriterien „Funktionalität“, speziell „Richtigkeit“ [Bal98] im Sinne von Funktionstreue und „Änderbarkeit“, speziell „Modifizierbarkeit“ [Bal98] besser zu entsprechen.
Übertragbarkeit - Belege aus der Literatur. Risikofaktoren aus der Literatur wie „Vernachlässigung von Analyse und Design“ bei McConnel zitiert nach [Gaul02] enthalten einen Hinweis auf das Qualitätskriterium der „Übertragbarkeit“ [Bal98] da sie entsprechende Schichtentrennungen unterstützen, welche die für den Einsatz in unterschiedlichen Umgebungen erforderliche Plattformunabhängigkeit leichter erzielen läßt.
Änderbarkeit - Belege aus der Literatur. Mit „Oberflächliche und mangelhafte Entwicklungs- und Produktdokumentation“ [Gru01] gibt ein Risikofaktor aus der Literatur einen Hinweis auf das Qualitätskriterium „Änderbarkeit“, und speziell auf „Modifizierbarkeit“ [Bal98]. Diese Qualitätskriterien sind durch eine unzureichende Dokumentation nicht erfüllt, da durch diese ein Wartungsentwickler die Funktionsweise der Software nicht aus der Dokumentation verstehen kann und sich diese unter Umständen allein aus dem Quellcode herauslesen muß, was ein zeitaufwendiges und fehleranfälliges Unterfangen darstellt. Dies wird auch von dem Risikofaktor wie „Vernachlässigung von Analyse und Design“ bei McConnel zitiert nach [Gaul02] adressiert, da diese die strukturelle Klarheit eines Softwaresystems unterstützen, welche Voraussetzung für „Änderbarkeit“ ist.
Zuverlässigkeit - Belege aus der Literatur. Die „Kritikalität des IT-Systems“ [Gaul04] wird als Projektkenngröße angeführt, die Auswirkungen auf die Qualitätsanforderungen hinsichtlich der „Zuverlässigkeit“, speziell „Reife“, „Fehlertoleranz“ und „Wiederherstellbarkeit“ [Bal98] hat. Beim V-Modell XT [VMX06] wird im Rahmen des Tailoring Mechanismus ein Projektmerkmal „Safety und Security“ abgefragt, um eine Einschätzung über die Kritikalität des zu erstellenden Softwaresystems zu gewinnen.
Projekte unterscheiden sich nach der Kritikalität ihres Entwicklungsergebnisses. Eine Web-Anwendung mit Verbraucherinformationen ist anderen Anforderungen bezüglich der Zuverlässigkeit ausgesetzt als die Steuerung eines Kernreaktors in einem Atomkraftwerk, da der Schaden im Falle des Fehlverhaltens bei der ersten aus geringen Unannehmlichkeiten besteht, bei der zweiten aber nicht mehr quantifizierbar ist, da Menschenleben in Gefahr sind.
Benutzbarkeit - Belege aus der Literatur. Die „Benutzerschnittstelle“ wird im V-Modell XT als Projektmerkmal für den Tailoring Mechanismus gesehen. Hierbei wird abgefragt, ob „die Benutzerschnittstelle ein Erfolgskriterium“ [VMX06] ist. Dies ist ein Hinweis auf die Gewichtigkeit der Benutzerschnittstelle für das Qualitätskriterium „Benutzbarkeit“ [Bal98].

„Oberflächliche und mangelhafte Entwicklungs- und Produktdokumentation“ [Gru01] ist ein Risikofaktor aus der Literatur mit einem Hinweis auf das Qualitätskriterium „Benutzbarkeit“, speziell auf „Verständlichkeit“, „Erlernbarkeit“ und „Bedienbarkeit“ [Bal98]. Diese Qualitätskriterien sind durch eine unzureichende Dokumentation nicht erfüllt, da durch diese ein Benutzer die Funktionsweise der Software nicht aus der Dokumentation verstehen kann und weitere Informationsquellen bemühen muß, was unnötigen Zusatzaufwand verursacht.
Bei einer dialogorientierten Anwendung ist es beispielsweise entscheidend für die Akzeptanz der Benutzer, ob die Oberfläche intuitiv bedienbar ist, weil sie sonst weniger zur Arbeitserleichterung ihrer Anwender beiträgt. Die Benutzeroberfläche ist neben der Antwortzeit eines Softwaresystems Qualitätsaspekte, welche Endbenutzer tatsächlich sehen und beurteilen können. Daher trägt ihre Eignung entscheidend zur Urteilsbildung der Benutzer, damit zur Akzeptanz des Entwicklungsproduktes und schlußendlich zum Projekterfolg bei.

2.2.2.2 Stabilität und Transparenz technischer Anforderungen

Definition des Merkmals *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien:*
Hierunter wird die Gesamtheit der im letzten Abschnitt definierten Qualitätskriterien verstanden, an denen die zu erstellende Software gemessen wird und die bei der Erfüllung der fachlichen Anforderungen nicht verletzt werden dürfen. Im Bedarfsfall kann bei der Anwendung des EVGM projektspezifisch nach einzelnen, besonders signifikanten Qualitätskriterien diversifiziert werden. Qualitätsanforderungen führen auf technische Anforderungen, welche diese konkretisieren. Stabilität steht für die Beständigkeit gegen unerwünschte Veränderungen, Transparenz für Erkennbarkeit und Durchschaubarkeit (siehe „Glossar“ im Anhang).
Abgrenzung zur *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien:* Dort besteht die Schwierigkeit in Menge der Qualitätsanforderungen und ihren Abhängigkeiten, hier in ihrer Erkennbarkeit und Beständigkeit.
Beispiel: Falls die Anforderung an die zu erstellende Software, wiederverwendbar zu sein nicht rechtzeitig bekannt wird, ist die Transparenz dieses Qualitätskriteriums nicht gegeben.

Motivation des Merkmals *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien*:
Bedeutung für Softwareprojekte Falls die technischen Anforderungen nicht hinreichend stabil sind, besteht ein hohes Risiko, daß diese durch das entstehende Projektprodukt nicht erfüllt werden. Mit der Nichterfüllung der Qualitätskriterien ist der Projektnutzen verringert und hierdurch sind die investierten Ressourcen für das Projekt ineffizient eingesetzt. Während fachliche Funktionen wechselseitige Abhängigkeiten besitzen, weisen Qualitätskriterien neben diesen auch meist einen Querschnittscharakter auf, der unter Umständen das ganze bestehende System Gegenstand der Änderungen werden läßt.
Falls die technischen Anforderungen nicht hinreichend transparent für das Projektteam sind, besteht ein hohes Risiko, daß diese durch das entstehende Projektprodukt verletzt werden und somit der Projektnutzen geschmälert ist. Hierdurch sind die investierten Ressourcen für das Projekt ineffizient eingesetzt.
Zur Veranschaulichung dieses Sachverhaltes kann folgende Argumentation ausgeführt

werden. Wenn in einem Softwareprojekt die gegebenen technischen Möglichkeiten und Machbarkeiten nicht hinreichend transparent sind, können sich vereinbarte Projektziele im Projektverlauf als nicht erreichbar herausstellen. Wenn Projektziele, an denen bereits gearbeitet wurde sich als nicht erreichbar herausstellen, sind die bis dahin bereits für sie eingesetzten Projektressourcen kalendarische Zeit und Projektbudget verschwendet.
Unter Umständen kann eine Applikation nur mit Antwortzeiten sinnvoll eingesetzt werden kann, die unter den gegebenen Rahmenbedingungen nicht erzielbar sind. Die Transparenz hierüber schafft eine Entscheidungsgrundlage über die Durchführung beziehungsweise Einstellung des Projektes und kann so Fehlinvestitionen verhindern.
Falls die Antwortzeitanforderungen zu einem späten Zeitpunkt im Projekt stark verschärft werden, kann dies die Grenzen des technisch Machbaren sprengen und den Projekterfolg gefährden.
Dies verdeutlicht die Transparenz bezüglich der Realisierbarkeit von technischen Anforderungen und Qualitätskriterien, sowie deren Stabilität als bedeutendes Merkmal zur Charakterisierung von Projekten.
Beispiel: Wenn sich Qualitätsanforderungen von „Änderbarkeit“ auf „Laufzeit- und Speicherplatz-Effizienz“ ändert, wie sie bei eingebetteten Systemen oft im Vordergrund steht, besteht Instabilität. Performance-Steigerungen können mitunter nur sehr aufwendig, nicht zu jedem beliebigen Zeitpunkt und nicht in jedem beliebigen Ausmaß erreicht werden. Diese Anforderungsänderung kann das gesamte System betreffen.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle adressieren Instabilität und Intransparenz von Qualitätskriterien unterschiedlich umfassend und detailliert. Dies geschieht innerhalb von Disziplinen wie Qualitätssicherung, Qualitätsmanagement, Requirements Engineering und Requirements Management. Etablierte Ansätze führen eine umfassendere und methodischere Analyse durch, um der Intransparenz und der Instabilität zu begegnen. Die Agilen Ansätze bauen auf redundanzfreie und deshalb leichter änderbare Arbeitsproduktmodelle.
Beispiel: Das V-Model XT [VMX06] fragt in seinem Arbeitsprodukttemplate *Anforderungen (Lastenheft)* im Kapitel „Nicht funktionale Anforderungen“ ausdrücklich nach den technischen Anforderungen. Hiermit wird die diesbezügliche Transparenz erhöht. Extreme Programming hingegen erhebt nicht explizit die Qualitätskriterien [Bec00]. Es besteht die Möglichkeit, diese auf den Story Cards zu vermerken. Im Extreme Programming können mit der Technik „Refactoring“ [Bec00], [Fow99] instabile technische Anforderungen bedingt adressiert werden. Agile Methoden sind allerdings insgesamt eher an funktionalen fachlichen Anforderungen ausgerichtet und wenig für nichtfunktionale Anforderungen geeignet [Boe04]. Dem kann hinzugefügt werden, daß sich die Agilen Methoden bei der Erhebung der Anforderungen und der Qualitätssicherung sehr stark an für die Benutzer erkennbaren Kriterien orientieren. Dadurch treten innere Qualitätsmerkmale in den Hintergrund (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Schlechte Stabilität und Transparenz technischer Anforderungen und Qualitätskriterien (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien)* vergeudet Projektressourcen Personal, Budget und Zeit. Fortwährende Überarbeitung von Arbeitsprodukten kostet Aufwand und die Fertigstellung der Software verzögert sich. Dies verschlechtert den Zustand der *EPP-BedDeck-Kap-Personell*, der *EPP-BedDeck-Kap-Projektbudget* und der *EPP-BedDeck-Kap-KalendarischeZeit*. Aufgrund des Querschnittscharakters technischer Anforderungen und Qualitätskriterien wird dieser Einfluß als überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.

Schlechte *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien* stellt erhöhte Anforderungen an die Fähigkeiten im Software Engineering (*EPSWE-BedDeck-Fähigk-SoftwareEngineering*), weil sich ihre Auswirkungen mitunter nicht auf Teile des entstehenden Systems begrenzen lassen und somit schwierig zu handhaben sind. Auch dieser Einfluß wird aufgrund des Querschnittscharakters technischer Anforderungen und Qualitätskriterien als überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Darüber hinaus vermindert fehlende *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien* Erfolgserlebnisse in Form von im Einsatz befindlicher Software und verschlechtert somit die Motivation der Projektmitarbeiter (*EPM-BedDeck-MotivationImTeam)*. Dieser Einfluß wird überproportional eingeschätzt und wird daher mit „≈≈“ vorgeschlagen.
Auf die *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien* wird im Rahmen der Unterstützung des Requirements Engineering (*EV-UnterstProjTheBer-RequirementsEngineering)*, der Unterstützung des Software Engineering (*EV-UnterstProj-TheBer-SoftwareEngineering)* und der Unterstützung des Software Management (*EV-UnterstProjTheBer-SoftwareManagement)* eingegangen. Auch ein flexibler Entwicklungsansatz, der iterativ und inkrementell strukturiert ist (*EV-Flexib-DurchIterativität-UndInkrementalität*), hilft bei der Beherrschung von instabilen und intransparenten technischen Anforderungen und Qualitätskriterien (siehe Kapitel 3.2).

Belege aus der Literatur

Als Risikofaktoren für Softwareprojekte werden in der Literatur „Überehrgeizige Projektziele“, bei KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul02], und „Utopische Benutzererwartungen“ [Gru01], genannt. Der Chaos Report empfiehlt als Projekterfolgskriterium „Realistic Expectations“ [Sta94]. Dies kann man einerseits auf die fachlichen Anforderungen und andererseits auf die technischen Anforderungen und Qualitätskriterien beziehen (siehe oben).

2.2.3 Projektthemenbereich Projektmanagement

Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Produkt“ und des Projektthemenbereichs „Projektmanagement“ zeigt der folgende Ontologieausschnitt in Abbildung 14.

2.2 Merkmale des Risikobereichs Entwicklungs-Produkt

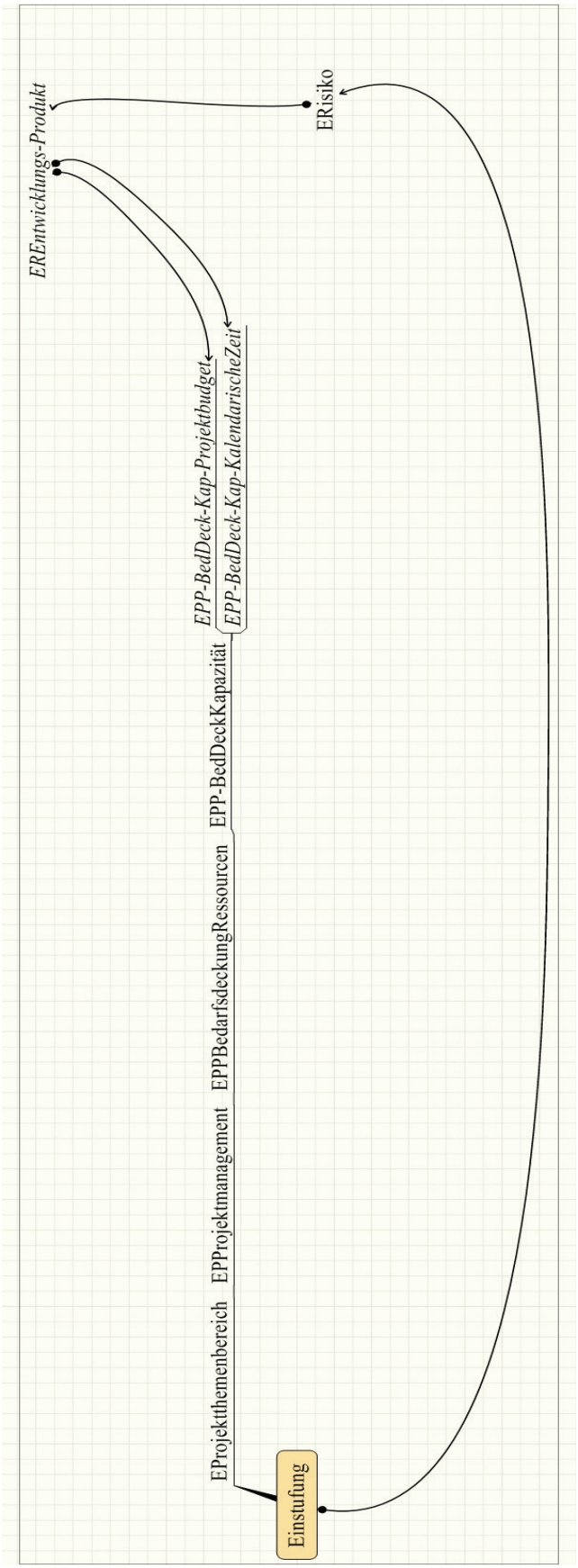

Abbildung 14 Merkmale Risikobereichs Entwicklungs-Produkt und Projektthemenber. Projektman.

2.2.3.1 Kalendarische Zeit

Definition des Merkmals *EPP-BedDeck-Kap-KalendarischeZeit*:
Diese steht für das Verhältnis von verfügbarer und erforderlicher Zeit in einem Projekt.
Beispiel: Wenn in einem Projekt ein geforderter Funktionsumfang mit den verfügbaren personellen Kapazitäten in der geforderten Zeit nicht erbracht werden kann, liegt eine Bedarfsunterdeckung vor.

Motivation des Merkmals *EPP-BedDeck-Kap-KalendarischeZeit:*
Bedeutung für Softwareprojekte. Ein Überschreiten des Zeitrahmens kann in einem Projekt bedeuten, daß eine Software wertlos wird oder daß zumindest größerer Schaden durch die nicht gegebene Verfügbarkeit der Software entsteht. Das Verhältnis von verfügbarer und benötigter Zeit hat entscheidenden Einfluß auf Vorgehensweise im Projekt wie die gesamte Projekt- und Kapazitätsplanung.
Beispiel: Die Projekte zur Erweiterung der Stelligkeit der Darstellung der Jahreszahlen vor dem Jahr 2000, zur Umstellung auf den Euro, sowie für das Maut-Abrechnungssystem Toll Collect können dies veranschaulichen. Bei einer Terminverfehlung in einem Jahr-2000-Projekt liefen Unternehmen Gefahr, Jahreszahlen nicht mehr eindeutig darstellen zu können und somit Vorgänge nicht mehr eindeutig datieren zu können. Bei Euro-Umstellungsprojekten hätte eine Terminverletzung mitunter bedeutet, finanzielle Buchungen nicht mit der gültigen Währung und Wertigkeit verbuchen zu können, wodurch unternehmensübergreifende Geldtransfers nicht mehr möglich gewesen wären. Das Erliegen der Zahlungsströme bedeutet aber für viele Unternehmen kurzfristig den Ruin. Im Falle des Toll Collect Projektes bedeutete die Terminverfehlung fehlende Staatseinnahmen in Milliardenhöhe.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen Vorgehensmodelle unterstützen in ganz unterschiedlicher Weise den Umgang mit Zeitengpässen im Projekt. Dies erfolgt bei den etablierten Ansätzen wie dem V-Modell XT [VMX06] mit Disziplinen des kaufmännischen und organisatorischen Projektmanagements durch die Planung, Kontrolle und Steuerung von Terminen und Arbeitsprodukten. Ein gezieltes Requirements Management mit dem Ziel, Funktionalität zu priorisieren und so einen Zeitengpass kompensieren zu können sind möglich. Die Agilen Ansätze arbeiten hier stärker mit Reduzierung der Projekt-Arbeitsprodukte auf Code und mit Iterativität bezogen auf die jeweils am höchsten priorisierten Anforderungen. In der speziellen Ausprägung von Iterativitiät, dem „Time Boxing“ wird eine Zeitspanne von wenigen Wochen fixiert, für die ein Funktionsumfang fixiert und implementiert wird. Am Ende der Time Box muß in jedem Fall Software ausgeliefert werden, Funktionalität wird im Zweifelsfall weggelassen.
Beispiel: Scrum [Sch02] optimiert einen im Projekt bestehenden Zeitengpaß mit dem *Sprint*, einer 30-tägigen Time Box (siehe Glossar im Anhang), in der ausschließlich eine hierfür priorisierte, definierte und für diesen Zeitraum fixierte Funktionalität entwickelt wird (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen
Eine Verschlechterung des Verhältnisses von verfügbarer zu erforderlicher kalendarischer Zeit erfordert mehr oder befähigtere Mitarbeiter. Das verschlechtert den Zustand der *EPP-BedDeck-Kap-Personell.* Da verknappte Termine nur bedingt mit Personalaufstockung kompensiert werden können, wird dieser Einfluß proportional eingeschätzt und mit „≈“ vorgeschlagen.

Der Zustand der Bedarfsdeckung Software Engineering Fähigkeiten (*EPSWE-BedDeck-Fähigk-SoftwareEngineering)* und der Bedarfsdeckung Software Management Fähigkeiten (*EPSWE-BedDeck-Fähigk-SoftwareManagement)* verschlechtert sich ebenfalls. Terminprobleme stellen erheblich erhöhte Anforderungen an die Fähigkeiten der Mitarbeiter, die in der verknappten Zeit möglichst effizient und effektiv arbeiten müssen. Daher werden diese beiden Beziehungen überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Eine verschärfte Terminsituation verschlechtert die Möglichkeit, Qualität zu erzeugen, oder verringert den erbringbaren Funktionsumfang. Die Einfachheit des Umfangs der technischen Anforderungen und Qualitätskriterien (*EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien)* oder die Einfachheit durch den Umfang der Funktionen (*EPD-Einf-DurchUmfangFunktionen)* kann sich daher durch Qualitäts- und Funktionsreduktion in Reaktion auf eine verkürzte Projektlaufzeit verbessern. Dies ist allerdings nur begrenzt möglich, daher wird dieser Einfluß proportional und gegenläufig eingeschätzt und mit „1/≈“ vorgeschlagen. Hierfür ist eine Management-Entscheidung erforderlich. Die *EPP-BedDeck-Kap-KalendarischeZeit* wird durch die Unterstützung des Projektmanagements (*EV-UnterstProjTheBer-ProjektManagement)* verbessert (siehe Kapitel 3.2).

Belege aus der Literatur
Die kalendarische Zeit ist eine Projektressource. Projekte sind per Definition zeitlich begrenzt [Kel94]. Die Zeit beziehungsweise der Zeitbedarf im Verhältnis zu ihrer Verfügbarkeit wird immer wieder als Risikofaktor gesehen. Dies untermauern Risikofaktoren wie „Aufgabe der Projektplanung unter Zeitdruck“ bei McConnel zitiert nach [Gaul04] mit der Wirkung einer nicht fundierten und nicht aktualisierten Projektplanung, „Zu zeitaufwendige Realisierung“ in der Computerwoche zitiert nach [Gaul04] als Folgeerscheinung der Risikofaktoren „Unrealistische Zeitschiene“ bei forsa zitiert nach [Gaul04], „Unrealistische Zeitplanung bei Jalote zitiert nach [Gaul04], „Zeitverlust bei der Projektinitiierung“ bei McConnel zitiert nach [Gaul04], “Ehrgeizige Zeitplanung“ bei McConnel zitiert nach [Gaul04], „Unrealistische Zeit- und Budgetpläne“ bei Boehm zitiert nach [Gaul04], „Inhärent fehlerhafter Zeitplan“ [DeM03], „Unrealistic Schedules“ [Hum02], zu optimistischer „Zeitplan“ bei [Hal98] zitiert nach [Wal04], „Zeitliche Verfügbarkeit“ [Wic00], sowie „Unrealistisch kurze Terminvorgaben für die Projektfertigstellung“ [Gru01]. Auch [Wal01] sieht die Zeit als kritische Größe in Softwareprojekten.

2.2.3.2 Projektbudget

Definition des Merkmals *EPP-BedDeck-Kap-Projektbudget*:
Sie beschreibt das Verhältnis von Bedarf und Verfügbarkeit von finanziellen Mitteln für ein Projekt.
Beispiel: Falls in einem Projekt nicht genügend Budget vorhanden ist, um die vereinbarte Leistung zu erbringen, besteht eine Bedarfsunterdeckung.

Motivation des Merkmals *EPP-BedDeck-Kap-Projektbudget*:
Bedeutung für Softwareprojekte. Die Investitionen in industrielle Projekte werden in der Erwartung eines Nutzens, der die Kosten übertrifft, getätigt. Daher kann ein aus dem Ruder laufender Bedarf an der Ressource Geld ein Projekt zum Scheitern bringen, weil sein Ergebnis im unfertigen Zustand keine Chance hat, jemals einen wirt-

schaftlichen Nutzen zu erzielen. Unter Umständen ist Budget gar nicht im ausreichenden Maße beschaffbar, um das Projekt zu Ende zu führen. Wenn in einem Projekt mit einem bisher nur halbfertigen und noch nicht benutzbaren Ergebnis das Geld ausgeht, können die Mitarbeiter nicht mehr bezahlt werden, die daraufhin dem Projekt nicht mehr zur Verfügung stehen können. Das bedeutet zumindest den Projektmißerfolg und hat im schlimmsten Fall Auswirkungen auf den Fortbestand der beteiligten Unternehmen, falls diese existenziell vom Projekterfolg abhängen. Wenn eine Unternehmung ihre finanziellen Reserven aufgebraucht hat und kein marktreifes Produkt vorweisen kann, ist ihr Fortbestand gefährdet.

Beispiel: Wenn bei einer Warenwirtschaftssoftware wegen Budgetmangels nur der Wareneingang, nicht aber der Verbrauch oder Verkauf von Waren implementiert werden kann, unterstützt die Software den Warenwirtschaftsprozeß nicht hinreichend und ist für den Auftraggeber nicht einsetzbar. Die geflossenen Investitionen können sich nicht amortisieren und das Projekt ist gescheitert.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle gehen ganz unterschiedlich mit der Budgetsituation in Projekten um. Dies erfolgt mit Disziplinen wie kaufmännischem und organisatorischem Projektmanagement durch die Planung, Kontrolle und Steuerung von Arbeitsprodukten, Ressourcen und Budget. Ein methodisches Requirements Management hilft bei der Priorisierung von Anforderungen. Somit können die für einen Einsatz erforderlichen und somit am höchsten priorisierten Anforderungen vorrangig umgesetzt werden. Diese Priorisierung hilft im Rahmen der vorher festgesetzten Kosten die wichtigste Funktionalität umzusetzen und somit einem von Anfang an bestehenden Budgetengpaß zu begegnen. Etablierte Ansätze wie das V-Modell XT [VMX06] setzen hier auf geplantes methodische Vorgehen, Agile Ansätze wie Extreme Programming auf Minimierung von Overhead und damit von Aufwänden.

Beispiel: Durch die radikale Devise, Arbeitsaufwand zu minimieren und nur die essentiell wichtigen Tätigkeiten auszuführen, hilft Extreme Programming [Bec00] einen von Projektbeginn an bestehenden Budgetengpaß zu adressieren. Das Budget wird im V-Modell XT [VMX06] im Rahmen des Vorgehensbausteins *Kaufmännisches Projektmanagement* betrachtet, die Kompensation einer bestehenden Engpaßsituation steht aber nicht im Vordergrund (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Eine Verschlechterung der Budgetsituation (*EPP-BedDeck-Kap-Projektbudget)* gebietet es mitunter, Personal einzusparen und verschlechtert so unter Umständen die *EPP-BedDeck-Kap-Personell*. Diese Auswirkung wird proportional eingeschätzt und mit „≈“ vorgeschlagen.

Möglicherweise kann eine Verschlechterung der Qualität oder eine Verminderung des Funktionsumfangs in Kauf genommen werden, was die *EPD-Einf-DurchUmfangFunktionen* und die *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien* verbessert. Da dies nur bedingt möglich ist, wird der Einfluß proportional und gegenläufig eingeschätzt und mit „1/≈“ vorgeschlagen. Für alle diese Beeinflussungen sind Managemententscheidungen erforderlich. Die *EPP-BedDeck-Kap-Projektbudget* wird durch die Unterstützung des Projektmanagements (*EV-UnterstProjTheBer-ProjektManagement)* verbessert (siehe Kapitel 3.2).

Belege aus der Literatur
Für [Kel94] sind „Kosten“ von entscheidender Bedeutung. Dies wird auch durch „Unrealistische Zeit- und Budgetpläne“ bei Boehm zitiert nach [Gaul04] und „Finanzielle Lage“ im Sinne von Budgetengpässen bei [Plo02] zitiert nach [Wal04] sowie „Finanzierung“ im Sinne von Budgetbeschränkungen bei [Hal98] zitiert nach [Wal04] bestätigt.

2.2.4 Projektthemenbereich Menschenführung

Für den Projektthemenbereich Menschenführung ergeben sich bezüglich des Entwicklungs-Produktes keine hinreichend signifikanten Merkmale.

2.3 Merkmale des Risikobereichs Entwicklungs-Team

Entsprechend der Systematik aus Kapitel 2.1.1 werden im folgenden die Merkmale innerhalb der Projektthemenbereiche Domänenwissen, Softwareentwicklung, Projektmanagement und Menschenführung beschrieben.

2.3.1 Projektthemenbereich Domänenwissen

Fähigkeiten als Projektressource. Als Fähigkeiten können generell Wissen, Kombinationsfähigkeit, Intuition, Antizipation oder Erfahrung aufgefasst werden.
Unter dem Fähigkeitsbegriff sind in der vorliegenden Arbeit Wissen, Fertigkeiten und Erfahrung zusammengefaßt. Fähigkeiten sind von fachlicher, methodischer beziehungsweise technischer Art, sowie bezogen auf das Management.
Die im Projekt verfügbaren Fähigkeiten sind eine wichtige Ressource, weil ohne sie die Tätigkeiten in den entsprechenden Disziplinen nicht adäquat ausgeführt werden können. Die ausreichende Verfügbarkeit der angeführten Fähigkeiten in einem Projekt begünstigen einen Projekterfolg.
Die Wichtigkeit von Fähigkeiten als Projektressource wird in der Fachliteratur betont. Gaulke [Gau02], [Gau04] benennt aus diversen Studien Erfolgs-und Risikofaktoren für Projekte. „Falsche Besetzung des Projektteams“ bei der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul02] ist ein Risikofaktor.
Hierzu muß genauer ausgeführt werden, daß bei einer falschen Besetzung des Projektteams die im konkreten Projekt erforderlichen Fähigkeiten nicht im ausreichenden Maß verfügbar sind. Dies wiederum erschwert eine adäquate Lösung der Projektaufgaben auf effektive und effiziente Weise, wodurch wiederum die aufgewendeten Projektressourcen kalendarische Zeit, Projektbudget und personelle Kapazitäten nicht optimal in Projektergebnisse umgesetzt werden. Auf die Bedeutsamkeit der Fähigkeiten der Projektmitarbeiter weisen auch „Mangelnde Ausbildung“ [Ver00], „Ressourcen“ [Wic00] im Sinne qualifizierter Mitarbeiter, „geringe Arbeitsleistung“ [DeM03] und „unfähige Mitarbeiter“ [Kel94] hin. Die Wichtigkeit ihres adäquaten Einsatzes betonen „ungünstige Teambesetzung bzw. Aufgabenverteilung“ bei Computerwoche zitiert nach [Gaul02], „Personelle Fehlbesetzungen“ bei forsa zitiert nach [Gaul04] und „inappropriate staffing“ [Hum02]. „Mitarbeiterfluktuation“ [DeM03] weist auf die Folge des Know-how-Verlustes hin. In der Chaos Studie wird eine „competent Staff“ [Sta94] empfohlen, und [Bro95] fordert “fewer better people“. In [DeM01] werden Menschen und ihr Wissen als wichtigstes Unternehmenskapital bezeichnet. Die verfügbaren Fähigkeiten in einem Projekt durch das Wissen und Können der Mitarbeiter können nicht beliebig schnell durch Personalaufstockung und Training gesteigert werden. Sie können aber andererseits durch Fluktuation dem Projekt leicht verloren gehen (siehe auch [Boe04]).

Ein „Einheitliches Grundverständnis über Fähigkeiten von Unternehmen und Organisation“ wird bei [ISA00] zitiert nach [Wal04] als kritischer Erfolgsfaktor des Projektmanagements gesehen.
Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Team“ und des Projektthemenbereichs „Domänenwissen“ zeigt der folgende Ontologieausschnitt in Abbildung 15.

2.3 Merkmale des Risikobereichs Entwicklungs-Team

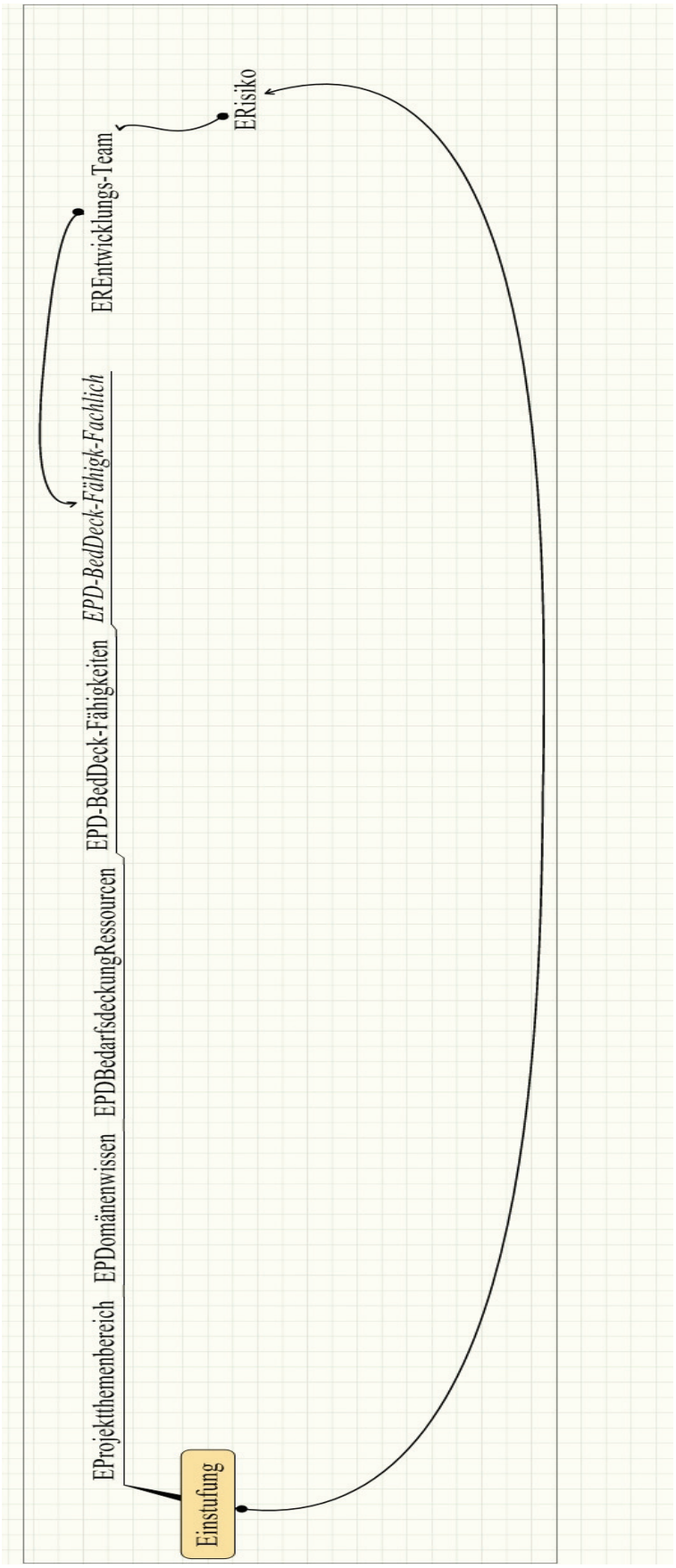

Abbildung 15 Merkmale Risikobereich Entwicklungs-Team und Projektthemenbereich Domänenwissen

2.3.1.1 Fachliche Fähigkeiten

Definition des Merkmals *EPD-BedDeck-Fähigk-Fachlich:*
Dieses Merkmal umfasst Wissen und Erfahrung bezüglich der fachlichen Domäne eines Softwareentwicklungsprojektes. Bedarfsdeckung steht für das Verhältnis erforderlicher und verfügbarer fachlicher Fähigkeiten.
Beispiel: Die Kenntnis fachlicher Ziele, Anforderungen, Sachverhalte, Zusammenhänge, Prinzipien, Regelungen, Verordnungen, Gesetze und Vereinbarungen, sowie die Fähigkeit zur fachlichen Antizipation, Kombination und Intuition bezüglich fachlicher Sachverhalte, Zusammenhänge und Tendenzen fallen hierunter. Wenn in einem Projekt die erforderlichen fachlichen Fähigkeiten nicht verfügbar sind, liegt eine mangelnde Bedarfsdeckung vor.

Motivation des Merkmals *EPD-BedDeck-Fähigk-Fachlich:*
Bedeutung für Softwareprojekte. Ohne verfügbare fachliche Fähigkeiten im Projekt können keine fachlichen Anforderungen ermittelt und spezifiziert werden.
Beispiel: Für die Entwicklung eines Buchhaltungssystems sind allgemeine Kenntnisse der Buchführung und des Rechnungswesens, sowie des organisationsspezifischen Kontenrahmens erforderlich. Wenn in einem Projekt zur Entwicklung einer Buchungssoftware Entwickler eingesetzt werden, die keinerlei kaufmännischen Wissenshintergrund vorweisen können, sind Kommunikationsprobleme und Mißverständnisse beim Requirements Engineering mit den Fachexperten abzusehen. Dies führt auf ineffiziente Arbeit und damit auf die Verschwendung der Projektressourcen *EPP-BedDeck-Kap-KalendarischeZeit*, *EPP-BedDeck-Kap-Projektbudget* und *EPP-BedDeck-Kap-Personell*, wodurch der Projekterfolg gefährdet ist.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle ergänzen fachliche Fähigkeiten mit den Disziplinen Requirements Engineering und Management. Sie beeinflussen dieses Merkmal sehr unterschiedlich. Das V-Modell XT [VMX06] bietet eine umfangreiche Beschreibung der Anforderungsanalyse, während Extreme Programming [Bec00] vergleichsweise knappe Beschreibungen enthält.
Beispiel: Das V-Modell XT [VMX06] bietet den Vorgehensbaustein *Anforderungsfestlegung*, Extreme Programming [Bec00] fordert den *on-site customer*, um fachliches Wissen in das Projekt einzubringen (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen
Je höher die Verfügbarkeit von fachlichen Fähigkeiten (*EPD-BedDeck-Fähigk-Fachlich)* im Projekt ist, desto höher ist auch die Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)*, weil Wissensdefizite bezüglich der Problemdomäne im Entwicklungsteam schneller beseitigt und Änderungen von Anforderungen besser vorhergesehen werden können. In der *EPD-StaTrans-FachlicheAnforderungen* spiegelt sich oft der Prozeß der fachlichen Erkenntnisgewinnung während der gesamten Projektlaufzeit wieder. Mangelnde Fähigkeiten der fachlichen Generalisierung und Antizipation schlagen sich oft in einer Verschlechterung der *EPD-StaTrans-FachlicheAnforderungen* nieder. Die Stärke der Beziehung wird überproportional eingeschätzt und deshalb mit „≈≈“ vorgeschlagen.
Eine gute *EPD-BedDeck-Fähigk-Fachlich* hilft, frühzeitig eine gute Basis gemeinsamen Problemverständnisses als verlässliche Basis für eine Weiterarbeit mit effizientem Einsatz der Projektressourcen Personal, Budget und Zeit aufzubauen. Sie verbessert so die *EPP-BedDeck-Kap-Personell, EPP-BedDeck-Kap-Projektbudget* und *EPP-BedDeck-*

Kap-KalendarischeZeit. Diese Einflüsse werden proportional eingeschätzt und jeweils mit „≈“ vorgeschlagen.
Komplizierte Tätigkeiten zur Erschliessung fachlichen Wissens können durch Aktivitätsmodelle unterstützt werden, komplizierte Strukturen zu seiner Dokumentation mit Arbeitsproduktmodellen. Dies kann im Rahmen der Unterstützung des Requirements Engineering (*EV-UnterstProjTheBer-RequirementsEngineering)* erfolgen (siehe Kapitel 3.2).

Belege aus der Literatur

Auf den spezifischen Projektrisikofaktor des fachlichen Wissens weisen „Unzureichendes Fachwissen“ bei Jalote zitiert nach [Gaul04] und „Unzureichende Fachkompetenz der Teammitglieder“ [Gru01] hin. Gaulke stellt „Erfahrung und Sachkenntnis“ [Gaul02] als wichtigen Projekterfolgsfaktor dar. Dies wird auch von [Wal01] bestätigt.

2.3.2 Projektthemenbereich Softwareentwicklung

Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Team“ und des Projektthemenbereichs „Softwareentwicklung“ zeigt der folgende Ontologieausschnitt in Abbildung 16.

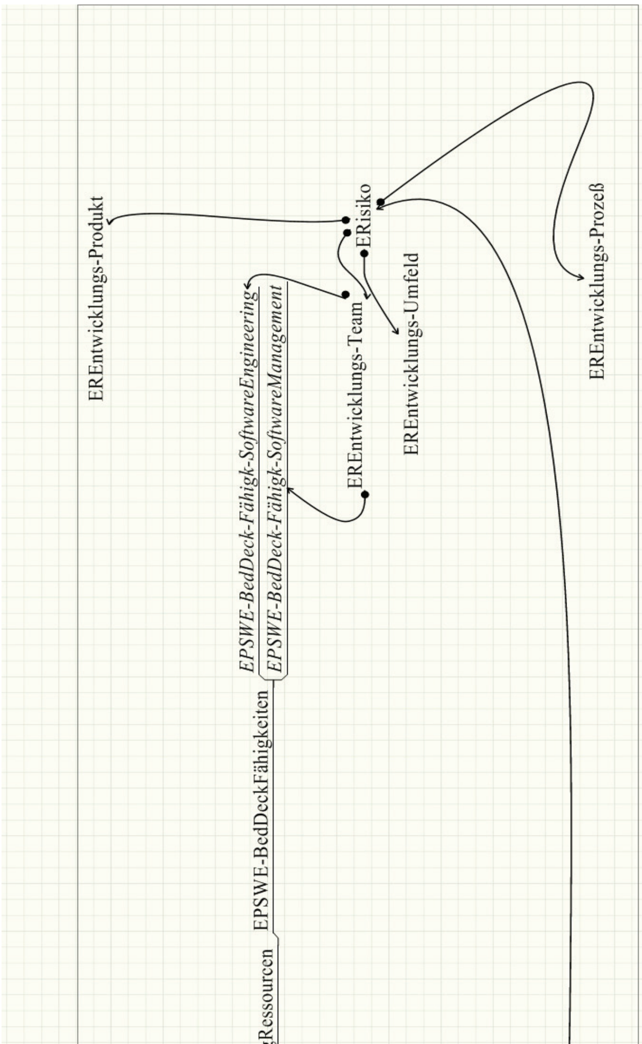

Abbildung 16 Merkmale Risikobereich Entwicklungs-Team und Projektthemenbereich Softwareentw.

2.3.2.1 Software Engineering Fähigkeiten

Definition des Merkmals *EPSWE-BedDeck-Fähigk-SoftwareEngineering*:
Software Engineering Fähigkeiten umfassen Wissen, Erfahrung und praktische Fertigkeiten über die Techniken, Paradigmen und Methoden der Softwaretechnik, die direkt beschreiben, wie Entwicklungs-Arbeitsprodukte im Rahmen von methodischen Entwicklungsschritten zu erzeugen, zu verändern, darzustellen und zu überprüfen sind. In der vorliegenden Arbeit umfassen die Software Engineering Fähigkeiten die Disziplinen Requirements Engineering, Analyse, Design, Implementierung und Qualitätssicherung. Bedarfsdeckung steht hier für die bedarfsgerechte Verfügbarkeit von Software Engineering Fähigkeiten.
Beispiel: Wenn Fähigkeiten bezüglich der Qualitätssicherung verfügbar sind, während ein Spezialist für Architektur und Design gebraucht wird, ist diesbezüglich keine Bedarfsdeckung gegeben.

Motivation des Merkmals *EPSWE-BedDeck-Fähigk-SoftwareEngineering*:
Bedeutung für Softwareprojekte. Ohne Software Engineering Fähigkeiten kann die eigentliche Entwicklungsarbeit nicht durchgeführt werden. Softwareentwicklung ist eine Tätigkeit, bei der vergleichsweise Neues geschaffen wird, bei der jedoch immer wieder Muster hinsichtlich Tätigkeiten und Arbeitsprodukten zu beobachten sind. Hier muß verdeutlicht werden, daß die industrielle Softwareentwicklung eine zu komplizierte Tätigkeit ist, um für Laien intuitiv zugänglich zu sein. Deshalb sind entsprechende methodische Ansätze aus der Fach- und Lehrmeinung von großer Bedeutung für den Erfolg eines Softwareprojektes.
Funktionalitäten zum Melden und Protokollieren von Fehlern in Softwaresystemen weisen konzeptionelle Ähnlichkeiten auf. Bei entsprechender Erfahrung in der Softwareentwicklung kann ein Entwickler mit größerer Wahrscheinlichkeit ein aktuelles Problem in einem Projekt schneller bearbeiten, weil er ein ähnliches Problem früher schon einmal gelöst hat. Die Verfügbarkeit der Ressource Software Engineering Erfahrung ermöglicht einen effizienteren Umgang mit den Ressourcen Zeit, Geld und personelle Kapazitäten und ist somit förderlich für den Projekterfolg.
Beispiel: Ohne Software Engineering Fähigkeiten können keine Modelle, keine Testfälle und kein Code erstellt werden.
Ein Datenbankspezialist, der in einem Projekt die Aufgabe erhält, grafische Benutzeroberflächen zu entwickeln, kann seine Fähigkeiten nicht optimal in das Projekt einbringen. Wenn die paradigmatische Basis der Entwicklungsarbeit für das Team neu ist, müssen erhebliche Schulungs- und Coachingaufwände kalkuliert werden, bevor das Team effektiv arbeiten kann. Als Zeitbedarf für die Umstellung auf Objektorientierung wurde in den neunziger Jahren ein Erfahrungswert von zwei Jahren gehandelt.
Der Wissensbereich Software Engineering ist so groß, daß er von niemandem umfassend und detailliert abgedeckt werden kann. Da der Bedarf an spezifischen Software Engineering Fähigkeiten sehr stark vom Projekt abhängt, muß hier betont werden, daß es sich bei diesem Merkmal immer um bedarfsgerechte Software Engineering Fähigkeiten handelt. Dies bezieht sich auf spezifische Fähigkeiten und Fähigkeitsgrade.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen Software Engineering Fähigkeiten ganz unterschiedlich.
Der Rational Unified Process [RUP06] unterstützt Software Engineering Fähigkeiten umfassend, während Scrum [Sch02] diesen Themenbereich ausspart.
Beispiel: Der Rational Unified Process [RUP06] beschreibt die Prozesse *Require-*

ments, *Analysis&Design*, *Implementation* und *Test*, das V-Modell XT [VMX06] unterstützt die Bedarfsdeckung der Software Management Fähigkeiten mit Vorgehensbausteinen wie *SW-Entwicklung* und *Qualitätssicherung,* während Scrum [Sch02] diese Fähigkeiten bei seinen Anwendern voraussetzt (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Verbesserte Fähigkeiten im Software Engineering (*EPSWE-BedDeck-Fähigk-SoftwareEngineering)* bewirken eine bessere Handhabung von technischen Anforderungen und erhöhen die *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien*. Diese Beeinflussung wird überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Die *EPSWE-BedDeck-Fähigk-SoftwareEngineering* ermöglicht effizienteren Einsatz der Projektressourcen Zeit, Budget und Personal auf qualifizierter Basis und verbessert somit die *EPP-BedDeck-Kap-KalendarischeZeit*, die *EPP-BedDeck-Kap-Projektbudget* und die *EPP-BedDeck-Kap-Personell*. Diese Einflüsse werden proportional eingeschätzt, werden also mit „≈“ vorgeschlagen.
Speziell durch den Requirements Engineering Anteil dieses Merkmals verbessert sich bei einer Erhöhung der *EPSWE-BedDeck-Fähigk-SoftwareEngineering* auch die Handhabbarkeit eines großen Funktionsumfangs *(EPD-Einf-DurchUmfangFunktionen),* der Gewinnung fachlicher Fähigkeiten (*EPD-BedDeck-Fähigk-Fachlich*) und auch den Umgang mit instabilen und intransparenten fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen*). Diese Beeinflussungen werden proportional eingeschätzt und mit „≈“ vorgeschlagen.
Komplizierte Tätigkeiten im Software Engineering können durch Aktivitätsmodelle unterstützt werden, komplizierte Strukturen zu Dokumentation von deren Resultaten mit Arbeitsproduktmodellen. Dies kann im Rahmen der Unterstützung des Software Engineering (*EV-UnterstProjTheBer-SoftwareEngineering)* und der Unterstützung des Requirements Engineering (*EV-UnterstProjTheBer-RequirementsEngineering)* erfolgen (siehe Kapitel 3.2).

Belege aus der Literatur

„Mangel erfahrener Mitarbeiter im Projektteam“ bei der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04] ist von Bedeutung für den Projekterfolg. „Unzureichende Softwareentwicklungsmethoden“ bei der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04] und „Mangel an technisch versierten Mitarbeitern“ bei Jalote zitiert nach [Gaul02], aber auch „Unzureichende Architektur, Performance, Qualität“ bedingt durch „Überschätzung der eigenen Informatikfähigkeiten“ bei [Boe98] zitiert nach [Wal04 sind weitere Risikofaktoren in Softwareentwicklungsprojekten. Bei [Plo02] zitiert nach [Wal04] wird „Falsche Wahl der Projektressourcen“, bei [Hal98] zitiert nach [Wal04] nicht hinreichende „Mitarbeiter-Fähigkeiten“ als Projektrisikofaktor bezeichnet, was erfahrungsgemäß ebenfalls bezogen auf Software Engineering Fähigkeiten gedeutet werden kann. Einen systematischen Überblick über die Vielfalt relevanter Skills bietet Watts Humphrey mit dem „Personal Software Process“ in [Hum95], [Hum97].
Als Beispiele für relevante Fähigkeiten aus weiteren Quellen können systematisches Testen [Mye95], hinreichendes „Testing“ bei [Hal98] zitiert nach [Wal04], nicht hinreichende „Integration und Test“ [Wal04] methodisches Requirements Engineering [Hoo05], [Rup01] oder ingenieurgemäßer Entwurf eines Softwaresystems [Bal96] und hinreichendes Design [Wal04] genannt werden. [Wal04] betont besonders die Wichtigkeit des Wissens um Nutzen und Entwicklung einer Softwarearchitektur. Für alle diese

Teildisziplinen wird von den Entwicklern das entsprechende Software Engineering Wissen, sowie die Fähigkeiten, dieses im Projekt anzuwenden, benötigt. Software Engineering Wissen und Erfahrung ist also eine Projektressource, ohne die entweder die Qualität des Entwicklungsproduktes, oder der effiziente Umgang mit den Ressourcen Zeit und Geld leiden.

2.3.2.2 Software Management Fähigkeiten

Definition des Merkmals *EPSWE-BedDeck-Fähigk-SoftwareManagement*:
Software Management Fähigkeiten umfassen das Wissen und die praktischen Fertigkeiten, die Entwicklungsschritte des Software Engineering zu überblicken und zu steuern (vergleiche auch [Bal96]). Sie repräsentiert die Kenntnisse, was unter welchen Vorbedingungen im Projekt getan wird, welche Methoden und Techniken wann zum Einsatz kommen und auf welche Art diese angeordnet sind. Software Management verwaltet die Arbeitsprodukte, die durch das Software Engineering erzeugt wurden. Es bildet seinen übergeordneten Rahmen. In der vorliegenden Arbeit umfassen die Software Management Fähigkeiten das Requirements Management, Change Management, Konfigurationsmanagement. Bedarfsdeckung steht hier für die bedarfsgerechte Verfügbarkeit von Software Management Fähigkeiten.
Beispiel: Falls in einem Projekt Fähigkeiten für Konfigurationsmanagement verfügbar sind, während Requirements Management benötigt würde, ist keine hinreichende Bedarfsdeckung gegeben.

Motivation des Merkmals *EPSWE-BedDeck-Fähigk-SoftwareManagement:*
Bedeutung für Softwareprojekte. Software Management Fähigkeiten sind unerläßlich für die Unterstützung des Softwareentwicklungsprozesses. Ohne Software Management Fähigkeiten geht der Überblick über die Vielzahl der softwaretechnischen Details verloren, was zu inhaltlichen Fehlentscheidungen im Projekt führen kann. Diese wiederum vergeuden Zeit, personelle Kapazität und Geld, was den Projekterfolg gefährdet.
Wenn eine Vorgehensweise angewendet werden soll, mit der das Projektteam nicht hinreichend vertraut ist, müssen Kapazitäten für das Erlernen der neuen Vorgehensweisen verwendet und eigene Aufwände hierfür geplant werden. Weiterhin müssen die Entwickler sowohl im Projekt, als auch bezüglich der neuen Vorgehensweise gleichzeitig die Einarbeitung in ihnen nicht vertraute Themen bewältigen. Diese Doppelbelastung vermindert die Konzentration auf die eigentlichen Projektziele und gefährdet so den Projekterfolg.
Wenn die Entwickler sich beispielsweise neben der fachlichen Einarbeitung in eine neue Domäne eines Mediengroßhändlers zusätzlich auf iterative Vorgehensweisen umstellen müssen, bedeutet dies eine erhebliche Zusatzbelastung, die auf Kosten der Projektinhalte gehen kann.
Die Kenntnisse im Software Management und in der hierfür angewendeten Vorgehensweise ist also eine bedeutende Projektressource.
Beispiel: Wenn durch mangelhaftes Requirements Management vordringliche Anforderungen nicht mit in das nächste Inkrement aufgenommen werden oder durch mangelndes Change Management widersprüchliche Fehlerkorrekturen darin einfliessen, entsteht daraus ein Release, das den Bedürfnissen der Benutzer nicht im möglichen Maß gerecht wird. Dies kann Akzeptanzverluste zur Folge haben und langfristig den Projekterfolg gefährden.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen das Software Management unterschiedlich umfassend und detailliert. Das V-Modell XT [VMX06] bietet ausführliche Beschreibungen, Extreme Programming [Bec00] verläßt sich hier auf die Fähigkeiten seiner Anwender.
Beispiel: Umfassend unterstützt das V-Modell XT [VMX06] die Bedarfsdeckung der Software Management Fähigkeiten mit Vorgehensbausteinen wie K*onfigurationsmanagement* und *Problem- und Änderungsmanagement*. Extreme Programming [Bec00] läßt Projektleiter mit unzureichenden Fähigkeiten bezüglich des Software Managements weitgehend ohne Unterstützung (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Verfügbare Fähigkeiten im Software Management (*EPSWE-BedDeck-Fähigk-Software-Management)* steigert die Effizienz der eingesetzten Projektressourcen Personal, Budget und Zeit. Dadurch wird der Zustand der *EPP-BedDeck-Kap-Personell*, der *EPP-Bed-Deck-Kap-Projektbudget* und der *EPP-BedDeck-Kap-KalendarischeZeit* verbessert. Diese Beeinflussungen werden proportional eingeschätzt und mit „≈“ vorgeschlagen.
Komplizierte Tätigkeiten im Software Management können durch Aktivitätsmodelle unterstützt werden, komplizierte Strukturen zu Dokumentation von deren Resultaten mit Arbeitsproduktmodellen. Dies kann im Rahmen der Unterstützung des Projektmanagement (*EV-UnterstProjTheBer-SoftwareManagement)* erfolgen (siehe Kapitel 3.2).

Belege aus der Literatur

Bei [Plo02] zitiert nach [Wal04] wird „Falsche Wahl der Projektressourcen“ und bei [Hal98] zitiert nach [Wal04] nicht hinreichende „Mitarbeiter-Fähigkeiten“ als Projektrisikofaktor bezeichnet, was sich auch in fehlenden Kenntnissen im Software Management auswirken kann. „Fehlende Erfahrung in neuen Vorgehensweisen“ bei Computerwoche zitiert nach [Gaul04] weist ebenfalls auf Defizite beim Know-how des Projektteams hin.

2.3.3 Projektthemenbereich Projektmanagement

Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Team“ und des Projektthemenbereichs „Projektmanagement“ zeigt der folgende Ontologieausschnitt in Abbildung 17.

2.3 Merkmale des Risikobereichs Entwicklungs-Team

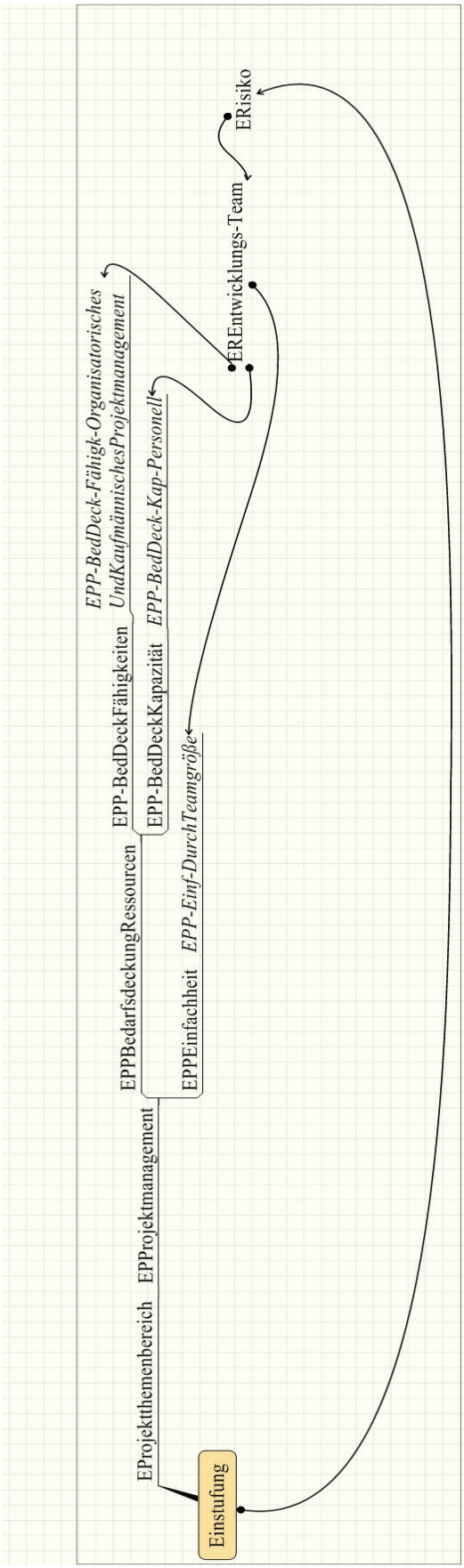

Abbildung 17 Merkmale Risikobereich Entwicklungs-Team und Projektthemenbereich Proj.Man.

2.3.3.1 Projektmanagement Fähigkeiten

Definition des Merkmals *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement*:
Kaufmännische Fähigkeiten: Sie stehen für die Kompetenz bezüglich aller finanziellen Belange des Projektes wie Stundensätze, Kostenkalkulationen, Zahlungsflüsse, Finanzierung und Budgetierung, aber auch Aufwandserfassung und -kontrolle.
Organisatorische Fähigkeiten: Sie beinhalten Aspekte der Ressourcen. Die Planung, das transparent machen und Stabilisieren, die Koordination von personellen Kapazitäten, Verantwortlichkeiten, Projektorganisation, Terminen, aber auch Ausstattung mit Arbeitsmitteln und Arbeitsplätzen fällt hierunter.
Projektmanagement umfasst die Planung, Kontrolle und Steuerung [Bur95] von Terminen, Arbeitsprodukten, Ressourcen, Aktivitäten, Budget, Risiken.
Bedarfsdeckung steht auch hier für das Verhältnis von verfügbaren und geforderten Fähigkeiten.
Beispiel: Wenn in einem Projekt aufgrund von mangelnder *EPP-BedDeck-Kap-Projektbudget* ein hoher Bedarf an kaufmännischen Fähigkeiten und ein hoher organisatorischer Bedarf aufgrund eines großen Teams besteht und nicht die entsprechenden kaufmännischen und organisatorischen Fähigkeiten vorhanden sind, ist keine Bedarfsdeckung gegeben.
Abgrenzung zur *EPSWE-BedDeck-Fähigk-SoftwareManagement*: Die im aktuellen Abschnitt beschriebenen Fähigkeiten sind im Gegensatz zu den Software Management Fähigkeiten nicht spezifisch für Softwareprojekte.

Motivation des Merkmals *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement:*
Bedeutung für Softwareprojekte. Ohne kaufmännische Fähigkeiten kann ein Projekt nicht kalkuliert und wirtschaftlich kontrolliert werden. Ohne organisatorische Fähigkeiten wird die Zusammenarbeit des Teams nicht koordiniert und somit werden die personellen Kapazitäten nicht effizient eingesetzt.
Hierzu muß ergänzt werden, daß der Projektleiter das effiziente und effektive Ineinandergreifen von Entwicklungsschritten koordiniert, die Kommunikation zwischen den Mitarbeitern organisiert und er den Überblick über den Einsatz und Verbrauch von Projektressourcen im Verhältnis zum erzielten Arbeitsfortschritt behalten muß, um planen, kontrollieren und nachsteuern zu können. Hierfür benötigt er Projektmanagement-Wissen und -Erfahrung, sowie die Fähigkeit, Teams zu organisieren und zu führen.
Falls also für Kommunikations- und Koordinationsaufgaben zur Leitung eines Projektes nur ein introvertierter, kommunikationsscheuer Mitarbeiter zur Verfügung steht, ist diese Aufgabe ungeeignet besetzt.
Beispiel: Wenn ein Projektleiter die inhaltlichen Experten in seinem Team nicht organisieren, ihnen nicht delegieren und ihre Arbeit nicht kontrollieren kann, führt dies zu Fehlleistungen, Doppelarbeiten und nicht zusammenpassenden Arbeitsprodukten. Somit werden Nacharbeiten erforderlich, welche die Projektressourcen kalendarische Zeit, Projektbudget und personelle Kapazität verschwenden, was den Projekterfolg gefährdet.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen die *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement* unterschiedlich. Das V-Modell XT [VMX06] bietet umfassende Unterstützung. Scrum adressiert dieses Merkmal vergleichsweise zurückhaltend und baut mehr auf die entsprechenden Fähigkeiten seiner Benutzer [Sch02].

Beispiel: Kaufmännische Fähigkeiten werden vom V-Modell XT [VMX06] im Vorgehensbaustein *Kaufmännisches Projektmanagement* unterstützt und ergänzt. Organisatorische Fähigkeiten werden vom V-Modell XT [VMX06] im Vorgehensbaustein *Projektmanagement* (siehe Kapitel 3.3) unterstützt.

Beeinflussungsbeziehungen mit anderen Merkmalen

Durch hohe Fähigkeiten im Projektmanagement (*EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement)* werden die Projektressourcen Personal, Budget und Zeit effizienter eingesetzt. Das verbessert die *EPP-BedDeck-Kap-Personell*, die *EPP-BedDeck-Kap-Projektbudget* und die *EPP-BedDeck-Kap-KalendarischeZeit.* Diese Beeinflussungen werden überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Durch die organisatorischen Fähigkeiten arbeiten auch größere Teams besser zusammen, der Zustand von *EPP-Einf-DurchTeamgröße* verbessert sich. Der Einfluß kann das Handicap großer Teams nur bedingt beeinflussen, daher wird er proportional eingeschätzt und mit „≈“ vorgeschlagen.
Komplizierte Tätigkeiten im Projektmanagement können durch Aktivitätsmodelle im Rahmen der Unterstützung des Projektmanagements (*EV-UnterstProjTheBer-Projekt-Management)* unterstützt werden, komplizierte Strukturen zur Dokumentation von deren Resultaten mit Arbeitsproduktmodellen in dieser Disziplin (siehe Kapitel 3.2).

Belege aus der Literatur

Der Risikofaktor „Falsche Besetzung der Position des Projektleiters“ bei der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04] deutet darauf hin, das die Fähigkeiten des Projektleiters von signifikanter Bedeutung für den Projekterfolg sind. [Wal04] fordert einen erfahrenen und geschickter Projektleiter. Diese Darstellung wird auch durch die Risikofaktoren „Unfähige Projektleiter“ [Kel94], „Personelles Versagen“ sowie „Unrealistische Zeit- und Kostenplanung“ bei [Boe98] zitiert nach [Wal04] und „Fehlendes Projektplanungs-Know-how“ bei Computerwoche zitiert nach [Gaul04] bestärkt. Als daraus resultierend können die Risikofaktoren „Schlechte Planung und Schätzung“ bei der KPMG Studie “IT Runaway Systems“ zitiert nach [Gaul04], zu ungenaue „Projektplanung“ bei [Hal98] zitiert nach [Wal04], „Managementprozeß“ (im Sinne einer ungenauen Planung) [Wal04], „Unzureichende Projektmanagement-Methoden“ bei KPMG Studie „IT Runaway Systems“ zitiert nach [Gaul04], „Mangelhafter Überblick über den Projektstand und Fortschritt“ [Gru01], „Fehlende Termin- und Kostenüberwachung“ [Gru01], „Zuwenig an Kontrolle“ [Kel94] aber auch „Zuviel an Kontrolle“ [Kel94] und „Unzureichende Projektplanung“ bei der KPMG-Studie „What went wrong?“ zitiert nach [Gau04] betrachtet werden. Die TechRepublic-Studie zitiert nach [Gau04] leitet als Empfehlung für IT-Projekte besser ausgebildete Projektleiter ab. Auch bei [Plo02] zitiert nach [Wal04] wird „Schlechtes Projektmanagement“ als einer der häufigsten Gründe für das Scheitern von IT-Projekten dargestellt. [Wal04] sieht Managementmethoden (im Sinne von nicht hinreichender Steuerung und Kontrolle) als Standardrisiko für Softwareprojekte. Auch in [Wal01] wird die Wichtigkeit der Projektmanagementfähigkeiten betont. Bei [Hal98] zitiert nach [Wal04] werden nicht hinreichend definierte Rollen und Verantwortlichkeiten als Projektrisikofaktor bezeichnet. Hier muß aus der Sicht der vorliegenden Arbeit eingeschränkt werden, daß Rollendefinitionen die Aufgabe eines Vorgehensmodells ist und nur bedingt in der Verantwortung eines Projektleiters liegt, auch wenn die Rollenzuweisung üblicherweise seine Aufgabe ist.

Der Standish Report empfiehlt als Projekterfolgskriterium „Proper Planing“ [Stan94]. Auch [Bro95] benennt bei seiner Darstellung wichtiger Projektkenngrößen das Merkmal „Organisation“. [Wal04] fordert den Einsatz eines akzeptierten und standardisierten Projektmanagementprozesses, bei Kritische Erfolgsfaktoren des Projektmanagements bei [ISA00] zitiert nach [Wal04] wird „Hinreichende Projektplanung“ empfohlen.

2.3.3.2 Teamgröße

Definition des Merkmals *EPP-Einf-DurchTeamgröße*:
Die *EPP-Einf-DurchTeamgröße* steht für den Kommunikations- und Koordinationsaufwand durch die Anzahl der am Projekt beteiligten Personen. Speziell Teilzeitkräfte müssen hinsichtlich des Koordinations- und Kommunikationsbedarfes als volle Personen gerechnet werden, da sie aufgrund partieller Verfügbarkeit hinsichtlich der Kommunikation noch schwieriger einzubinden sind als Vollzeitkräfte, ihr Informationsbedarf aber mit deren vergleichbar sein kann.
Beispiel: In einem Projekt mit drei Mitarbeitern gibt es drei Kommunikationspfade, mit vier Mitarbeitern sind es bereits sechs.

Motivation des Merkmals *EPP-Einf-DurchTeamgröße*:
Bedeutung für Softwareprojekte. Die Organisation im Projekt sorgt für das effiziente Ineinandergreifen von Rollen und Aktivitäten im Projekt und wird mit steigender Projektgröße immer wichtiger. Projekte unterscheiden sich mitunter stark nach ihrer Teamgröße. Eine steigende Teamgröße führt zu erhöhtem Kommunikationsaufwand im Team. Durch eine größere Anzahl Mitarbeiter wird bei deren Projektarbeit an mehr Stellen neues Wissen erschlossen, welches anderen Projektmitarbeitern nutzbringend zugänglich gemacht werden muß. Wissensaustausch ist dadurch zwischen mehr Menschen erforderlich. Die Kommunikationspfade werden bei einer flachen Projekthierarchie mit direkter Kommunikation mehr und bei einer tieferen Projekthierarchie mit Kommunikation über Zwischenstationen länger [Boe81]. Größere Teams erfordern dauerhaft einen höheren Kommunikationsumfang, was auf lange Sicht die Produktivität verschlechtert. Die Kommunikation, das Management, die Führung und Kontrolle kann oftmals in großen Projekten nicht mehr direkt erfolgen [Hum97], was sich ebenfalls nachteilig auf die Effizienz auswirkt. Personelle Verstärkungen haben nur auf lange Sicht das Potential, die Entwicklungsgeschwindigkeit zu erhöhen. Bei einer Verstärkung des Projektteams oder einem Wechsel von Mitarbeiten muß anfänglich sehr viel fachliches und technisches Wissen an die neuen Projektmitarbeiter vermittelt werden. Durch diese Zusatzbeanspruchung sinkt die Produktivität der etablierten Projektmitarbeiter. Durch den erhöhten Kommunikationsanteil der Arbeitszeit sinkt aber der Anteil der wertschöpfenden Entwicklungsarbeit und damit die Effizienz der eingesetzten Projektressourcen kalendarische Zeit, Projektbudget und personelle Kapazitäten, was den Projekterfolg gefährdet.
Dies macht die personelle Teamgröße als wichtige Kenngröße für Softwareentwicklungsprojekte plausibel.
Beispiel: Wenn ein Projektteam von zehn auf fünfzehn Personen verstärkt wird, bedeutet dies neben dem Einarbeitungsaufwand für die neuen Projektmitglieder, daß potentiell jedes Projektmitglied mit maximal vierzehn und nicht neun anderen kommunizieren muß.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Eine große Anzahl an Mitarbeitern wird in Vorgehensmodellen unterschiedlich durch Rollenmodel-

le organisiert. Das V-Modell XT [VMX06] und der Rational Unified Process [RUP06] bieten umfassende Rollenmodelle, während die Agilen Ansätze wie Extreme Programming [Bec00] und Scrum [Sch02] ein Rollenmodell höchstens andeuten und die Definition von Verantwortungsbereichen im Projekt den vorhandenen organisatorischen Fähigkeiten im Projekt überlassen.
Beispiel: Das V-Modell XT [VMX06] bietet ein umfassendes Rollenmodell mit über dreissig Rollen, um die Verantwortlichkeiten der Projektmitarbeiter zu definieren und ihre Zusammenarbeit zu organisieren.
Scrum [Sch02] unterstützt den Projektleiter hingegen wenig bei der Organisation vieler Projektmitarbeiter, weil es nur wenige Rollen benennt und deren Verantwortlichkeiten für Aktivitäten und Arbeitsprodukte nur vage beschreibt (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Eine hohe Anzahl Projektmitarbeiter vermindert die *EPP-Einf-DurchTeamgröße.* Durch sie sinkt die Effizienz der eingesetzten Ressourcen Personal, Budget und Zeit im Projekt, weil mit steigender Teamgröße die Anzahl der erforderlichen Kommunikationspfade und damit der Kommunikationsanteil der Arbeitszeit steigt und der Anteil der direkt wertschöpfenden Arbeit sinkt. Dadurch verschlechtert sich die *EPP-BedDeck-Kap-Personell*, die *EPP-BedDeck-Kap-Projektbudget* und die *EPP-BedDeck-Kap-KalendarischeZeit*. Dieser Einfluß wird proportional eingeschätzt und daher mit „≈“ vorgeschlagen.
Mit wachsenderTeamgröße steigt der Kommunikationsaufwand und damit verschlechtert sich die *EPM-BedDeck-KommunikationImTeam*. Dieser Einfluß wird proportional eingeschätzt und ebenfalls mit „≈“ vorgeschlagen.
Durch den erhöhten Bedarf an Organisation verschlechtert sich die *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement* und erhöhen sich die Anforderungen an ein Rollenmodell in den betroffenen Disziplinen. Diese Beeinflussung wird proportional eingeschätzt und mit„≈“ vorgeschlagen. Die *EPP-Einf-DurchTeamgröße* kann durch die Unterstützung durch Projektmanagement (*EV-UnterstProjTheBer-ProjektManagement)* verbessert werden (siehe Kapitel 3.2).

Belege aus der Literatur

[You04] sieht die Teamgröße als wichtiges Projektkennzeichen. Auch die „TechRepublic“-Studie zitiert nach [Gau04] zeigt einen starken Zusammenhang zwischen Projekterfolg, Teamgröße und Funktionsumfang, drückt dies aber in Projektdauer und -Kosten aus. Projektdauer und Kosten sind nach [Eck04] nur indirekte Einflußfaktoren, die sich aus der Teamgröße und dem Funktionsumfang herleiten lassen. Auch in [Wal01] wird die Teamgröße als kritischer Faktor bei Softwareprojekten gesehen.
„Drastische Personalaufstockung“ wird bei McConnel zitiert nach [Gaul04] als Risikofaktor für Projekte aufgefasst.

2.3.3.3 Personelle Kapazität

Definition des Merkmals *EPP-BedDeck-Kap-Personell:*
Diese bezeichnet das Verhältnis zwischen Bedarf und Verfügbarkeit von Mitarbeitertagen für Projektarbeiten.
Abgrenzung von *EPP-Einf-DurchTeamgröße*: Während bei der Teamgröße das Anwachsen des Kommunikations- und Koordinationsaufwandes und der damit verbundene sinkende Anteil von wertschöpfender Arbeit im Vordergrund steht, wird hier die Ver-

fügbarkeit von Mitarbeitern und deren Arbeitsstunden betrachtet.
Beispiel: Wenn für ein Projekt unter Berücksichtigung der kalendarischen Zeit bis zum Liefertermin und des zu erbringenden Funktionsumfanges nicht genügend Mitarbeiter zur Verfügung stehen, liegt eine Bedarfsunterdeckung vor.

Motivation des Merkmals *EPP-BedDeck-Kap-Personell*
Bedeutung für Softwareprojekte. Die Bedarfsdeckung der personellen Kapazitäten zwingt Softwareprojektleiter oft zu Begrenzungen des Funktionsumfanges oder zur Verschiebung von Terminen.
Beispiel: Falls die zu erbringende Funktionalität doppelt so viele Mitarbeiter erfordern würde als verfügbar sind, müssen Funktionen gestrichen werden, wenn der Termin unveränderlich ist.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Das V-Modell XT [VMX06] unterstützt die Planung von Ressourcenauslastungen, optimiert aber nicht bei vorhandenen Engpässen. Hierfür eignet sich eher Extreme Programming [Bec00].
Beispiel: Bestehenden personellen Kapazitätsengpässen im Projekt wird am ehesten XP [Bec00] gerecht. Durch seinen minimalistischen Ansatz, nur *Story Cards*, Testfälle und Code zu entwickeln und nur die Arbeit zu machen, die unbedingt erforderlich ist, wird versucht, kurzfristig äußerst schonend mit den Arbeitsaufwänden der Projektmitarbeiter umzugehen (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen
Eine geringe Bedarfsdeckung personeller Kapazitäten (*EPP-BedDeck-Kap-Personell)* verschlechtert die Terminsituation (*EPP-BedDeck-Kap-KalendarischeZeit)*, weil bestehende Aufgaben durch zu wenige Leute erledigt werden müssen.
Der hieraus resultierende Optimierungsbedarf erhöht die Anforderungen an das Projektmanagement und verschlechtert also den Zustand der *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement*. Diese Einflüsse werden als überproportional eingeschätzt und mit „≈≈“ vorgeschlagen.
Die *EPP-BedDeck-Kap-Personell* kann durch die Unterstützung durch Projektmanagement (*EV-UnterstProjTheBer-ProjektManagement)* verbessert werden (siehe Kapitel 3.2).

Belege aus der Literatur
[Gru01] führt „Unzureichende Freistellung des Projektleiters“ als Risikofaktor für Projekte an. Hier ist zu verdeutlichen, daß ein nicht im ausreichenden Maße verfügbarer Projektleiter seinen Aufgaben im Projekt nicht hinreichend nachkommen kann, wodurch entweder Koordinationsaufgaben zu Lasten der Entwicklungsarbeit auf Entwickler übertragen werden oder gar nicht erledigt werden, wodurch die Effizienz des ganzen Teams leidet. Dies führt zur ineffizienten Ausnutzung der Projektressourcen Zeit und Geld, was den Projekterfolg gefährden kann. Hieraus läßt sich ableiten, das personelle Kapazitäten eine wichtige Projektressource darstellen, deren Verknappung den Projekterfolg gefährdet. Dies läßt sich auch an weiteren Risikofaktoren aus der Literatur wie „Fehlende Ressourcen“ [Ver00], „Verfügbarkeit und Qualität der Mitarbeiter“, bei Daily Telegraph zitiert nach [Gaul02], „Ressourcenklau zwischen rivalisierenden Projekten“ [Kel94], „Unrealistische Ressourcenplanung“ bei forsa zitiert nach [Gaul04], „Überbeanspruchung der Mitarbeiter“ bei Jalote zitiert nach [Gaul04] und “Mangel an Personal“ bei Boehm zitiert nach [Gaul04] und nicht ausreichende „Ressourcen“ [Wal04] verdeutlichen.

Auch [Gau04] sieht mit „Ressourcen“ die personellen Kapazitäten als wichtige Projektkenngröße. Dies wird auch in [Wal01] betont. Daß aber auch bei guter Verfügbarkeit von personellen Kapazitäten sehr bewußt mit ihrer Veränderung umgegangen werden muß, belegt der Risikofaktor „Drastische Personalaufstockung“ bei McConnel zitiert nach [Gaul04] und bestätigt so auch Brooks' Law „Adding man power to a late project makes it later“ [Bro95]. Personen sind nicht beliebig zwischen Projekten austauschbar [DeM01].
Da die Verfügbarkeit personeller Kapazitäten Auswirkungen auf den Projekterfolg hat, sind diese eine bedeutsame Projektressource.

2.3.4 Projektthemenbereich Menschenführung

Menschenführung spielt bei einer so wissensgetriebenen Tätigkeit wie der Softwareentwicklung eine bedeutsame Rolle. Der Bedarf, hinsichtlich von Menschenführung einzuwirken, unterscheidet sich von Projekt zu Projekt stark. Bei einem kleinen eingespielten Team, das seine Teambildungsphasen wie „Forming, Storming, Norming, Performing“ (vergleiche auch [Kel94]) bereits erfolgreich durchlaufen hat und auf mehrere erfolgreiche Projekte zurückblicken kann, ist eine diesbezügliche Unterstützung weniger erforderlich.
Bei einem krisenbelasteten Projekt unter existentiellem Druck oder einer durch große Interessengegensätze gekennzeichneten Situation kann es dagegen sehr entscheidend sein, ein Bewußtsein für Menschenführung im Projekt zu wecken und sie zu gezielt zu berücksichtigen.
„Der Faktor Mensch“ [DeM99] wird in der Literatur als wesentlicher Risikofaktor für Softwareentwicklungsprojekte gesehen und oft wird dargestellt, daß Projekte eher an Menschen als an Technik scheitern. In [DeM01] werden Menschen und ihr Wissen als das wichtigste Kapital einer Organisation bezeichnet. In [Hum07] wird „Motivating technical and professional people“, in [Hum02] „disciplined and motivated people“ von seiner Wichtigkeit unterstrichen. [Hum02] benennt „Team motivation“ als sehr bedeutsam. [You04] benennt „Loyalty, Commitment, Motivation, Communication“ als wichtige menschliche Erfolgsfaktoren für Projekte. [Kot96] skizziert sehr anschaulich den Unterschied zwischen „Management“ (siehe Kapitel 2.3.3„Projektthemenbereich Projektmanagement“) und „Leadership“ im aktuellen Kapitel).
Die Herleitung der Projektmerkmale Risikobereichs „Entwicklungs-Team“ und des Projektthemenbereichs „Menschenführung“ zeigt der folgende Ontologieausschnitt in Abbildung 18.

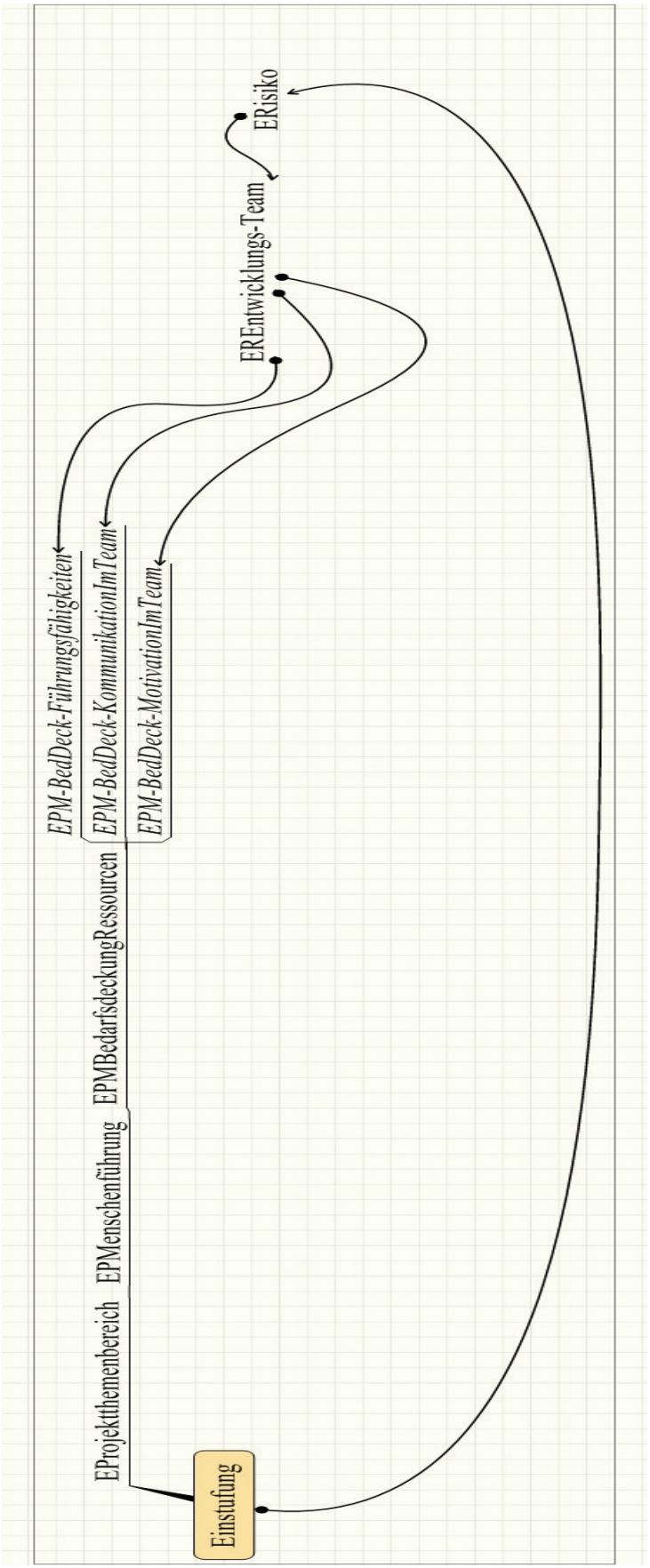

Abbildung 18 Merkmale Risikobereich Entwicklungs-Team und Projektthemenber. Menschenführ.

2.3.4.1 Motivation im Team

Definition des Merkmals *EPM-BedDeck-MotivationImTeam*:
Die Motivation im Team steht für den Willen, sich mit dem Projekt zu identifizieren und so gut wie möglich mit seinen Fähigkeiten in ein Projekt einzubringen. Hierunter sind auch beispielsweise der konstruktive Umgang mit abweichenden Individualzielen, Lernwilligkeit, Lernfähigkeit, Identifikation mit dem Projekt, sowie gegenseitiger Respekt und Akzeptanz zu verstehen. Bedarfsdeckung steht für das Verhältnis zwischen erforderlicher und vorhandener Motivation.
Beispiel: Wenn ein Requirements Engineer Anforderungen beschreibt, müßte jemand, der diese mit letzter Konsequenz überprüfen will, fachlich und methodisch dasselbe Wissen haben, was in einem Projekt aber eine teure Doppelbesetzung eines Qualifikationsprofils darstellt, die oft nicht gegeben ist. Die Qualität der Anforderungen ist somit nicht schlußendlich überprüfbar und hängt deshalb im starken Maß von der Motivation des Requirements Engineers ab.

Motivation des Merkmals *EPM-BedDeck-MotivationImTeam*:
Bedeutung für Softwareprojekte. Projekte unterscheiden sich nach den beteiligten Menschen und sozialen Strukturen. Qualifizierte Menschen sind unbedingt für die Softwareentwicklung notwendig. Menschliche Fähigkeiten wie Intuition, Kreativität, Folgern und Schlüsse ziehen, kombinieren, abstrahieren, differenzieren oder antizipieren sind für eine schöpferische Tätigkeit wie die Softwareentwicklung unerläßlich. Unvermeidbar ist aber, daß man neben den positiven menschlichen Eigenschaften auch mit deren negativen Seiten für das Projekt wie Egoismus, Geltungsbedürfnis, Eifersucht, Individualziele, Machtstreben oder Eigensinn umgehen muß. Damit Menschen ihre positiven Eigenschaften nutzbringend in ein Projekt einbringen können, müssen sie sich emotional und sozial wohl fühlen [Ros03]. Die Motivation der Mitarbeiter im Projekt hängt von den Rahmenbedingungen wie zum Beispiel der Übereinstimmung ihrer Projekttätigkeit mit ihren Individualzielen ab. Bei fehlender Motivation der Mitarbeiter im Projekt nutzen diese ihre bestehenden Entscheidungsspielräume mitunter zu Ungunsten des Projektes aus. Da Softwareentwicklung eine Wissensarbeit ist, bei der Spezialisten unterschiedlicher Disziplinen mit üblicherweise hohem Innovationsgrad zusammenarbeiten, ist eine wirklich effektive Kontrolle ihrer Arbeit nur unter sehr hohem Aufwand möglich. Dies ist oftmals wirtschaftlich nicht vertretbar. Daher hängt es von der Motivation des spezialisierten Einzelnen ab, ob er seine Arbeit wirklich so gut macht, wie es ihm möglich ist, oder ob er nur die Arbeiten ordentlich erledigt, deren Arbeitsprodukte besser überprüfbar sind [Kei05].
Beispiel: Wenn ein Entwickler sich von seinem Projektleiter nicht entsprechend gewürdigt und durch dessen Aufwandsschätzungen bevormundet fühlt, stört dies den optimalen Einsatz seines technischen Wissens und Könnens. Wenn eine Schätzung durch jemanden erfolgt, der die betreffende Aufgabe nicht durchführt, fühlt sich der für die Aufgabe Verantwortliche unter Umständen nicht gebunden und ist wenig motiviert, sie einzuhalten.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Speziell die Agilen Ansätze legen großes Gewicht auf „motivated Individuals“ [AGI06] und stellen diese über die Befolgung von Prozessen. Sie propagieren generell eine an sozialen und emotionalen Faktoren orientierte Geisteshaltung. Allerdings wird auch die Unterstützung der Motivation allenfalls mit Prinzipien beschrieben und nicht durch konkrete Rollen, Aktivitäten und Arbeitsprodukte wie zum Beispiel dem Ausdrücken von Anerken-

nung unterstützt. Die etablierteren Ansätze wie V-Modell XT [VMX06] oder Rational Unified Process [RUP06] betonen dieses Thema nicht.
Beispiel: Extreme Programming [Bec00] definiert den *Value Respect* als Voraussetzung für Motivation der Mitarbeiter (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Der Zustand der Motivation im Team (*EPM-BedDeck-MotivationImTeam)* ist im Zusammenhang mit den Menschen im Projekt sehr bedeutungsvoll für eine produktive Projektarbeit. Bei mangelnder Motivation werden die Projektressourcen Zeit, Budget und personelle Kapazität ineffizient eingesetzt, wenn pro Zeiteinheit tendenziell weniger Ergebnisse erzielt, mitunter aber trotzdem auf das Projekt gebucht wird. Dies verschlechtert die *EPP-BedDeck-Kap-KalendarischeZeit*, die *EPP-BedDeck-Kap-Projektbudget* und die *EPP-BedDeck-Kap-Personell*, was den Projekterfolg gefährdet.
Wenn unmotivierte Mitarbeiter weniger konstruktiv und zielführend für das Projekt kommunizieren, verschlechtert sich die *EPM-BedDeck-KommunikationImTeam*. Alle diese Beziehungen werden proportional eingeschätzt und daher mit „≈“ vorgeschlagen.
Die Bedarfsdeckung der Motivation im Team kann durch die Unterstützung der Menschenführung (*EV-UnterstProjTheBer-Menschenführung)* verbessert werden (siehe Kapitel 3.2).

Belege aus der Literatur

Als Projektrisikofaktoren werden in der Literatur „Motivationsmangel“ bei McConnel zitiert nach [Gaul04] benannt. Die Ursache für Motivationsmängel können Konflikte im Team sein, die ebenfalls als Risikofaktoren genannt werden wie „Streit im Team“ [Kel94] und „Spannungen und Konflikte im Projektteam“ [Gru01]. Auch einzelne „Problemmitarbeiter“ bei McConnel zitiert nach [Gaul04] sind ein Risikofaktor, der Konflikte auslösen und damit die Motivation für die Projektarbeit vermindern kann. Ebenso kann sich eine negative „Projekthistorie“ [Gaul04] motivationsmindernd auswirken, wie der Risikofaktor „Demotivation der Anwender aufgrund früherer Projektfehlschläge“ [Gru01] verdeutlicht.
Der Chaos Report empfiehlt „Ownership“ [Stan94] für ein Projekt als Erfolgsfaktor und benennt eine „Hard working and focused Staff“ [Stan94].

2.3.4.2 Kommunikation im Team

Definition des Merkmals *EPM-BedDeck-KommunikationImTeam*:
Sie steht für den verbalen oder schriftlichen, formellen oder informellen Austausch projektrelevanter fachlicher, technischer, administrativer und sozialer Informationen im Projekt. Bedarfsdeckung steht hier für die rechtzeitige und hinreichend umfassende Weitergabe von Informationen, sowie für die Bereitschaft hierfür.
Beispiel: Die nicht rechtzeitige oder nicht komplette Kommunikation von Anforderungsänderungen, von Terminen oder personellen Veränderungen im Team ist keine bedarfsgerechte Kommunikation im Team.

Motivation des Merkmals *EPM-BedDeck-KommunikationImTeam*:
Bedeutung für Softwareprojekte. Bei einer innovationsgetriebenen und -bestimmten Tätigkeit wie der Softwareentwicklung, die zudem ständigen Veränderungen ausgesetzt ist, wird eine funktionierende Kommunikation zum existentiell notwendigen Einfluß [DeM97].

Hier muß verdeutlicht werden, daß ohne hinreichende Kommunikation aktuelle Informationen im Projekt nicht im benötigten Maße weiterverbreitet werden, was zu Doppelarbeit, Fehlern und Verzögerungen führen kann, weil Projektmitarbeitern der erforderliche fachliche, technische, methodische oder organisatorische Kenntnisstand für ihre aktuelle Projektarbeit nicht zur Verfügung stand. Hieraus resultieren oft Nacharbeiten, welche Zeit, Budget und personelle Kapazität vergeuden und überdies die Motivation verschlechtern. Mangelnde Kommunikation verstärkt die Intransparenz im Projekt. Bedarfsgerechte Kommunikation unterstützt ein soziales Klima, in dem Menschen produktiv arbeiten können.

Beispiel: Wenn eine Framework-Änderung nicht an die Applikationsentwickler kommuniziert wird, benutzen sie veraltete Funktionen oder Schnittstellen, was Nacharbeiten erforderlich macht, die bei rechtzeitigem Kommunizieren dieser Information nicht nötig gewesen wäre. Dadurch werden die Projektressourcen Zeit, Budget und personelle Kapazität ineffizient ausgenutzt, was den Projekterfolg gefährden kann.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen den Kommunikationsaspekt unterschiedlich umfassend und detailliert. Die etablierten Vorgehensmodelle bewirken ein Institutionalisieren der Kommunikation auf formalisiertem, schriftlichem Weg über Arbeitsprodukte, wie das V-Modell XT [VMX06] in seinem Produktmodell. Speziell die Agilen Methoden [AGI06] betonen wiederholt den Wert offener, ehrlicher, direkter und insbesondere verbaler Kommunikation und geben ihr den Vorzug vor der schriftlichen Wissenskonservierung und -weitergabe durch Modelle und Dokumente. Sie propagieren eine kommunikative Denkrichtung und Arbeitshaltung, beschreiben allerdings keine konkreten und über das übliche Maß hinausgehenden Rollen, Aktivitäten oder Arbeitsprodukte, welche die Kommunikation unterstützen sollen.

Beispiel: Das V-Modell XT [VMX06] beschreibt Produkte wie den *Projektplan* zur Konservierung und Weitergabe von Projektplanungsinformationen. Extreme Programming [Bec00] definiert die *Values Communication, Feedback,* die *Principles Rapid Feedback, Open Honest Communication* [Bec04], die *Techniques On-Site Customer, Planning Game, Metaphor, Pair Programming, Coding Standards,* und *Practices Sit together, Informative Workspace, Stories, Weekly Cycle, Quarterly Cycle, Real Customer Involvement, Daily Deployment, Negotiated Scope Contract* [Bec04], die dem Austausch und der Weitergabe von Informationen dienen und kommunikationsunterstützend aufgefasst werden können (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Gute Teamkommunikation *(EPM-BedDeck-KommunikationImTeam)* verbessert die Weitergabe und Verbreitung von für das Projekt relevanten Kenntnissen und Informationen. Es verbessert somit die in Summe über die Projektmitarbeiter verfügbaren fachlichen Fähigkeiten (*EPD-BedDeck-Fähigk-Fachlich)*, die verfügbaren Fähigkeiten im Software Engineering (*EPSWE-BedDeck-Fähigk-SoftwareEngineering)*, die verfügbaren Fähigkeiten im Software Management (*EPSWE-BedDeck-Fähigk-SoftwareManagement)* und die verfügbaren Projektmanagementfähigkeiten (*EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement).* Diese Beziehungen werden als überproportional eingeschätzt und daher mit „≈≈“ vorgeschlagen.

Die *EPM-BedDeck-KommunikationImTeam e*rhöht die Effizienz der eingesetzten Ressourcen Zeit, Budget und personelle Kapazität (siehe oben) und verbessert somit die *EPP-BedDeck-Kap-KalendarischeZeit*, die *EPP-BedDeck-Kap-Projektbudget*, und die

EPP-BedDeck-Kap-Personell.
Wenn im Rahmen einer bedarfsgerechten Kommunikation klare Aufgaben und Ziele vermittelt werden, erleichtert dies Erfolgserlebnisse, was die Motivation im Projektteam (*EPM-BedDeck-MotivationImTeam)* erhöhen kann.
Diese Beziehungen werden proportional eingeschätzt und mit „≈“ vorgeschlagen.
Die *EPM-BedDeck-KommunikationImTeam* kann durch die Unterstützung der Menschenführung (*EV-UnterstProjTheBer-Menschenführung* verbessert) werden (siehe Kapitel 3.2).

Belege aus der Literatur
In der Literatur findet die Kommunikation als Risikofaktor in Projekten vielfache Erwähnung und Beachtung: „Mangelnde Kommunikation im Projekt“ [Ver00] beziehungsweise „Schlechte Kommunikation“ bei der Daily Telegraph Studie zitiert nach [Gaul04] werden als Risikofaktor gesehen. Auch mit „Schlechte Kommunikation zwischen Projektmitgliedern“ bei der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04] und „Kommunikationsprobleme – Sprachbarrieren zwischen den Projektbeteiligten“ [Rup01] wird dies benannt. Neben der Kommunikation im Entwicklungsteam wird auch „Kommunikationsprobleme zwischen Team und Anwendern“ [Gru01], „Mangelnde Information der Fachabteilung“ [Gru01] und „Unzureichende Benutzereinbindung“ McConnel zitiert nach [Gaul04] als kritisch für Projekte eingestuft.
In der Chaos Studie wird „User Involvement“ [Stan94] als Projekterfolgsfaktor benannt, was nichts anderes als regelmäßige Kommunikation mit den Benutzern bedeutet. Mangelhafte Kommunikation nach außen führt zu „Fehlende Kenntnis über Projektprobleme beim Management“ in der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04] und schließlich auf „Fehlende Benutzerakzeptanz“ [Gru01], wodurch der Projekterfolg gefährdet ist. Auch [Wal01] sieht die Kommunikation im Team als wichtigen Erfolgsfaktor in Softwareprojekten.

2.3.4.3 Führungsfähigkeiten

Definition des Merkmals *EPM-BedDeck-Führungsfähigkeiten*:
Sie beschreibt das Bewußtsein und die Fähigkeit, im Projekt ein kooperatives Klima des Vertrauens, des Respekts, der Identifikation und des Weiterlernens zu schaffen. Auch Konfliktmanagement, Krisenmanagement, emotionale und soziale Kompetenz werden unter Führungsfähigkeiten zusammengefaßt.
Beispiel: Führungsfähigkeit kann sich darin ausdrücken, einem engagierten Projektmitglied ausdrücklich Anerkennung auszusprechen.

Motivation des Merkmals *EPM-BedDeck-Führungsfähigkeiten*:
Bedeutung für Softwareprojekte. Die Mitarbeiter in Softwareentwicklungsprojekten unterscheiden sich in ihrem Bedürfnis nach Anerkennung, Fairness oder Respekt nicht von anderen Menschen. Durch das Einbringen von Führungsfähigkeiten wird ein soziales und emotionales Klima geschaffen, in dem Menschen produktiv und motiviert an den Projektinhalten arbeiten können. Bedarfsdeckung drückt das Verhältnis zwischen erforderlichen und vorhandenen Führungsfähigkeiten aus.
Beispiel: Wenn Projektmitarbeiter durch fehlende Anerkennung unmotiviert sind, kann das bei einer innovativen Wissensarbeit wie der Softwareentwicklung eine schädliche Auswirkung auf die Arbeitsergebnisse und die Produktivität haben. Diese sind aber nur sehr eingeschränkt unter wirtschaftlich vertretbarem Aufwand überprüfbar. Um ein

Fachkonzept auf Vollständigkeit zu überprüfen, muß man ebenso viel über die Fachdomäne wissen wie der Autor. Eine derartige Doppelbesetzung eines Wissensbereiches wird in Projekten aber selten bezahlt.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Die Erfordernis von Führungsfähigkeiten wird von den Agilen Methoden [AGI06] propagiert und adressiert. Die etablierten Ansätze wie V-Modell XT [VMX06] und Rational Unified Process [RUP06] fokussieren sich auf inhaltliche und rationale Aspekte.

Beispiel: In Extreme Programming [Bec00] wird mit Bezug auf Führungsfähigkeiten der *Value Respect* beschrieben (siehe Kapitel 3.3).

Beeinflussungsbeziehungen mit anderen Merkmalen

Bedarfsgerechte Führungsfähigkeiten verbessern die Motivation der Projektmitarbeiter (*EPM-BedDeck-MotivationImTeam)*, beispielsweise wenn Menschen durch sie die angemessene Anerkennung für ihre Arbeit erhalten und wichtig für das Wohlbefinden ist(vgl. [Ros03]). Diese Beziehung wird überproportional eingeschätzt und daher mit „≈≈“ vorgeschlagen.

Die *EPM-BedDeck-Führungsfähigkeiten* kann durch die Unterstützung der Menschenführung (*EV-UnterstProjTheBer-Menschenführung)* verbessert werden (siehe Kapitel 3.2).

Belege aus der Literatur

In [DeM99] wird eine Anekdote von einem jungen Programmierer geschildert, der krank in die Arbeit kommt, um einen wichtigen Termin zu halten. Seine Managerin würdigt seinen persönlichen Einsatz, indem sie ihm eine Schüssel heiße Suppe bringt, die ihm helfen soll, bei Kräften zu bleiben, bis er nach Hause kann, um sich auszukurieren. Auf seine Frage hin warum sie das mache, erklärt sie ihm, daß es nach ihrer Auffassung zu ihrem Job als Manager gehöre, den Entwicklern die Rahmenbedingungen für ihre Arbeit zu verschaffen. Der Ausdruck ihrer Anerkennung für sein Engagement gehört ihrer Meinung nach dazu. Dies führt auf das Merkmal der Führungsfähigkeiten.

2.4 Einflußfaktoren aus dem Entwicklungs-Umfeld

Fragen zu Einflußfaktoren aus dem Projektumfeld. Viele Kenngrößen aus der Literatur sind zwar charakterisierend und bedeutsam für Projekte, repräsentieren aber nicht direkt den Projektzustand und werden auch nicht durch verschiedene Vorgehensmodelle unterschiedlich adressiert. Sie können daher nicht direkt als Kriterium für die Eignungseinstufung von Vorgehensmodellen herangezogen werden, geben aber Hinweise für die Auswahl von Projektmerkmalen.

Projekte unterscheiden sich nach ihren Rahmenbedingungen durch das Entwicklungs-Umfeld. Da Projekte nicht nur durch ihren Inhalt, sondern auch durch das gesamte Umfeld beeinflusst werden, sind die Projektrahmenbedingungen wesentlich. Um ein Projekt beurteilen zu können, muß man auch sein näheres Umfeld betrachten, analysieren und charakterisieren. Aus diesem wirken Einflüsse, die den Projekterfolg signifikant fördern oder beeinträchtigen können.

Die Fragen zum Projektumfeld in diesem Kapitel betreffen Einflußfaktoren, die nicht direkt beim Entwicklungs-Team oder Entwicklungs-Produkt des Softwareprojektes, sondern in seinem Entwicklungs-Umfeld angeordnet sind und für die Beeinflussungsbeziehungen mit Projektmerkmalen bestehen. Mit Hilfe der Fragen zu Einflußfaktoren aus dem Projektumfeld können Merkmale hinsichtlich ihrer Bedeutung für ein konkretes

Projekt eingeschätzt und ihr Zustand bewertet werden. Dies unterstützt bei der Anwendung des EVGM die projektspezifische Belegung von Merkmalen mit einem Startwert. Fragen zu Einflußfaktoren aus dem Projektumfeld dienen der Selektion und Bewertung von Projektmerkmalen.
Fragen zu Einflußfaktoren aus dem Projektumfeld helfen den Benutzern des EVGM bei der systematischen Untersuchung des Umfeldes, in welchem das Softwareentwicklungsprojekt durchgeführt werden soll. Sie unterstützen die Anwender des EVGM, sich klassische Risiken zu vergegenwärtigen und ihr eigenes Projekt diesbezüglich zu hinterfragen.

2.4.1 Definieren von Fragen zu Einflußfaktoren des Projekt-Umfelds

Die Fragen zu Einflußfaktoren aus dem Projektumfeld sind nicht direkt mit Werten von „--“ bis „++“ (siehe Kapitel 4) bewertbar, stehen aber immer in Beeinflussungsbeziehung zu mindestens einem Projektmerkmal. Sie haben immer einen konkreten Gegenstand und fragen nach dessen Einfluß auf Merkmale des Projektes. Fragen zu Einflußfaktoren aus dem Projektumfeld sind immer nach der Art
„Welche Beeinflussungen auf das Projekt sind durch <Einflußfaktor> zu erwarten und in welchem Ausmaß?“ formuliert. Teilweise sind sie dabei bewußt nicht auf spezifische Merkmale bezogen, um den Blickwinkel der Anwender des EVGM nicht zu verengen.
Die Fragen zu Einflußfaktoren aus dem Projektumfeld sind ein Werkzeug zur Unterstützung des Risikomanagements im Rahmen des EVGM. Die Notwendigkeit von Risikomanagement in Projekten belegen „Fehleinschätzung von Projektrisiken“ bei forsa zitiert nach [Gaul04], sowie „Außergewöhnliche Risiken“ [Gaul04] im Sinne von externen Risiken.
Zusammenfassung. Für die Untersuchung des Umfeldes von Projekten formuliert das EVGM Fragen bezüglich der folgenden Charakteristika (Bezeichner sind in Klammern dargestellt, die Kürzel entstehen durch die ontologische Herleitung, welche im Anhang, Abschnitt „Die Systematik“ verdeutlicht wird).
Die Stabilität und Transparenz Projektziele (EPD-StaTrans-Projektziele) steht für die Beständigkeit und Erkennbarkeit der angestrebten Zustände, die durch das Projektergebnis erreicht werden sollen. Die Stabilität und Transparenz der Eigenschaften von Werkzeugen (EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen) repräsentiert die Beständigkeit und Erkennbarkeit der Eigenschaften von Werkzeugen für die Entwicklung, sowie von Entwicklungs- und Zielplattformen. Geografische Verteilung oder Offshoring (EPP-Einf-DurchRäumlicheGeografischeVerteilung) ist gegeben, wenn ein Projekt an räumlich voneinander getrennten Standorten betrieben wird. Wirtschaftlich-Juristische Verteilung oder Outsourcing (EPP-Einf-DurchWirtschaftlichJuristischeVerteilung) ist relevant, wenn ein Projekt durch wirtschaftlich eigenständige Organisationen betrieben wird. Die Art des vertraglichen Verhältnisses (EPP-Einf-DurchArtDesVertraglichenVerhältnisses) steht bei Verträgen zwischen Projektparteien für die Unterscheidung zwischen Dienstleistungs- und Gewerksvertrag. Entwicklungstiefe (EPP-Einf-DurchEntwicklungstiefe) repräsentiert, für welche Tätigkeiten im Entwicklungszyklus eine Projektpartei verantwortlich ist. Die unternehmenspolitischen Rahmenbedingungen (EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen) bilden ab welche Unterstützung ein Projekt aus der Organisation zu erwarten hat. Organisatorische Rahmenbedingungen (EPP-Einf-DurchOrganisatorischeRahmenbedingungen) stehen für die Arbeitsbedingungen für das Projektteam. Die Rahmenbedingungen des Zielmarktes (EPP-Einf-DurchRahmenbedingungenDesZielmarktes) repräsentieren die Kon-

kurrenzsituation, in der sich das Projekt befindet. Risiken durch Abhängigkeiten (EPP-Einf-DurchRisikenDurchAbhängigkeiten) bestehen, wenn Sachverhalte, die für das Projekt entscheidend sind, nicht entscheidend durch das Projekt beeinflussbar sind. Kompatibilität zu Werten und Unternehmenskultur (EPM-Einf-DurchKompatibilitätZuWertenUndUnternehmenskultur) repräsentiert die Verträglichkeit des Projektergebnisses mit den in der Projektorganisation etablierten Standards und Wertvorstellungen.

2.4.2 Projektthemenbereich Domänenwissen

Die Herleitung der Fragen zu Einflußfaktoren aus dem Projektumfeld Risikobereichs „Entwicklungs-Umfeld“ und den Projektthemenbereich „Domänenwissen“ zeigt der folgende Ontologieausschnitt in Abbildung 19.

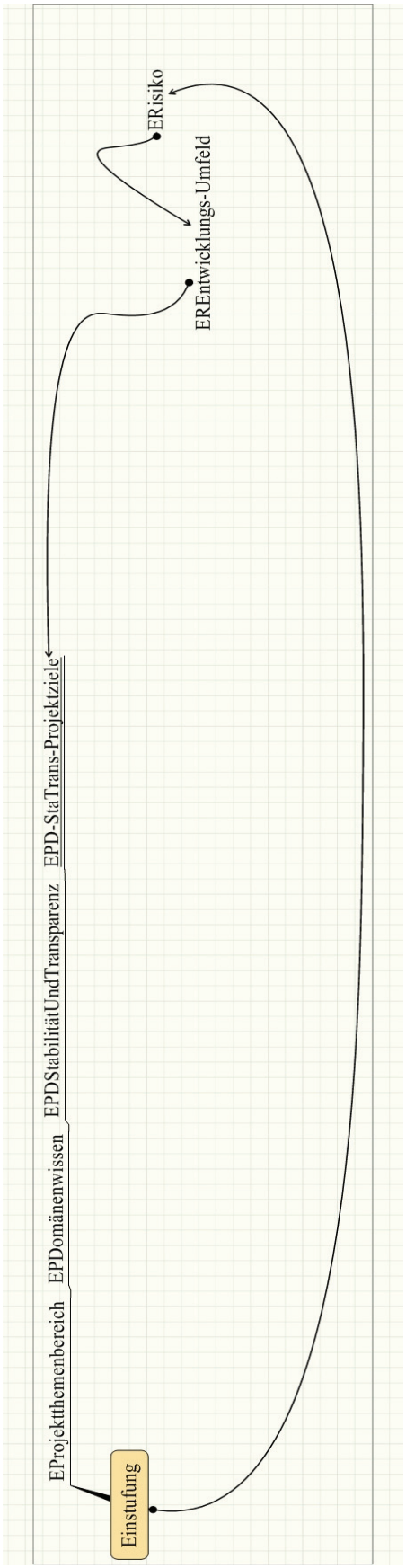

Abbildung 19 Fragen Einflußfakt. Risikober. Entw.-Umfeld und Projektthemenber. Domänenwiss.

2.4.2.1 Stabilität und Transparenz Projektziele

Definition der „EPD-StaTrans-Projektziele“:
Projektziele sind die angestrebten Zustände von definierten Zielgrößen, die durch ein Projekt in der Zielorganisation, im Zielumfeld oder bei der Zielgruppe auf fachlicher Ebene erreicht werden sollen und wegen denen das Projekt durchgeführt wird. Stabilität steht für die Beständigkeit der Projektziele, Transparenz für ihre Erkennbarkeit (siehe auch Glossar im Anhang).
Unterstützungsfrage: Wie und in welchem Ausmaß beeinflußt die Stabilität und Transparenz der Projektziele (EPD-StaTrans-Projektziele) das Softwareentwicklungsprojekt?
Beispiel: Ein Projektziel kann sein, sich mit einem weiterentwickelten Softwareprodukt in einem neuen Marktsegment zu plazieren. Falls dieses Ziel den Entwicklern nicht bekannt ist, messen sie die Relevanz von Anforderungen an das Produkt nur an dem Marktsegment, in dem es bereits etabliert ist und kommen zu einer nicht bedarfsgerechten Priorisierung.

Motivation der „EPD-StaTrans-Projektziele“:
Bedeutung für Softwareprojekte. Ein Projektteam kann die Projektziele nicht erreichen, wenn ihm diese nicht bekannt sind. Ohne ein gemeinsames Verständnis konkreter Projektziele ist ein Projekt in äußerster Gefahr, Ergebnisse zu produzieren, die dem Bedarf der Stakeholder nicht entsprechen, da man aufgrund des Umfangs von verschiedenen Lösungsmöglichkeiten nicht davon ausgehen kann, daß die Entwickler zufällig ein hinreichend bedarfsgerechtes Ergebnis entwickeln. Dadurch kann das Projektergebnis seinen Nutzen für den Auftraggeber nicht erzielen und dessen Investitionen in Form von kalendarischer Zeit und Projektbudget sind zumindest teilweise verschwendet.
Die Projektziele stellen den schlußendlichen Maßstab für den Projekterfolg dar. Daher leiten sich die fachlichen Anforderungen aus ihnen ab und müssen mit ihnen konform sein. Starke Veränderungen dieses Maßstabes bewirken starke Seiteneffekte in allen durch ihn zu bewertenden Teilen des Projektes und produzieren so erhebliche Zusatzaufwände bei der Veränderung von bereits erstellten Projekt-Arbeitsprodukten. Wenn diese Zusatzaufwände nicht kalkuliert waren, kann der hierdurch entstehenden Zusatzbedarf für die Ressourcen *EPP-BedDeck-Kap-Personell, EPP-BedDeck-Kap-Projektbudget* und *EPP-BedDeck-Kap-KalendarischeZeit* eine Gefährdung für den Projekterfolg darstellen.
Beispiel: Wenn das Projektziel, „Beschleunigung der Durchlaufzeiten von Vorgängen“ eines zu entwickelnden Workflow Management System dem Projektteam bekannt ist, können die Entwickler entsprechende Vorschläge zur deren Verbesserung entwickeln. Funktionalitäten wie „ausdrückliche Benachrichtigung eines Bearbeiters beim Eintreffen eines Vorgangs“, oder „Sortierung der eingehenden Vorgänge nach Dringlichkeit“ bekommen durch die Kenntnis des Projektzieles „Beschleunigung von Durchlaufzeiten“ erst einen Sinn.
Bei der Entwicklung einer Stammdatenverwaltung können zunächst die Ziele bestehen, die Lagerbuchhaltung, den Einkauf und den Verkauf zu unterstützen. Wenn dies dann in die Unterstützung des Vertriebes mit Customer Relationship Management auf Kosten der Unterstützung des Einkaufs verändert wird, kann nur der wiederverwendbare Teil der bereits erstellten Funktionalität für den Einkauf kann für die Vertriebsunterstützung eingesetzt werden. Für die nicht wiederverwendbaren Teile waren die investierten Projektressourcen vergeudet und fehlen für die Vertriebsunterstützung-Software.

Beeinflussungsbeziehungen mit Merkmalen
Eine Verschlechterung der Stabilität und Transparenz der Projektziele (EPD-StaTrans-Projektziele) zieht eine Verschlechterung der Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* und der technischen Anforderungen (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien*) nach sich, da Anforderungen sich direkt aus Zielen ableiten und zu ihnen konform sein sollten. Diese Beeinflussung wird proportional eingeschätzt und mit „≈“ vorgeschlagen. Die EPD-StaTrans-Projektziele bewirkt aber umgekehrt nicht unmittelbar die Verbesserung der *EPD-StaTrans-FachlicheAnforderungen* und der *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien,* sie schafft nur eine Voraussetzung hierfür.

Belege aus der Literatur
Transparenz der Projektziele. In vielen Studien wird dargestellt, daß intransparente Projektziele als Risikofaktor in Projekten gesehen werden. Konkret werden „Nicht ausreichend definierte Projektziele“ bei der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04], „unzureichend definierte Projektziele“ bei Daily Telegraph zitiert nach [Gaul04], „Unklare Zieldefinitionen“ bei forsa zitiert nach [Gaul04], „Fehlende und unklare Zielsetzungen des Projektes“ [Gru01], „Unklare Ziele“ [Kel94] und „unklare Zielvorstellung für das System“ [Rup01] genannt. Resultierend hieraus kann „unklarer Projektauftrag“ bei Computerwoche zitiert nach [Gaul04] aufgefasst werden. Der Chaos Report nennt „Clear Vision and Objectives“ [Sta94] als wichtigen Projekterfolgsfaktor. In [Plo02] zitiert nach [Wal04] werden „Unrealistische Erwartungen“ als Projektrisikofaktoren benannt, welche auf überzogenen, nicht hinreichend analysierte Projektziele hindeuten.
Stabilität der Projektziele. Instabile Projektziele werden in der Literatur ebenfalls als Projektrisikofaktor benannt, zum Beispiel „sich ständig ändernde Ziele und Anforderungen“ [Rup01].

2.4.3 Projektthemenbereich Softwareentwicklung

Die Herleitung der Fragen zu Einflußfaktoren aus dem Projektumfeld Risikobereichs „Entwicklungs-Umfeld“ und den Projektthemenbereich „Softwareentwicklung“ zeigt der folgende Ontologieausschnitt in Abbildung 20.

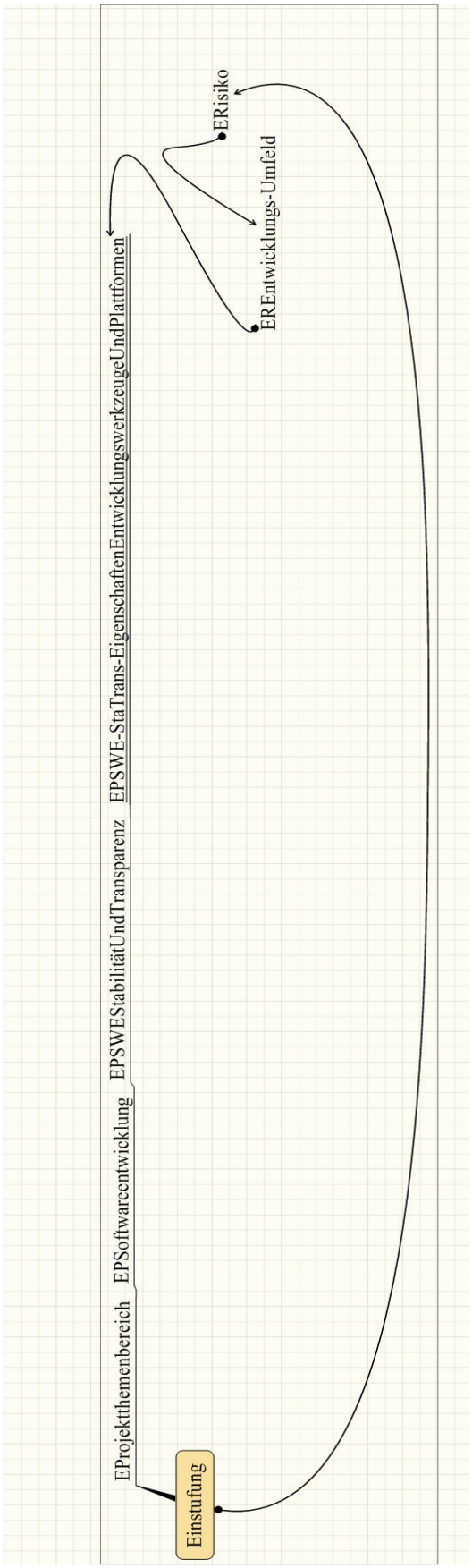

Abbildung 20 Fragen Einfl.Fakt Risikobereich Entwicklungs-Umfeld und Projektthemenbereich SWE

2.4.3.1 Stabilität und Transparenz der Eigenschaften von Werkzeugen

Definition „EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen“: Unter der Stabilität und Transparenz der Eigenschaften von Entwicklungswerkzeugen und -Plattformen sind die Stärken und Beschränkungen von Entwicklungswerkzeugen, Entwicklungsplattformen und Produktivplattformen zu verstehen. Stabilität steht hier für die Beständigkeit und Zuverlässigkeit [Bal98] von Stärken, Transparenz für die Erkennbarkeit von Stärken und Beschränkungen (siehe Glossar im Anhang).
Unterstützungsfrage: Welche Beeinflussungen sind durch EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen möglich und in welchem Ausmaß?
Beispiel: Wenn ein Entwicklungswerkzeug kritische Fehler bei der Eventbehandlung von grafischen Oberflächen erzeugt, ist es teilweise unumgänglich, einen Patch einzuspielen. Dabei kommt es jedoch vor, daß nicht nur bekannte Fehler behoben werden, sondern auch neue Fehler einfliessen, was bewirkt, daß vorher als fehlerhaft bekannte Teile der Applikation zwar dann funktionieren, aber andere bisher bereits funktionierende Teile nun nicht mehr fehlerfrei ablaufen.

Motivation „EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen:“
Bedeutung für Softwareprojekte. Das heutige Entwicklungsumfeld in der Informationstechnologie zeichnet sich durch einen zumeist hohen technischen Innovationsgrad aus, der auch durch schnelle technische Innovationszyklen bei den Entwicklungsumgebungen, CASE-Werkzeugen sowie Entwicklungs- und Zielplattformen verursacht wird.
Diese bedingen Unsicherheiten in Bezug auf die Eigenschaften der verwendeten Werkzeuge in Softwareentwicklungsprojekten. In der Softwareentwicklung ist keine systematisch vorgelagerte Vorentwicklung wie beispielsweise in der Automobilbranche üblich, wie der Autor aus seiner eigenen Industrieerfahrung weiß. Bei einer institutionalisierten und systematischen Vorentwicklung wird die Erprobung neuer Technologien und Verfahren getrennt vom eigentlichen Entwicklungsprojekt durchgeführt. Ihre Ergebnisse werden in Form von Wissens- und Erfahrungsgewinn oder Prototypen der eigentlichen Produktentwicklung zur Verfügung gestellt. In der Softwareentwicklung hingegen wird oft allenfalls im Rahmen einer Machbarkeitsstudie ein exemplarischer Durchstich für eine neue Technologie durchgeführt, aber nicht zwingend deren Eigenschaften umfassend auslotet. Dadurch entsteht einen hohe Intransparenz bezüglich der in einem Entwicklungsprojekt verwendeten Technologie. Durch fehlendes Wissen und Erfahrung bezüglich einer neuen und möglicherweise noch nicht ausgereiften Technologie steigt das Ausmaß an Unwägbarkeiten signifikant an.
Durch die bereits erwähnten schnellen technische Innovationszyklen relativiert sich der Nutzen von technischen Erfahrungswerten und die technischen Unwägbarkeiten für ein Softwareentwicklungsprojekt steigen an. Bei unreifen Entwicklungsständen bei Softwareengineering-Werkzeugen und Produktivplattformen muß mitunter während der Projektlaufzeit die Werkzeug- und Plattformkonfiguration mehrfach geändert werden. Bei jeder Änderung müssen neue Unwägbarkeiten in Kauf genommen werden, die den Projektfortschritt behindern und die Effizienz der eingesetzten Ressourcen vermindern können.
Wenn während des Projektes Werkzeuge oder Plattformen ausgewechselt werden, bedeutet dies Zusatzaufwände für das Umrüsten und das vertraut werden müssen mit den Eigenschaften des neuen technischen Umfeldes. Da diese Aufwände nichts zur Wertschöpfung des Projektes beitragen, resultiert hieraus eine verminderte Effizienz des Res-

sourceneinsatzes.
Funktionalitäten und Beschränkungen der verwendeten Werkzeuge und Plattformen müssen dem Projektteam hinreichend transparent und hinreichend stabil gegen unerwünschte Veränderungen sein, um die Machbarkeit der Anforderungen unter den gegebenen Rahmenbedingungen einschätzen zu können. Falls diese Stabilität und Transparenz nicht gegeben ist, besteht die Gefahr, daß sich im Projektverlauf herausstellt, daß aufgrund technischer Beschränkungen der Werkzeuge und Plattformen Anforderungen nicht erfüllt werden können oder teuere applikatorische Lösungen entwickelt werden müssen. Dies vermindert den Projektnutzen und die Effizienz der eingesetzten Ressourcen Projektbudget, kalendarische Zeit und personelle Kapazität.
Beispiel: Wenn ein national verteiltes Warenwirtschaftssystem eines Großhändlers die Echtzeitsynchronisierung von Veränderungen national verteilter Bestands- und Kundendaten erfordert und hierbei Spalten und Zeilen in der Datenbank abhängig von der Niederlassung geändert werden dürfen, ist hierfür ein sogenannter „vertical split“ beziehungsweise „horizontal split“ durch das Datenbank-Managementsystem hilfreich. Dabei werden Datenbanktabellen spalten- beziehungsweise zeilenweise auf örtlich entfernte Server verteilt und gepflegt, was für die applikatorische Zugriffslogik aber verborgen ist. Hierdurch wird erreicht, daß Daten einer logischen Tabelle physisch an verschiedenen Stellen gepflegt werden können. Sie werden durch das Datenbankmanagementsystem (DBMS) synchronisiert. Die Applikationslogik muß somit die verteilten Datenbanken nicht explizit ansprechen und ist dadurch von den Details der Verteilung unabhängig. Wenn der „vertical“ beziehungsweise „horizontal split“ trotz gegenteiliger Herstelleraussage im Datenbankmanagementsystem nicht funktioniert, stellt die Unkenntnis der Projektleitung und des Auftraggebers über diesen Sachverhalt ein erhebliches Projektrisiko dar. Eine applikatorische Lösung des Problems der verteilten Datenpflege bedeutet erhebliche Aufwandserhöhungen und Terminverschiebungen.

Beeinflussungsbeziehungen mit Merkmalen

Eine Verschlechterung der Stabilität und Transparenz der Eigenschaften von Entwicklungswerkzeugen und Zielplattformen (EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen) vergeudet die Entwicklungsressourcen personelle Kapazität, Budget und Zeit, weil beispielsweise Aufwände investiert werden müssen, um ein Tool zum Funktionieren zu bringen oder Workarounds zu entwickeln. Dadurch verschlechtert sich der Zustand der *EPP-BedDeck-Kap-Personell*, der *EPP-BedDeck-Kap-Projektbudget* und der *EPP-BedDeck-Kap-KalendarischeZeit*. Schlecht funktionierende Werkzeuge gehören zu den Hauptstressfaktoren im Arbeitsleben [DeM99] und verschlechtern auch die *EPM-BedDeck-MotivationImTeam*. Diese Beeinflussungen werden überproportional eingeschätzt und daher mit „≈≈“ vorgeschlagen.

Belege aus der Literatur

Unbekannte Eigenschaften von Entwicklungswerkzeugen, Entwicklungsplattformen und Zielplattformen können ein Softwareentwicklungsprojekt beeinträchtigen. Hinweise auf Probleme in Projekten aus der Literatur wie „Neue Technologie“ [Gaul04], „Anwendung neuer Technologien“ in der KPMG-Studie "IT Runaway Systems" zitiert nach [Gaul04], „Neue Technologie“ bei Jalote zitiert nach [Gaul04], „Überschätzung der technischen Möglichkeiten“ bei Boehm zitiert nach [Gaul04], „Technische Schwierigkeiten“ bei [Plo02] zitiert nach [Wal04] „Gefahr mit Standardsoftware beziehungsweise externen Komponenten“ bei [Boe98] zitiert nach [Wal04], „Gefahr mit Altsystemen“ bei [Boe98] zitiert nach [Wal04] und „unbekannte Einsatzbedingungen“ [Wic00] unter-

mauern diesen Gesichtspunkt. Auch [Wal01] betont die Kritikalität der Entwicklungswerkzeuge und Plattformen für ein Softwareentwicklungsprojekt. „Zu späte Integration“ [Ver00] deutet auf einen zu passivem Umgang mit dem Integrationsrisiko während der Entwicklung einer Software hin. Die Eigenschaften der Zielplattform bleiben hierbei unüberprüft und sind damit nicht hinreichend transparent, um die Lauffähigkeit der Software in ihrer Einsatzumgebung schlußendlich gewährleisten zu können.

2.4.4 Projektthemenbereich Projektmanagement

Die Herleitung der Fragen zu Einflußfaktoren aus dem Projektumfeld Risikobereichs „Entwicklungs-Umfeld“ und den Projektthemenbereich „Projektmanagement“ zeigt der folgende Ontologieausschnitt in Abbildung 21.

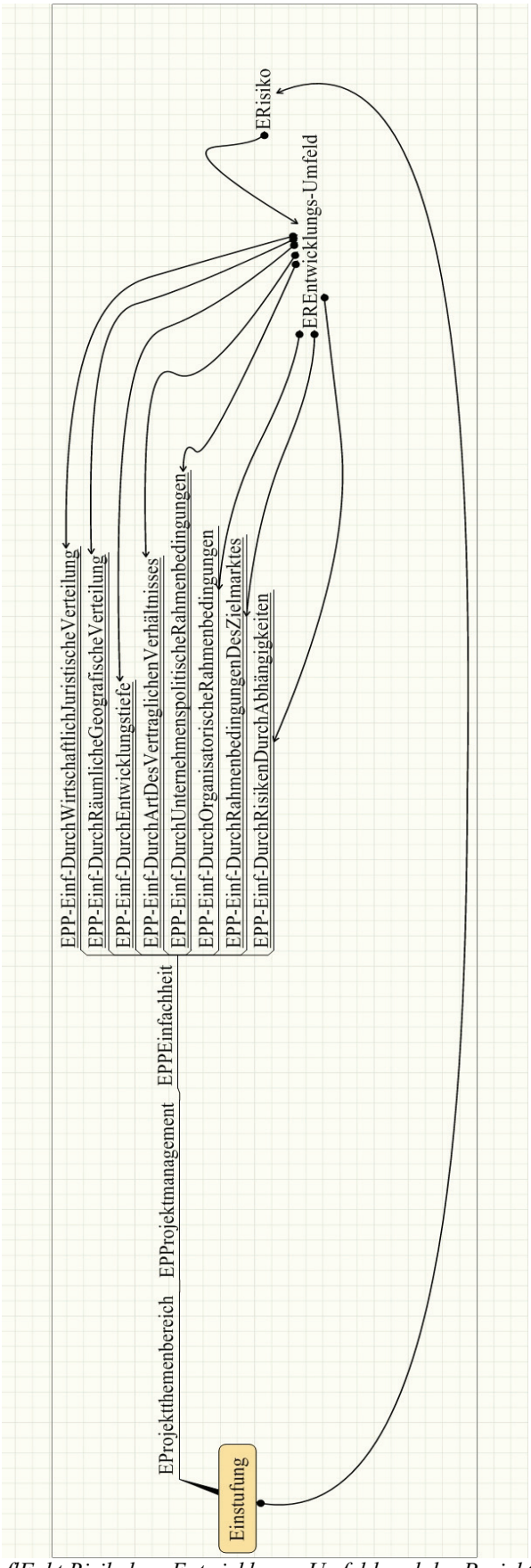

Abbildung 21 Fragen EinflFakt Risikober. Entwicklungs-Umfeld und des Projektthemenber. Projektman.

2.4.4.1 Geografische Verteilung - Offshoring

Definition der „EPP-Einf-DurchRäumlicheGeografischeVerteilung“:
Projekte unterscheiden sich stark nach der räumlichen Verteilung über ein Gebäude bis hin zur geografischen Verteilung im Sinne von Offshoring. Unter EPP-Einf-DurchRäumlicheGeografischeVerteilung soll im folgenden betrachtet werden, ob das gesamte Entwicklungsteam einschließlich Benutzer räumlich konzentriert in einem eigens dafür bereitgestellten Raum arbeitet, oder ob es verteilt über mehrere Stockwerke, Gebäude, Städte, Länder oder Kontinente und Zeitzonen ist und auf wie viele verschiedene Standorte es verteilt wird.
Beispiel: Wenn die Stakeholder eines zu entwickelnden Softwaresystems in einem anderen Stockwerk desselben Gebäudes untergebracht sind als das Entwicklungsteam, kann bereits von einer räumlichen Verteilung gesprochen werden.
Unterstützungsfrage: In welchem Umfang ist EPP-Einf-DurchRäumlicheGeografischeVerteilung gegeben und in welchem Ausmaß beeinflußt dies die in Beziehung stehenden Merkmale?
Motivation der „EPP-Einf-DurchRäumlicheGeografischeVerteilung“:
Bedeutung für Softwareprojekte. EPP-Einf-DurchRäumlicheGeografischeVerteilung behindert Kommunikations- und Informationsflüsse in einer durch ihren Innovationsgehalt, durch ihre Komplexität und ihre Instabilität kommunikations- und kooperationsintensiven Tätigkeit. Verteilung bewirkt Grenzen, die quer durch das Projektteam laufen und eine gemeinsame Vision und Identitätsbildung, sowie den Informationsfluß erschweren.

Kommunikation Die Informationsflüsse zwischen räumlich verteilten Entwicklungsstandorten sind wesentlich schwieriger zu gestalten als bei einem Team in einem Raum. In einem dezidierten Projektraum kann im Bedarfsfall die Kommunikation und der Informationsaustausch im Team völlig ad-hoc und ohne zeitlichen und organisatorischen Overhead erfolgen, während bei multinationalen Projekten unter Umständen bei einer Projektpartei Arbeitszeit ist, während bei der anderen gerade Nachruhe ist, was bewirkt, daß eine synchrone Kommunikation schwieriger zu organisieren ist. Dies hat zur Folge, daß das Team bei weitem nicht so effizient Veränderungen weiter kommunizieren und flexibel darauf reagieren kann.
Bei EPP-Einf-DurchRäumlicheGeografischeVerteilung entsteht an den Schnittstellen der Projektpartner Informationsverlust und Verzögerung der Kommunikation, welche mit steigender Entfernung noch verstärkt werden. Dadurch wird es schwieriger, eine gemeinsame Zielvorstellung für das Projekt zu entwickeln. Falls zusätzlich Sprachbarrieren (noch mehr wenn beide Parteien als Projektsprache eine Fremdsprache benutzen) und unterschiedliche Zeitzonen bestehen, werden die genannten Schwierigkeiten noch verstärkt. Diese Umstände verschärfen das Problem, einen Projektstatus zu beurteilen, da die inoffizielle Zusatzinformation fehlt, die das offizielle Bild vervollständigt und die man sich vor Ort leichter besorgen kann. Die Notwendigkeit und Schwierigkeit von Kommunikation, Koordination und Kontrolle in geografisch verteilten Projekten wird auch in [Hol+06] belegt. Da aktuelle Informationen in einem innovativen und dynamischen Umfeld wichtig für effiziente Entwicklungsarbeit sind, vermindert ihr Fehlen den Wirkungsgrad der eingesetzten Projektressourcen kalendarische Zeit und Projektbudget.

Detaillierungsgrad EPP-Einf-DurchRäumlicheGeografischeVerteilung hat Auswirkungen auf den Bedarf an schriftlichen oder modellhaften Dokumentationen. Es macht einen Unterschied bezüglich des erforderlichen Detaillierungsgrades einer Spezifikation, wen man mit ihrer Umsetzung beauftragt. Wenn man sie für einen In-house

Entwickler schreibt, der in einem Nebengebäude sitzt, kann dieser für informelle Abstimmungen und Präzisierungen problemlos ad-hoc Kontakt aufnehmen. Er kann gegebenenfalls kleinere Lücken und Mehrdeutigkeiten mit Hilfe seines firmenspezifischen Hintergrundwissens leichter ergänzen. Falls man hingegen über Standort-, Sprach-, Mentalitäts- und Zeitzonengrenzen hinweg einen Projektpartner auf der anderen Seite der Erde beauftragt, fehlt diesem der gemeinsame Wissens-, Erfahrungs- und Kultur-Kontext. Mit diesem könnte er Lücken in der Spezifikation ergänzen und zielkonforme eigene Entwurfsentscheidungen treffen. Auch eine Kontaktaufnahme gestaltet sich hier viel aufwendiger.

Identifikation Zudem begünstigt ein in einem Raum konzentriertes Projektteam viel leichter die Bildung eines vertrauensvollen sozialen Klimas mit ehrlicher direkter Kommunikation [Bec00], was einer effizienten Arbeitsweise zuträglich und somit förderlich für den Projekterfolg ist. Die Kooperation und Kommunikation kann stark von geografischen, gesellschaftlichen oder kulturellen Grenzen, Sprachbarrieren und Zeitzonen behindert werden, wie sie durch verteilte Entwicklung entstehen. Mentalitätsunterschiede, unterschiedlicher kultureller Hintergrund, unterschiedliche Begriffe von Pünktlichkeit, Zuverlässigkeit, Umgangsformen, Werten und Geschäftsgebaren erschweren hier die Identitätsbildung. Der Aufbau von Teamgefühl, Identifikation mit dem Projekt, Motivation sowie gegenseitigem Vertrauen und Respekt ist schwieriger. Bei der geografischen Verteilung verstärken sich diese Probleme, je mehr Projektpartner beteiligt sind und und je unerprobter die Zusammenarbeit noch ist.

Beeinflussungsbeziehungen mit Merkmalen

Bei einem Projekt, in dem Offshoring angewendet wird (EPP-Einf-DurchRäumliche-GeografischeVerteilung) erhöht sich der Kommunikationsaufwand (*EPM-BedDeck-KommunikationImTeam),* weil Informationsstände über unterschiedliche Entwicklungsstandorte synchron gehalten werden müssen. Der hierfür erforderliche Koordinationsaufwand verschlechtert die Bedarfsdeckung personeller Kapazitäten (*EPP-BedDeck-Kap-Personell)*.

Wenn die Verteilung der Teammitglieder über unterschiedliche Standorte eine schwierigere Identitätsbildung zur Folge hat, bewirkt dies eine Verschlechterung der Motivation (*EPM-BedDeck-MotivationImTeam)*. Es werden in jedem Fall stärkere Führungsfähigkeiten (*EPM-BedDeck-Führungsfähigkeiten)* und Projektmanagementfähigkeiten (*EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement*) erforderlich.

Durch entstehende Reisekosten kann sich der Zustand der *EPP-BedDeck-Kap-Projektbudget* verschlechtern. Diese Beeinflussungen werden überproportional eingeschätzt und daher mit „≈≈“ vorgeschlagen. Offshoring erhöht den Bedarf nach Unterstützung durch ein detailliertes, umfassendes Arbeitsproduktmodell in den relevanten Projektthemenbereichen.

Belege aus der Literatur

[Gaul02] sieht „Verteilung“ als bedeutsam für Softwareentwicklungsprojekte. Die Darstellung von [Cha02] schafft ein umfassendes Bild von den vielfältigen Problemen, die durch räumlich verteilte Entwicklung entstehen. Er führt hierbei die folgenden konkreten Schwierigkeiten bei verteilten Projekten an: „Communication is more difficult as project team members are at different locations“, „Data information, and documents are often placed in different locations“, „Coordination among project team members can be difficult as they are located in different sites and work in different time zones“, „Diffi-

cult to control and manage the quality of the project tasks performed by different members working on a variety of development environments and from different locations“, „Difficult to control versioning of documents from multiple sites“, „Need to synchronize vast amounts of data from multiple sites over WAN“, „Variety of development environments and platforms on different sites“, „Project control is more difficult as there are more uncertainties and a larger number of changing parameters when more than one site is involved“, „More difficult to search among a large number of documents residing on different sites“, „Project task assignment less efficient and time lag between completion of one task and start of another may be large“, „Difficult to track project status as the project team members are on different sites“, „Monitoring of individual project tasks according to an expected schedule is more difficult“, „Building up of team spirit and sense of togetherness is more difficult“. Der hinderliche Einfluß von zeitlicher, geografischer und soziokultureller Distanz wird auch in [Hol+06] bestätigt. [Smi04] führt als Problemfaktoren die Zeitzonen, Fremdsprachenkenntnisse, unterschiedliche Terminologien, Mißverständnisse bei Arbeitsaufträgen und differierende Organisations- und nationale Kulturen an. [Bir05] betont eine mangelnde Wertekompatibilität. [Win+07] veranschaulichen die Auswirkungen kultureller Einflüsse in Off-Shoring-Projekten auf Kommunikationsverhalten und Effizienz der Projektarbeit. Der Sachverhalt der Verteilung als solches wird bei [Kot02] bereits als Projektrisiko gesehen, was auch [Wal01] betont. Dies bestätigt „Gefahr mit extern ausgeführten Aufgaben“ bei [Boe98] zitiert nach [Wal04], sowie Interessenkonflikte an den „Projektschnittstellen“ bei [Hal98] zitiert nach [Wal04] und unzuverlässige „IT- und Software-Lieferanten“, sowie „Arbeitsumgebung“ (im Sinne von geografische Verteilung) [Wal04].

2.4.4.2 Wirtschaftlich-Juristische Verteilung - Outsourcing

Definition der „EPP-Einf-DurchWirtschaftlichJuristischeVerteilung“:
Es wird betrachtet, ob die Projektpartei, für die ein Vorgehensmodell ausgewählt werden soll, als Auftraggeber oder als Auftragnehmer fungiert, oder ob sie beide Rollen annimmt.
EPP-Einf-DurchWirtschaftlichJuristischeVerteilung im Sinne von Outsourcing betrifft nicht nur Projekte mit verschiedenen juristischen Personen als Projekt- und Vertragspartner, sondern beginnt bereits bei verschiedenen Profit Centern innerhalb einer Unternehmung, da diese aufgrund ihrer wirtschaftlichen Eigenständigkeit ebenfalls konkurrierende wirtschaftliche Interessen vertreten. EPP-Einf-DurchWirtschaftlichJuristischeVerteilung bedeutet, daß zwischen mindestens zwei Projektpartnern ein vertragliches Auftraggeber-Auftragnehmer-Verhältnis besteht. Gegenstand des Vertrages sind zu erbringende Leistungen für das Softwareentwicklungsprojekt in Form der Ressourcen Projektbudget, kalendarische Zeit, personelle Kapazität und in Form von Projektergebnissen, vergleiche auch [Kei05].
Unterstützungsfrage: In welchem Ausmaß ist wirtschaftlich-juristische Verteilung gegeben und in welchem Ausmaß beeinflußt dies die in Beziehung stehenden Merkmale?
Beispiel: Wenn ein Unternehmen ein Softwareentwicklungsprojekt oder Teile davon an ein anderes Unternehmen beauftragt, liegt EPP-Einf-DurchWirtschaftlichJuristischeVerteilung vor.

Motivation der „EPP-Einf-DurchWirtschaftlichJuristischeVerteilung“:
Bedeutung für Softwareprojekte. In der Konstellation Auftraggeber-Auftragnehmer stehen sich wirtschaftlich eigenständige juristische Personen gegenüber, die beide

Gewinnmaximierung betreiben. Sie haben also potentiell das Bestreben, für möglichst wenig Eigenleistung möglichst viel Fremdleistung einzutauschen. Es existieren im Projekt also widersprüchliche Individualziele, die einen Interessenkonflikt bewirken können.
Softwareentwicklung hat aufgrund der technischen Innovationszyklen und der fachlichen Dynamik einen sehr hohen Innovationsgrad, es sind keine umfassenden Standards etabliert, die relevante Belange festlegen wie die DIN und ISO Normen in der klassischen produzierenden Industrie. Eine Vorentwicklung zur Verminderung des Innovationsgrades ist nicht üblich. Daher ist Softwareentwicklung nicht nur eine sehr dynamische und schöpferische, sondern auch eine sehr kooperations, – innovations- und kommunikationsintensive Tätigkeit. Wissen, Erfahrung und Know-how verschiedener Parteien und unterschiedlicher Disziplinen müssen kombiniert und neues Wissen erschlossen werden. Eine „offene, ehrliche Kommunikation“ [Bec00] stellt einen wichtigen Erfolgsfaktor dar. Wissen ist bei der Softwareentwicklung nicht nur eine Projektressource, sondern auch ein Projektergebnis, daß kooperativ erarbeitet werden muß.
EPP-Einf-DurchWirtschaftlichJuristischeVerteilung erzeugt wirtschaftliche Interessenkonflikte welche der erforderliche Kommunikation und Kooperation im Projekt mitunter im Wege stehen. In einer wissenslastigen Tätigkeit wie der Softwareentwicklung werden im Projekt erworbene Kenntnisse zum Wirtschaftsgut, das nicht bereitwillig geteilt wird. Das durch die EPP-Einf-DurchWirtschaftlichJuristischeVerteilung erzeugte Spannungsfeld zwischen den Projekt- und Vertragspartnern zieht sich je nach definiertem Leistungsumfang als Entscheidungs- und Verantwortungsgrenze durch das Projekt. Dieses Spannungsverhältnis und die Grenze bewirken, daß Informationen wechselseitig durch den Filter der individuellen Interessen fliessen, da keine Seite einen Informationsvorteil preisgeben will, obwohl die entstehenden Nachteile für die Projektpartner sich zuletzt auch als Nachteil für das Projekt auswirken.
Dieser Effekt wird verstärkt durch die Auftraggeber-Auftragnehmer-Schnittstelle an sich, da eine Schnittstelle immer einen Informationsverlust mit sich bringt, wenn wie per Definition nur selektiv Information durchgereicht wird. Oftmals wird nach dem One Face-to-the-Customer Prinzip von einem Projektpartner ein dezidierter Ansprechpartner für einen anderen Projektpartner zwischengeschaltet, der einerseits die Informationsflüsse kanalisieren kann, aber andererseits immer eine zusätzliche Station im Kommunikationsfluß darstellt, an der Verzögerungen, Informationsverluste und Mißverständnisse entstehen können. Dies erhöht die Verständigungsschwierigkeiten der Projektpartner und resultiert in erhöhtem Koordinationsaufwand.
Im Konfliktfall müssen noch die entsprechenden Aufwände erbracht werden, um Schuldnachweise zu führen, damit Haftungen zugewiesen werden können. Falls externe Auftragnehmer die ihnen beauftragten Arbeiten aufgrund der dargestellten Situation nicht hinreichend ausführen, entsteht neben dem erhöhten Koordinationsaufwand zusätzlicher Ressourcenbedarf für die Kontrolle der Nachbesserungen oder eigene Nacharbeiten. Dies vermindert die Effizienz der eingesetzten Projektressourcen Zeit und personelle Kapazität, da Arbeiten mitunter doppelt ausgeführt, zumindest aber mit erhöhten Aufwand kontrolliert werden müssen. Auch der Nachweis, daß Spezifikationen durch einen Auftragnehmer fehlerhaft umgesetzt wurden, wodurch Nacharbeiten zu Lasten des Auftragnehmers folgen, ist mitunter zeitaufwendig zu erbringen. Je mehr und je neuer die Unterauftragnehmerverhältnisse sind, desto weniger sind sie überschaubar und eingespielt und desto schwieriger gestaltet sich die Kommunikation und Zusammenarbeit.
Eine schlechte Zusammenarbeit zwischen Projektparteien führt potentiell zu fehlerhaf-

ten Arbeitsprodukten mit den eben diskutierten Nacharbeiten. Dies vermindert die Effizienz der eingesetzten Projektressourcen kalendarische Zeit und personelle Kapazität.
Beispiel: Wenn eine extern entwickelte Dialogkomponente nicht den GUI-Standards im Rahmen des Corporate Designs eines Unternehmens entspricht, müssen entsprechende Nacharbeiten geleistet werden, die potentiell nicht in der Aufwandsschätzung abgedeckt waren.

Zusammenfassung und Abgrenzung Das wesentliche an den genannten Risikofaktoren der juristisch-wirtschaftlichen und der räumlichen und-geografischen Verteilung ist, daß der erforderliche Zusammenhalt und Informationsfluß im Projekt durch sie beeinträchtigt wird. Während bei wirtschaftlich-juristischer Verteilung oft das Phänomen des nicht-hinreichend-kommunizieren-wollens dominiert, steht bei räumlichen und geografischer Verteilung häufig das nicht-hinreichend-kommunizieren-können im Vordergrund.
Beispiel: Wenn eine Änderung einer zentralen Anforderung, die viele Standorte betrifft, über verschiedene Kontinente, Sprach- und Zeitzonen kommuniziert wird und ihre Auswirkungen diskutiert und transparent gemacht werden muß, kann dies sehr viel aufwendiger werden als bei einem in einem Raum lokalisierten Team mit einem kurzfristig einberufenen Meeting.
Die Information über eine Architekturänderung kann in einem Projektraum verbal kommuniziert und gegebenenfalls durch sofortige Rückfragen betroffener Entwickler präzisiert werden. In einem multinationalen Projekt muß sie sehr genau formuliert und asynchron per Mail oder vermittels eines Repository verteilt werden. Aufgrund von Zeitzonengrenzen kann sich durch Rückfragen die Wiederherstellung der Synchronizität der Arbeitsprodukte aber mehrere Tage hinziehen. Sie stellt also größere Anforderungen an die Projektthemenbereiche Projekt-, Konfigurations- und Change Management mit den nötigen Aktivitäten, Rollen und Arbeitsprodukten.

Beeinflussungsbeziehungen mit Merkmalen

In einem Projekt, in dem Outsourcing praktiziert wird (EPP-Einf-DurchWirtschaftlichJuristischeVerteilung), verschlechtert sich mitunter die Kommunikation zwischen den Projektparteien (*EPM-BedDeck-KommunikationImTeam)*, wenn die Projektpartner nicht offen kommunizieren um Eigeninteressen, zum Beispiel Know-how, zu schützen.
Durch erhöhten Koordinationsaufwand zwischen den Projektparteien verschlechtert sich die *EPP-BedDeck-Kap-Personell* und der Bedarf an Projektmanagementfähigkeiten (*EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement*) erhöht sich. Diese Beeinflussungen werden überproportional eingeschätzt und daher mit „≈≈“ vorgeschlagen.

Belege aus der Literatur

[Gaul04] benennt die Existenz „Externer Auftragnehmer“ als wichtiges Projektkennzeichen. Hierdurch wird deutlich, daß es als bedeutender Unterschied gesehen wird, ob mehrere Vertragspartner in einem Projekt zusammenarbeiten oder ein Projekt nur innerhalb einer Organisation abgewickelt wird. Konkreter sind die Risikofaktoren „Schlechte Leistung des Lieferanten von Hardware und Software“ aus der KPMG-Studie "IT Runaway Systems", zitiert nach [Gaul04], „Mängel an extern gelieferten Komponenten“, „Mängel an extern durchgeführten Aufgaben“ bei Boehm, zitiert nach [Gaul04] und „Probleme durch inkompatible Softwareprodukte verschiedener Lieferanten“ [Gru01].Dies wird durch „Gefahr mit extern ausgeführten Aufgaben“ bei [Boe98] zitiert nach [Wal04], sowie durch „Interessenkonflikte an den „Projektschnitt-

stellen“ bei [Hal98] zitiert nach [Wal04] und unzuverlässige „IT- und Software-Lieferanten“ [Wal04] bestätigt. „Mangelnde Zusammenarbeit zwischen Kunde und Softwarelieferant“ wird bei Jalote zitiert nach [Gaul04] angeführt, bestätigt durch „Kooperativer Umgang Auftraggeber und Auftragnehmer“ als kritischen Erfolgsfaktor des Projektmanagements bei [ISA00] zitiert nach [Wal04]. Auch die betonte Konzentration des V-Modell XT auf die Auftraggeber-Auftragnehmer-Schnittstelle [VMX06] ist ein Hinweis darauf, daß Verteilung im Sinne von Outsourcing relevant für die Charakterisierung von Softwareprojekten ist.

2.4.4.3 Art des vertraglichen Verhältnisses

Definition der „EPP-Einf-DurchArtDesVertraglichenVerhältnisses“:
Mit der Art des vertraglichen Verhältnisses wird betrachtet, ob zwischen Projektpartnern eine Dienstleistung- beziehungsweise eine Werklieferung vereinbart ist. Dienstleistung bedeutet, daß der Auftragnehmer beauftragt ist, ein fixiertes Stundenkontingent zu erbringen und berechtigt ist, dieses abzurechnen. Das Ergebnis ist hier variabel. Werklieferung bedeutet, daß der Auftragnehmer beauftragt ist, eine spezifiziertes Gewerk in Form eines Softwaresystems oder Teilen davon zu liefern und berechtigt ist, ein vorher fixiertes Entgelt abzurechen. Die hierfür zu erbringenden Aufwände sind variabel.
Beispiel: Ein Auftragnehmer kann beauftragt werden, einhundert Programmierstunden für eine Datenbankzugriffsschicht zu erbringen oder er kann beauftragt werden, eine Datenbankzugriffsschicht für einen Festpreis zu entwickeln.
Fragen zu Einflußfaktoren aus dem Projektumfeld:
Welches wirtschaftliche Risiko und welche Haftung entstehen durch die Vertragsart im Projekt, in welchem Ausmaß?
Welche Auswirkungen auf die Stabilität und Transparenz der Anforderungen entsteht durch die Vertragsart im Projekt?

Motivation der „EPP-Einf-DurchArtDesVertraglichenVerhältnisses“:
Bedeutung für Softwareprojekte. Bei einem Festpreisprojekt lastet das wirtschaftliche Risiko hauptsächlich auf dem Auftragnehmer, der an der Erfüllung der Spezifikation gemessen wird. Der Subunternehmer braucht für einen Festpreis einen detaillierte, verbindliche und stabile Spezifikation als Schätz- und Vertragsgrundlage. Nachträgliche Änderungen bedürfen eines kontrollierten Change Managements damit die wirtschaftliche Situation für den Auftragnehmer beherrschbar bleibt. Für den Auftraggeber bedeutet ein Werklieferungsvertrag weniger Verantwortung und Risiko als ein Dienstleistungsvertrag, er bietet aber auch weniger Flexibilität.
Bei einem Aufwandsprojekt hingegen liegt der Hauptteil des Risikos und der Verantwortung beim Auftraggeber. Da jede nachträgliche Änderung Zusatzaufwand für den Auftraggeber bedeutet, wird er versuchen, diese soweit wie möglich zu beschränken. Für den Auftragnehmer hingegen bedeuten Änderungen Zusatzumsatz, er wird daher nach der Maxime „Embrace Change“ [Bec00], [AGI06] auf Anforderungsänderungen reagieren.
Beispiel: Ein Auftraggeber kann einen Auftragnehmer bezüglich einer Datenbankzugriffsschicht zur Mitwirkung gegen Aufwand oder zur Erstellung beauftragen. Im ersten Fall kann man mit einer eher rudimentären Spezifikation starten und über Feedback Schleifen intern informell iterieren, um sich in jeder Iteration an das optimale Ergebnis anzunähern.
Im zweiten Fall ist diese Vorgehensweise zumindest bei einem Werklieferungsvertrag

nicht für alle Beteiligten sinnvoll, da sie für den Subunternehmer inakzeptable Risiken beinhaltet, über den vereinbarten Festpreis hinausgehende Kosten aus eigenen Mitteln bezahlen zu müssen.

Beeinflussungsbeziehungen mit Merkmalen
Risiken durch die Art des vertraglichen Verhältnisses (EPP-Einf-DurchArtDesVertraglichenVerhältnisses) können beispielsweise bei einem Festpreisvertrag die Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen*) verschlechtern. Dies tritt auf, wenn der Auftraggeber Definitionslücken bei den Anforderungen dazu benutzt, Änderungswünsche zu kaschieren und somit als im Festpreis abgegolten darzustellen. Bei einem Aufwandsprojekt wird er hingegen eher anstreben, mit stabilen und transparenten Anforderungen in das Projekt zu starten, weil jede Änderung zusätzliche Kosten bedeuten. Diese Beeinflussung hängt stark von der projektspezifischen Vertragsgestaltung ab. Sie wird proportional eingeschätzt und mit „≈“ vorgeschlagen.

Belege aus der Praxis und der Literatur
Mehrere Diskussionen mit ebenfalls industrieerfahrenen Kollegen, speziell mit Gerd Beneken, sind die Quelle für die EPP-Einf-DurchArtDesVertraglichenVerhältnisses als Einflußfaktor auf Softwareentwicklungsprojekte [Ben06]. Auch in [Ben06a] und [Kei05] wird die Thematik der Vertragsverhältnisse in Softwareentwicklungsprojekten und ihre Auswirkungen auf die Projektparteien aufgegriffen.

2.4.4.4 Entwicklungstiefe

Definition der „EPP-Einf-DurchEntwicklungstiefe“:
Die Entwicklungstiefe besagt, bezogen auf welche Disziplinen des Entwicklungszyklusses eine Projektpartei die Verantwortung bekommt. Die Verantwortung in Bezug auf die Projektdisziplinen beschreibt, welche Projektthemenbereiche (siehe Kapitel 3.2.2) sie dadurch abdecken muß. Einfachheit steht für den sich aus der Entwicklungstiefe ergebenden Schwierigkeitsgrad der Projektarbeit.
Beispiel: Eine Projektpartei kann für fachliche oder technische Konzeptarbeit, Implementierung, Test, Integration, oder ausschließlich für das Subcontractor Management verantwortlich sein.
Fragen zu Einflußfaktoren aus dem Projektumfeld:
Welche Fachliche Verantwortung entsteht durch die Entwicklungstiefe im Projekt?
Welche Software Engineering Verantwortung entsteht durch die Entwicklungstiefe im Projekt?
Welche Software Management Verantwortung entsteht durch die Entwicklungstiefe im Projekt?
Welche Projektmanagement Verantwortung entsteht durch die Entwicklungstiefe im Projekt?

Motivation der „EPP-Einf-DurchEntwicklungstiefe“:
Bedeutung für Softwareprojekte. Durch Verteilung jedweder Art decken die Projektpartner oft unterschiedliche Entwicklungstiefen und Projektthemenbereiche im Projekt ab. Hierdurch variieren die zu unterstützenden Tätigkeiten, die zu erstellenden Arbeitsprodukte, die zu übernehmenden Verantwortungen und die zu kontrollierenden und bedienenden Schnittstellen für eine Projektpartei. Hierdurch können sich Problem-

schwerpunkte verschieben, was in unterschiedlichen Verantwortlichkeiten für Projektthemenbereiche im Lebenszyklus und somit unterschiedlichen Anforderungen an einen Entwicklungsprozeß resultiert.
Beispiel: Es ist für eine Projektpartei ein Unterschied hinsichtlich der für sie anfallenden Aufgaben, Tätigkeiten, Verantwortlichkeiten und Vorgehensweisen, ob sie ein Arbeitsprodukt wie eine Architektur selbst entwickelt, oder für einen Auftragnehmer spezifiziert und das Arbeitsprodukt gegen die Spezifikation kontrolliert. Im ersten Fall muß überwiegend Engineering Tätigkeit abgedeckt werden, während im zweiten Fall das Subcontractor Management im Vordergrund steht.
Speziell in der Automobilindustrie ist nach der Erfahrung des Autors der vorliegenden Arbeit ein starker Rückzug auf die Kernkompetenzen der Herstellerunternehmen zu beobachten. Wenn der Auftraggeber im Bereich der Softwareentwicklung ausschließlich als Systemintegrator oder „kompetenter Auftraggeber" wirkt und sich auf das Subcontractor Management beschränkt, leisten die Subunternehmer in überwiegendem Ausmaß die Software Engineering Arbeit.

Beeinflussungsbeziehungen mit Merkmalen

Die Entwicklungstiefe bewirkt Verantwortlichkeiten für Projektthemenbereiche. Je nach Entwicklungstiefe der betreffenden Projektpartei erhöhen sich die Fähigkeitsanforderungen bezüglich fachlicher, Software Engineering, Software Management und Projektmanagement. Bei gleichbleibenden verfügbaren Fähigkeiten verschlechtert sich die *EPD-BedDeck-Fähigk-Fachlich*, die *EPSWE-BedDeck-Fähigk-SoftwareEngineering*, die *EPSWE-Deck-Fähigk-SoftwareManagement* oder die *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement*. Diese Beeinflussungen hängen stark von der projektspezifischen Entwicklungstiefe ab. Sie werden proportional eingeschätzt und mit „≈" vorgeschlagen.

Belege aus der Literatur

Der Tailoring Mechanismus des V-Modell XT fragt als Projektmerkmal den abzudeckenden „Systemlebenszyklusausschnitt" [VMX06] ab. Dadurch kann es eine geeignete Auswahl an Vorgehensbausteinen zur Unterstützung der abzudeckenden Projektthemenbereiche im relevanten Lebenszyklusausschnitt vorschlagen. Auch die Konzentration des V-Modell XT auf die Auftraggeber-Auftragnehmer-Schnittstelle [VMX06] ist ein Hinweis darauf, daß Projektparteien oft nicht das gesamte Spektrum der Projektthemenbereiche abdecken, da Auftraggeber und Auftragnehmer unterschiedliche Teile des Entwicklungszyklus und somit unterschiedliche Projektthemenbereiche abdecken. Auch „Gefahr mit Standardsoftware beziehungsweise externen Komponenten" bei [Boe98] zitiert nach [Wal04] belegt, daß die Entwicklungstiefe Einfluß auf die Erfolgschancen eines Softwareprojektes hat. Dies bestätigt „Gefahr mit extern ausgeführten Aufgaben" bei [Boe98] zitiert nach [Wal04], unzuverlässige „IT- und Software-Lieferanten" [Wal04] und Interessenkonflikte an den „Projektschnittstellen" bei [Hal98] zitiert nach [Wal04]. „Kooperativer Umgang Auftraggeber und Auftragnehmer" wird bei [ISA00] zitiert nach [Wal04] als kritischen Erfolgsfaktor des Projektmanagements gesehen.

2.4.4.5 Unternehmenspolitische Rahmenbedingungen

Definition der „EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen":
Unter diesen wird die Gesamtheit der Tatsachen und Maßnahmen verstanden, die ver-

schiedenen Ziele der Stakeholder im Unternehmen zu erreichen, Interessen zu vertreten und zu wahren.
Beispiel: Die Positionierung des eigenen Organisationsbereiches als Technologieführer im Unternehmen kann ebenso ein Ziel der internen Unternehmenspolitik sein wie signifikanter Personalabbau.
Fragen zu Einflußfaktoren aus dem Projektumfeld:
Welche Beeinflussungen auf das Projekt sind durch die EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen zu erwarten und in welchem Ausmaß?
Welches Ausmaß an Management Unterstützung besteht für das Projekt?
Welches Ausmaß an Widerständen besteht aufgrund von Interessenkonflikten gegen Projekt?

Motivation der „EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen“:
Bedeutung für Softwareprojekte. Verschiedene Interessengruppen im Unternehmen vertreten unterschiedliche Ziele, die sich widersprechen können. Aus Zielkonflikten können Widerstände gegen das Projekt und seine Ergebnisse resultieren. Falls sich durch ein Projekt oder dessen Ergebnis Personen oder Gruppen im Unternehmen hinsichtlich ihrer Interessen beziehungsweise ihrer Ziele bedroht oder beeinträchtigt fühlen, kann es zu aktivem oder passiven Widerständen gegen das Projekt kommen. Dieser kann sich im verschweigen oder verzögern von Informationen, im verweigern oder verzögern von Ressourcen bis hin zur verdeckten oder offenen Einflußnahme gegen das Projekt beziehungsweise sein Ergebnis äußern (siehe auch [Moh98]).
Beispiel: Die politischen Rahmenbedingungen, von denen ein Softwareprojekt umgeben ist, haben mitunter Auswirkung auf dessen Ressourcensituation. Ein Projekt, welches die Aufmerksamkeit und Unterstützung des höheren Linienmanagements genießt, wird speziell im Fall von Ressourcenkonflikten eher eine benötigte Ressource zugeteilt bekommen, als ein Projekt, welches diese Unterstützung aus verschiedenen Gründen nicht genießt. Falls ein Datenbankprogrammierer mit hundert Prozent seiner Kapazität gleichzeitig in zwei Projekten auf dem kritischen Pfad benötigt wird, ist zu erwarten, daß ein weniger populäres Projekt diesen nicht im erforderlichen Maß zugeteilt bekommt und dadurch in Terminverzug gerät.

Beeinflussungsbeziehungen mit Merkmalen
Die politischen Rahmenbedingungen innerhalb einer softwareentwickelnden Organisation (EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen) bedingen Unterstützung oder Widerstände bezüglich eines Projektes, beispielsweise durch das gehobene Management. Dadurch kann sich die Ressourcensituation bezüglich Geld, personeller Kapazität aber auch projektrelevanter Fähigkeiten verschlechtern, was sich negativ auf die *EPP-BedDeck-Kap-Projektbudget*, *EPP-BedDeck-Kap-Personell*, die *EPD-BedDeck-Fähigk-Fachlich*, die *EPSWE-BedDeck-Fähigk-SoftwareEngineering*, die *EPSWE-BedDeck-Fähigk-SoftwareManagement*, die *EPP-BedDeck-Kap-Projektbudget* oder die *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement* auswirken kann. Diese Beeinflussungen hängen stark von den projektspezifisch gegebenen Rahmenbedingungen in einem Unternehmen ab. Sie werden proportional eingeschätzt und mit „≈“ vorgeschlagen.

Belege aus der Literatur
Die politischen Rahmenbedingungen, die ein Projekt umgeben, werden in der Literatur als Risikofaktor für Projekte gewertet wie folgende Punkte zeigen: „Unternehmenspoli-

tik“ in der Daily Telegraph Studie zitiert nach [Gaul04], „Zielübereinstimmung“ [Gaul04], „Schwacher Zusammenhang mit Geschäftsstrategie“ und „Mangelnde Einbeziehung und Unterstützung durch das Management“ in der KPMG-Studie „What went wrong?“ 97 zitiert nach [Gau04], „Fehlende Unterstützung für den Projektmanager“ in der Daily Telegraph Studie zitiert nach [Gaul02], „Mangelnde Unterstützung des Projektes durch die Geschäftsleitung“ [Gru01], „Unterstützung durch das Business“ [Gaul02], „Unternehmensinterne Widerstände“ in der forsa Studie zitiert nach [Gaul04], „Widerstand durch die zukünftigen Benutzer“ [Kel94], „Kompetenzgerangel und Intrigenwirtschaft zwischen Projektteam und Anwendern“ [Gru01], „Konflikte zwischen DV- und Fachabteilung“ [Kel94], „Interne Gründe“ bei [Plo02] zitiert nach [Wal04], oder „Abteilungsdenken mit „Scheuklappen“ [Gru01]. Der Chaos Report der Standish Group empfiehlt als Projekterfolgskriterium „Executive Management Support“ [Stan94], bei [ISA00] zitiert nach [Wal04] wird ebenso „Geschäftsleitung unterstützt Projekte“ als kritischen Erfolgsfaktor des Projektmanagements dargestellt.

2.4.4.6 Organisatorische Rahmenbedingungen

Definition der „EPP-Einf-DurchOrganisatorischeRahmenbedingungen“:
Unter diesen werden die Einfüsse auf das Projekt verstanden, die durch die Organisation des Unternehmens, in dessen Räumlichkeiten die Projektarbeit durchgeführt wird, entstehen.
Beispiel: Die Eignung der Arbeitsplätze des Projektteams in Bezug auf ungestörtes Arbeiten oder die personelle Fragmentierung durch Multiprojekttätigkeit der Projektmitarbeiter sind EPP-Einf-DurchOrganisatorischeRahmenbedingungen.
Fragen zu Einflußfaktoren aus dem Projektumfeld:
Welche Beeinflussungen auf das Projekt sind durch die EPP-Einf-DurchOrganisatorischeRahmenbedingungen zu erwarten und in welchem Ausmaß?
Welche Beeinflussungen entstehen für das Projekt durch Störungen und Beeinträchtigungen am Arbeitsplatz des Projektteams?
Welche Beeinflussungen entstehen für das Projekt durch Fragmentierung durch Multiprojekttätigkeit beziehungsweise Fluktuation der Projektmitglieder?

Motivation der „EPP-Einf-DurchOrganisatorischeRahmenbedingungen“:
Bedeutung für Softwareprojekte. Die EPP-Einf-DurchOrganisatorischeRahmenbedingungen beeinflussen die Erfolgschancen eines Projektes, da die Mitarbeiter diesen täglich und unmittelbar ausgesetzt sind. Günstige Rahmenbedingungen fördern Konzentration, Motivation und Effizienz der Mitarbeiter und damit auch die Effizienz der eingesetzten Ressourcen kalendarische Zeit, Projektbudget und personelle Kapazität, ungünstige wie Lärmbelästigung vermindern sie. Speziell Tom DeMarco [DeM99] thematisiert immer wieder die Auswirkungen speziell des Arbeitsplatzes, der Lärmbelästung, sowie ständiger Unterbrechungen und ihre Auswirkungen auf die Effizienz der Projektmitarbeiter. Da Softwareentwicklung eine sehr innovationslastige und komplexe Tätigkeit darstellt, erfordert sie eine hohe Konzentration durch die Projektmitarbeiter. Es erfordert laut DeMarco im Mittel ungefähr fünfzehn Minuten, um ein Maß an Konzentration erreicht zu haben, daß ausreichend für Entwicklungstätigkeiten ist. Dieses Hineindenken in Sachverhalte können Menschen laut DeMarco nicht beliebig oft an einem Tag bewerkstelligen. Ein hoch konzentriertes Arbeiten gelingt Menschen nicht beliebig lange und nicht zu jedem beliebigen Zeitpunkt. Dies hat zur Auswirkung, daß Störungen und Unterbrechungen aus dem Arbeitsumfeld eines Projektmitarbeiters die Effizienz des

Einsatzes der Ressource personelle Kapazität signifikant verschlechtern können.
Beispiel: Falls ein Projektmitarbeiter ständig durch Störungen aus seiner Konzentration für die Projektarbeit gerissen wird, kann er diese nur unter erschwerten Bedingungen zügig und erfolgreich durchführen (vergleiche auch [DeM99]). Hierdurch wird die Effizienz des Einsatzes der Projektressource personelle Kapazität vermindert.

Beeinflussungsbeziehungen mit Merkmalen
Ungünstige organisatorische Rahmenbedingungen (EPP-Einf-DurchOrganisatorischeRahmenbedingungen) wie beispielsweise enge Arbeitsplätze oder ständige Störungen vermindern die Effizienz der für das Projekt eingesetzten Ressourcen Personal, Budget und Zeit. Sie verschlechtern so die *EPP-BedDeck-Kap-Personell,* die *EPP-BedDeck-Kap-Projektbudget* und die *EPP-BedDeck-Kap-KalendarischeZeit* (siehe Beispiel oben).
Weiterhin bewirken sie eine Demotivierung der Projektmitarbeiter (*EPM-BedDeck-MotivationImTeam),* wenn Erfolge, da diese unter schlechten Rahmenbedingungen schwieriger zu erzielen sind, ausbleiben. Diese Beeinflussung hängt stark von den projektspezifischen organisatorischen Rahmenbedingungen ab. Sie werden proportional eingeschätzt und mit „≈“ vorgeschlagen.

Belege aus der Literatur
Die EPP-Einf-DurchOrganisatorischeRahmenbedingungen sind eine bedeutungsvolle Rahmenbedingung für Projekte. „Laute und enge Arbeitsplätze“ bei McConnel zitiert nach [Gaul04] und „Wichtigkeit des geeigneten Arbeitsumfeldes“ [DeM99] belegen diesen Sachverhalt. DeMarco bezeichnet ständige Störung und Unterbrechungen als zwei der Hauptstressfaktoren im modernen Arbeitsleben [DeM99]. Auch [Wal01] sieht die Arbeitsplatzgestaltung als kritischen Erfolgsfaktor für Softwareprojekte. Ein Mindestmaß an organisationaler Stabilität ist auch eine Voraussetzung für die mit der Einführung eines Vorgehensmodells verbundenen Prozeßveränderung [Rai01].

2.4.4.7 Rahmenbedingungen des Zielmarktes

Definition der „EPP-Einf-DurchRahmenbedingungenDesZielmarktes“:
Unter den Rahmenbedingungen des Zielmarktes ist der Konkurrenzdruck am Markt zu verstehen, gegen den sich das Projekt oder sein Ergebnis behaupten muß.
Beispiel: Wenn ein Unternehmen ein Produkt entwickelt, muß dieses hinsichtlich seines Preises, seiner Qualität, seiner Funktionalität und seines Einführungstermins an der Konkurrenz orientiert sein.
Fragen zu Einflußfaktoren aus dem Projektumfeld:
Welche Beeinflussungen auf das Projekt sind durch die EPP-Einf-DurchRahmenbedingungenDesZielmarktes zu erwarten und in welchem Ausmaß?
In welchem Ausmaß und auf welche Art beeinflußt Konkurrenzdruck das Projekt hinsichtlich Liefertermin, Budget, Qualität und Funktionsumfang?

Motivation der „EPP-Einf-DurchRahmenbedingungenDesZielmarktes“:
Bedeutung für Softwareprojekte. Da die kommerzielle Softwareentwicklung mit dem Ziel durchgeführt wird, entweder Entwicklungsdienstleistungen oder Softwareprodukte am Markt zu plazieren und aufgewendete Investitionen durch Erlöse zu amortisieren, sind die EPP-Einf-DurchRahmenbedingungenDesZielmarktes für solche Vorhaben von großer Bedeutung. Durch den Konkurrenzdruck werden Lösungsspielräume für ein

Projekt enger, die Anzahl und Stringenz der Restriktionen für das Projekt steigt. Insbesondere der Konkurrenzdruck hinsichtlich Preis, Funktionalität, Qualität oder Liefertermin engen die Spielräume in Projekten ein. Ein Produkt oder eine Dienstleistung, die teurer, qualitativ schlechter oder später als die Konkurrenz am Markt angeboten wird, hat schlechtere Chancen, von Kunden nachgefragt zu werden, wirtschaftlich erfolgreich zu sein und sich zu amortisieren.
Beispiel: In der Mobiltelefon-Sparte stehen die verschiedenen Hersteller hinsichtlich Liefertermin, Preis, Funktionalität und Qualität miteinander im Wettbewerb.

Beeinflussungsbeziehungen mit Merkmalen

Ungünstige Rahmenbedingungen des Zielmarktes (EPP-Einf-DurchRahmenbedingungenDesZielmarktes) beispielsweise bezüglich des Zeitpunktes der Markteinführung, des Verkaufspreises, der Features oder von Querschnittseigenschaften wie Performance eines Konkurrenzproduktes führen zu Restriktionen im Projekt.
Dadurch kann sich für das eigene Produkt die verfügbare Entwicklungszeit (*EPP-BedDeck-Kap-KalendarischeZeit),* das verfügbare Entwicklungsbudget (*EPP-BedDeck-Kap-Projektbudget)*, der Bedarf an personeller Kapazität (*EPP-BedDeck-Kap-Personell)* verringern. Der erforderliche Funktionsumfang (*EPD-Einf-DurchUmfangFunktionen)* und die erforderlichen Querschnittseigenschaften (*EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien)*, beispielsweise hinsichtlich Stabilität oder Performance können sich mitunter erhöhen (s.o.). Diese Beeinflussungen hängen stark von der Restriktivität der Rahmenbedingungen des Zielmarktes ab. Sie werden proportional eingeschätzt und mit „≈“ vorgeschlagen.

Belege aus der Literatur

„Äußere Einflüsse“ bei [Plo02] zitiert nach [Wal04], können durch die Rahmenbedingungen des Zielmarktes auf das Projekt einwirken und die Projektsituation terminlich, oder finanziell verschlechtern. Weitere Hinweise auf die Einflüsse durch die Rahmenbedingungen des Zielmarktes auf ein Softwareprojekt geben [Mou+98] und [Rag+05].

2.4.4.8 Risiken durch Abhängigkeiten

Definition der „EPP-Einf-DurchRisikenDurchAbhängigkeiten“:
Risiken stehen für potentielle oder antizipierte Probleme, die den Projekterfolg gefährden können. Abhängigkeiten stehen für Sachverhalte, die nicht innerhalb des Projektes entschieden werden, aber den Projekterfolg beeinflussen können. Abhängigkeiten sind deshalb Risiken, weil sich durch sie Entwicklungen ergeben können, die den Erfordernissen und Zielen des Projektes zuwiderlaufen und nicht hinreichend beeinflußt werden können.
Beispiel: Technische Abhängigkeiten sind zu bedienende Schnittstellen oder zu befolgende Standards. Organisatorische, politische, juristische oder wirtschaftliche Abhängigkeiten sind externe Entscheidungsbefugnisse bezüglich des Projektes.

Fragen zu Einflußfaktoren aus dem Projektumfeld:

Welche Beeinflussungen bestehen für das Projekt durch Risiken wegen fachlicher, technischer, organisatorischer, politischer, wirtschaftlicher oder juristischer Abhängigkeiten und in welchem Ausmaß?
Welche Beeinflussungen entstehen für das Projekt durch die Abhängigkeiten von Zulieferern und in welchem Ausmaß?

Motivation der „EPP-Einf-DurchRisikenDurchAbhängigkeiten“:
Bedeutung für Softwareprojekte. Risiken durch externe Abhängigkeiten sind per Definition innerhalb eines Projektes nicht hinreichend beeinfluß- beziehungsweise kontrollierbar. Sie müssen bei der Durchführungsentscheidung eines Projektes mit einer Abschätzung der Eintrittswahrscheinlichkeit und der Auswirkung bewertet und akzeptiert werden. Risiken und Abhängigkeiten erfordern meist hohen Ressourceneinsatz um sie abzusichern oder um ihre Auswirkungen zu kompensieren. Die Investition dieser Ressourcen trägt nicht direkt und sichtbar etwas zur Funktionalität oder Qualität des Projektproduktes bei und muß den Entscheidungsträgern oft erst plausibel gemacht werden.
Beispiel: In einem Projekt zur Entwicklung eines deutschlandweit einzusetzenden Warenwirtschaftssystems vom Datenbankmanagementsystem wird dringend die Funktionalität eines „vertical“ und „horizontal split“ benötigt. Wenn dies vom Hersteller erst für ein nicht rechtzeitig verfügbares Release zugesagt ist, werden im Projekt Aufwände für Risikomanagement im Sinn der Entwicklung einer applikatorischen Fallbackstrategie erforderlich.

Beeinflussungsbeziehungen mit Merkmalen
Risiken durch bestehende Abhängigkeiten, beispielsweise bezüglich verbindlicher Standards oder Gesetze, sind aufwendig abzusichern und vermindern dadurch die Effizienz der eingesetzten Projektressourcen Personal, Zeit und Geld. Dies verschlechtert bei gleichbleibender Ressourcensituation den Zustand von *EPP-BedDeck-Kap-KalendarischeZeit, EPP-BedDeck-Kap-Projektbudget* und *EPP-BedDeck-Kap-Personell.*
Hohe Risiken durch bestehende Abhängigkeiten in Bezug auf Anforderungen beeinträchtigen auch die fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)*, sowie die technischen Anforderungen (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien).* Diese Beeinflussungen werden proportional eingeschätzt und mit „≈“ vorgeschlagen.

Belege aus der Literatur
Projekte sind per Definition risikobehaftet [Kel94]. Unterschiedlich ist allerdings das Ausmaß der Risiken, die Probleme für Projekte verheißen. Speziell die Abhängigkeiten, denen ein Projekt ausgesetzt ist, implizieren Projektrisiken. In der Literatur findet dies vielfache Beachtung: „Außergewöhnliche Risiken“ [Gaul02] im Sinne von externe Risiken, „Externe Abhängigkeiten“ [Gaul02], „Externe Einflüsse“ bei der forsa-Studie zitiert nach [Gaul04], „Externe Entscheidungen“ bei Jalote zitiert nach [Gaul04], „Schnittstellenprobleme“ [Gru01], „Anpassungsprobleme bei Standardsoftware“ [Gru01], „Gefahr mit Standardsoftware beziehungsweise externen Komponenten“ bei [Boe98] zitiert nach [Wal04] und „Projekterschwernis durch IT-Altlasten“ [Gru01] vermitteln ein Bild davon. Auch „Gefahr mit extern ausgeführten Aufgaben“ bei [Boe98] zitiert nach [Wal04], unzuverlässige „IT- und Software-Lieferanten“ [Wal04], sowie Interessenkonflikte an den „Projektschnittstellen“ bei [Hal98] zitiert nach [Wal04], sowie „Kooperativer Umgang Auftraggeber und Auftragnehmer“ als kritischen Erfolgsfaktor des Projektmanagements bei [ISA00] zitiert nach [Wal04] bestätigen das Risiko durch Abhängigkeiten von Subunternehmern. Auch in [Wal01] werden Abhängigkeiten hinsichtlich unterschiedlichster Aspekte als Projektrisiko dargestellt.

2.4.5 Projektthemenbereich Menschenführung

Die Herleitung der Fragen zu Einflußfaktoren aus dem Projektumfeld Risikobereichs „Entwicklungs-Umfeld“ und des Projektthemenbereichs „Menschenführung“ zeigt der folgende Ontologieausschnitt in Abbildung 22.

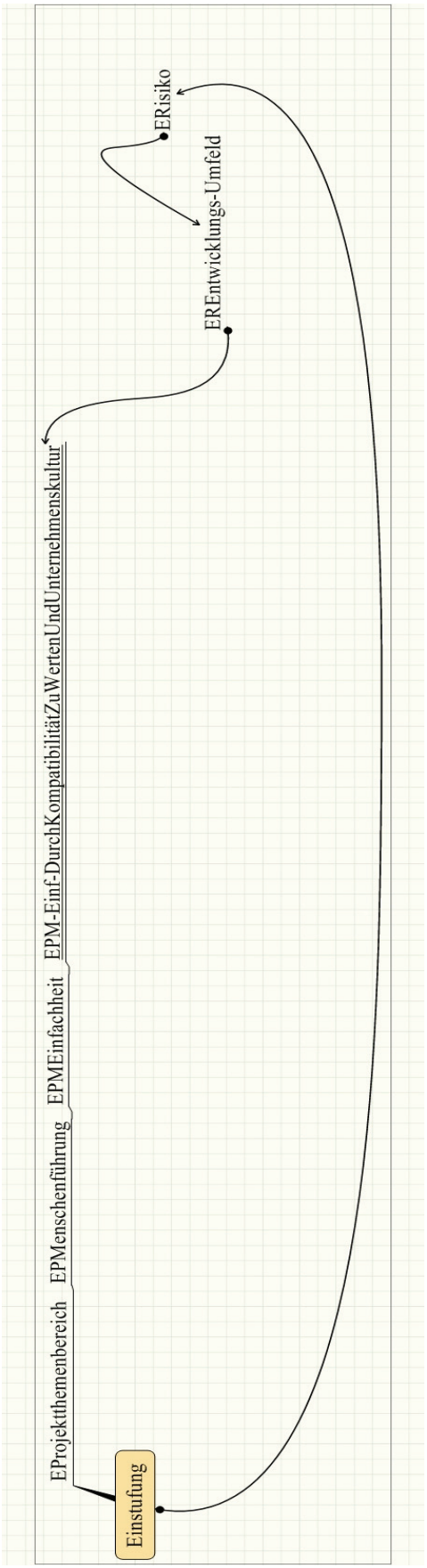

Abbildung 22 Fragen Einfl.Fakt. Risikober. Entw.-Umfeld und Projektthemenber. Menschenführ.

2.4.5.1 Kompatibilität zu Werten und Unternehmenskultur

Definition der „EPM-Einf-DurchKompatibilitätZuWertenUndUnternehmenskultur“:
Unter Werten werden in der vorliegenden Arbeit erstrebenswerte Zustände von Größen verstanden, die über individuellen Nutzen hinausgehen (siehe auch Glossar im Anhang). Werte verbinden sich zu einem Wertesystem und einer Kultur. Diese steht für die Sichtweisen und Gewohnheiten, die im Unternehmen etabliert sind und mit denen sich die Majorität identifizieren kann [Ros03]. Werte stehen im Einklang zur vorherrschenden Mentalität und Kultur. Werte werden durch Ziele konkretisiert. Kompatibilität steht für die Vereinbarkeit verschiedener Werte und Kulturen.
Beispiel: Werte können Mitarbeiterzufriedenheit, Qualitätsbewußtsein, offene Kommunikation oder Kundenorientierung sein.
Fragen zu Einflußfaktoren aus dem Projektumfeld:
Welche Beeinflussungen auf das Projekt sind durch Kompatibilitätsdefizite bezüglich der Werte und Kultur in der Organisation zu erwarten und in welchem Ausmaß?
Welche Beeinflussungen bestehen für das Projekt durch das Maß der Kompatibilität seines Entwicklungsprozesses und seines Entwicklungsergebnisses zu Werten und Unternehmenskultur?

Motivation der „EPM-Einf-DurchKompatibilitätZuWertenUndUnternehmenskultur“:
Bedeutung für Softwareprojekte. Die fehlende Kompatibilität von Projektinhalten zu Werten und Kultur in einer Organisation führt zu Widerständen unter den Mitarbeitern. Wertekompatibilität fördert die Identifikation der Projektmitarbeiter mit dem Projekt und damit deren Motivation. Wenn die Werte und die Kultur eines Unternehmens durch ein Projekt verletzt werden, sind emotionale Widerstände der Mitarbeiter zu erwarten. Dies wirkt sich schädlich auf die Motivation und das Engagement der Mitarbeiter aus.
Beispiel: Wenn Software für ein Rüstungsprojekt durch ein Systemhaus mit pazifistisch ausgerichtetem Werteverständnis entwickelt werden soll, verschlechtert dies die Motivation der Projektmitarbeiter.

Beeinflussungsbeziehungen mit Merkmalen
Mangelnde Kompatibilität von Projektinhalten und Ergebnissen zu Werten und Kultur in einer Organisation verschlechtert die Motivation der Projektmitarbeiter (*EPM-BedDeck-MotivationImTeam)*.
Widerstände erhöhen den Bedarf an Führungsfähigkeiten, um positiv auf die Mitarbeiter einzuwirken. Daher verschlechtert sich die *EPM-BedDeck-Führungsfähigkeiten*. Da diese Beeinflussungen stark von der jeweilig projektspezifisch gegebenen Unternehmenskultur abhängen, werden sie proportional eingeschätzt und mit „≈“ vorgeschlagen.

Belege aus der Literatur
Die Unternehmenskultur und deren Werte sind eine wichtige Rahmenbedingung für Projekte: [Gaul02] benennt hier die Punkte „Organisationskultur“ [Ver00], „Zu hohe Dokumentenorientierung“ im Sinne einer Kultur des Mißtrauens und der Absicherung. „Unternehmenskultur der Schuldzuweisungen“ in der Daily Telegraph Studie zitiert nach [Gaul04], sowie „Kultur der Angst“ [DeM01]. [Wal01] verdeutlicht die Unternehmenskultur als bedeutsam für den Projekterfolg. Mit „Cultural Diversity“ [Sia02] wird die Bedeutsamkeit von Kultur am Beispiel von deren Auswirkungen auf das Qualitätsmanagement gezeigt. Kultur beeinflusst Werte was sich auf organisatorische Zielsetzun-

gen und Entscheidungsprozesse auswirkt. (Original: „Culture influences Values which affects organizational objectives and decision making processes“) [Hof81]. Werte sind im Zentrum der Organisationskultur (Original: „Values are in the center of organizational culture“) [Hof+90]. [Win+07] betonen die Auswirkungen von kulturellen Differenzen in Projekten. Eine umfassende Darstellung kultureller Diversifikationen in Organisationen findet sich in [Hof05].
Die Werte und Kultur in einem Unternehmen sind also eine bedeutungsvolle Rahmenbedingung und ein Risikofaktor für Projekte. Es ist wichtig, daß Ziele und Vorgehensweisen in Projekten kompatibel zu den etablierten Werten und zur Kultur einer Organisation sind, da sonst Widerstände entstehen können.

2.5 Überblick über die Systematik für Projekte

Die Merkmale für die Modellierung von Projekten, sowie die Fragen zu Einflußfaktoren aus dem Projektumfeld für ihre Auswahl und Bewertung wurden in den vergangenen Abschnitten erarbeitet. In der nachfolgenden Tabelle 2 werden sie noch einmal im Überblick dargestellt.

Risikobereich Projektthemen bereich	***Entwicklungs-Produkt Merkmale***	***Entwicklungs-Team Merkmale***	***Entwicklungs-Umfeld Fragen zu Einflußfaktoren aus dem Projektumfeld***
Domänenwissen	• EPD-Einf-DurchUmfang-Funktionen • EPD-StaTrans-FachlicheAnforderungen	• EPD-BedDeck-Fähigk-Fachlich	• EPD-StaTrans-Projektziele
Softwareentwicklung	• EPSWE-Einf-DurchUmfangTechnische-AnforderungenUndQualitätskriterien • EPSWE-StaTrans-TechnischeAnforderungenUnd-Qualitätskriterien	• EPSWE-BedDeck-Fähigk-SoftwareEngineering • EPSWE-BedDeck-Fähigk-SoftwareManagement	• EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUnd-Plattformen
Projektmanagement	• EPP-BedDeck-Kap-Projektbudget • EPP-BedDeck-Kap-KalendarischeZeit	• EPP-BedDeck-Fähigk-OrganisatorischesUnd-KaufmännischesProjektmanagement • EPP-Einf-DurchTeamgröße • EPP-BedDeck-Kap-Personell	• EPP-Einf-DurchWirtschaftlichJuristischeVerteilung • EPP-Einf-DurchRäumlicheGeografischeVerteilung • EPP-Einf-DurchEntwicklungstiefe • EPP-Einf-DurchArtDesVertraglichenVerhältnisses • EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen • EPP-Einf-DurchOrganisatorischeRahmenbedingungen • EPP-Einf-EPP-Einf-DurchRahmenbedingungenDesZielmarktes • EPP-Einf-DurchRisikenDurchAbhängigkeiten

Risikobereich Projektthemen bereich	***Entwicklungs-Produkt Merkmale***	***Entwicklungs-Team Merkmale***	***Entwicklungs-Umfeld Fragen zu Einflußfaktoren aus dem Projektumfeld***
Menschenführung		• EPM-BedDeck-MotivationImTeam • EPM-BedDeck-KommunikationImTeam • EPM-BedDeck-Führungsfähigkeiten	• EPM-Einf-DurchKompatibilitätZuWerten-UndUnternehmenskultur

Tabelle 2 Projektmerkmale und Fragen zu Einflußfaktoren aus dem Entwicklungs-Umfeld

Die Herleitung der Merkmale der Risikobereiche „Entwicklungs-Team“ und „Entwicklungs-Produkt“, sowie der Fragen zu Einflußfaktoren aus dem Projektumfeld des Risikobereichs „Entwicklungs-Umfeld“ aus den Projektthemenbereichen „Domänenwissen“, „Softwareentwicklung“, „Projektmanagement“ und „Menschenführung“ zeigt die folgende Ontologie in Abbildung 23.

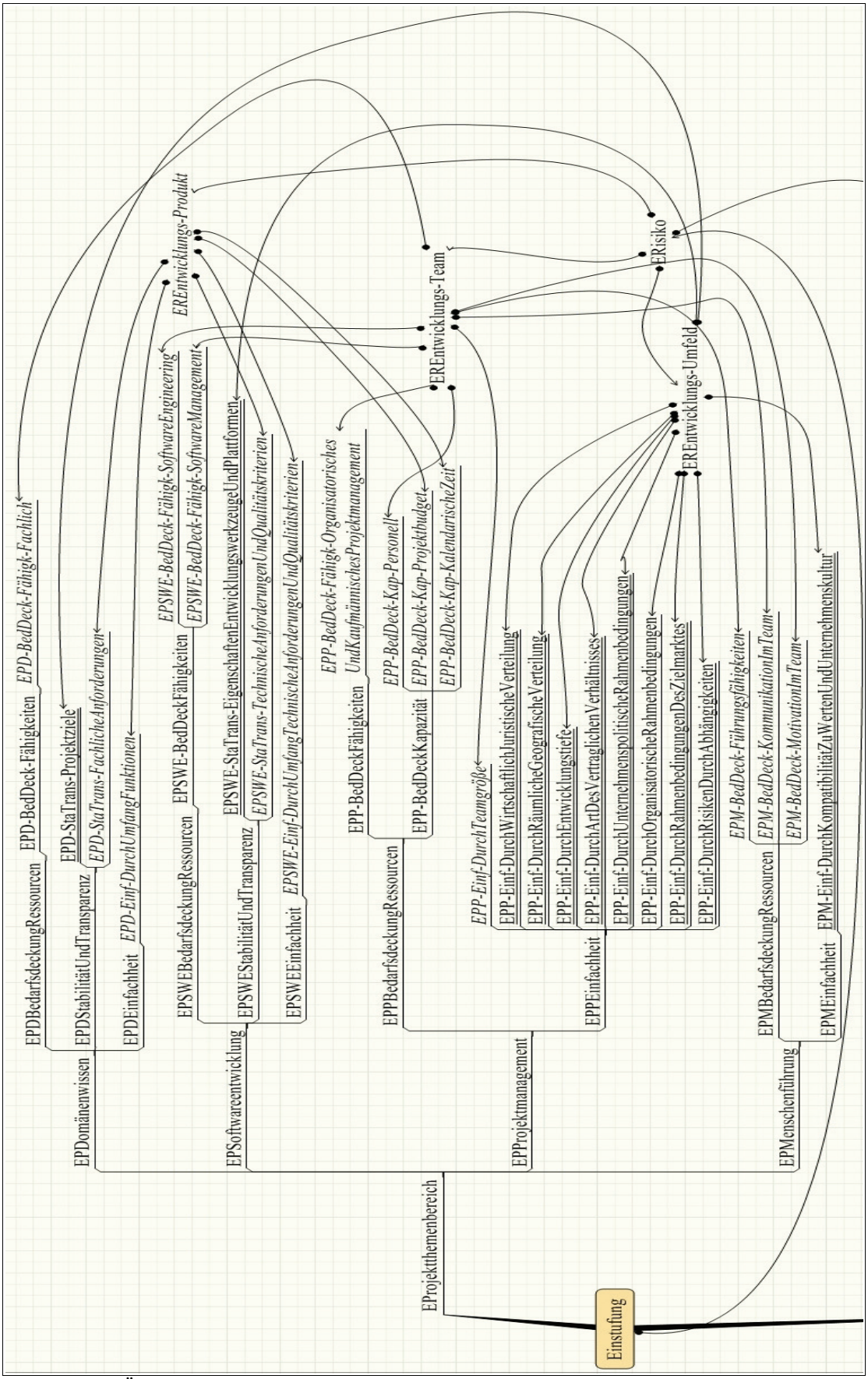

Abbildung 23 Überblick über Projektmerkmale und Fragen zu Einflußfaktoren aus dem Projektumfeld

2.5 Überblick über die Systematik für Projekte

In diesem Kapitel wurden Merkmale zur Repräsentation von wesentlichen Risiken in Projekten, sowie Fragen zur Untersuchung des Projektumfeldes hergeleitet, begründet und definiert. Sie wurden zueinander in Beziehung gesetzt und Ausblicke auf die positive oder negative Beeinflussung der Projektmerkmale durch Vorgehensmodelle gegeben. Diese Beeinflussung durch Vorgehensmodellmerkmale und deren Beeinflussungsbeziehungen wird im folgenden Kapitel beschrieben.

3 Charakterisierung von Vorgehensmodellen

In diesem Kapitel werden Merkmale von Vorgehensmodellen erarbeitet. Die Vorgehensmodellmerkmale in diesem Kapitel sind aus den in Kapitel 2 „Charakterisierung von Projekten“ durch die Projektmerkmale repräsentierten Problemschwerpunkten in Softwareentwicklungsprojekten abgeleitet. Diese werden in Kapitel 4 „Das Modellierungsverfahren im EVGM“ verwendet. Sie unterstützen Zustandsverbesserungen der Projektmerkmale.

Den Anfang des Kapitels bildet eine kompakten Zusammenfassung seines Inhalts. Die detaillierten Herleitungen und Begründungen in den ihr nachfolgenden Abschnitten ersetzt die Zusammenfassung jedoch nicht. Dort wird jedes Vorgehensmodellmerkmal mit Hilfe von Quellen belegt, seine Relevanz für die Problemstellung verdeutlicht, sowie definiert. Die Merkmale sind in eine Systematik eingeordnet (Kapitel 3.1). Diese Systematik verdeutlicht, welche Projektthemenbereiche aus der Systematik aus Kapitel 2„Charakterisierung von Projekten“ von Vorgehensmodellen abgedeckt werden. Mit Hilfe der Vorgehensmodellmerkmale (siehe Kapitel 3.2) werden in Kapitel 4 „Das Modellierungsverfahren im EVGM“ Anforderung an ein projektspezifisch ideales Vorgehensmodell formuliert.

Die vorliegende Arbeit berücksichtigt die Vorgehensmodelle Extreme Programming, Rational Unified Process, Scrum und V-Modell XT. Für alle Vorgehensmodellmerkmale werden pro berücksichtigtem Vorgehensmodell spezifische Bewertungen vergeben (Kapitel 3.3).

Zusammenfassung. Für die Charakterisierung von Vorgehensmodellen beschreibt das EVGM folgende Merkmale. Die ontologischen Bezeichner der Vorgehensmodellmerkmale sind *kursiv* und in Klammern dargestellt.

Vorgehensmodelle unterstützen die Mitarbeiter in einem Softwareentwicklungsprojekt bei ihren Arbeiten in den verschiedenen Projektthemenbereichen. Daher enthalten die Bezeichner der meisten Vorgehensmodellmerkmale das Präfix „Unterstützung Projektthemenbereich“. Dieses drückt aus, wie umfassend und detailliert ein Vorgehensmodell einen Projektthemenbereich durch die Beschreibung von Aktivitäten, Arbeitsprodukten, Rollen, Prinzipien oder Methoden beschreibt, beziehungsweise festlegt.

Die Unterstützung Projektthemenbereich Requirements Engineering (*EV-UnterstProj-TheBer-RequirementsEngineering)* repräsentiert, wie umfassend und detailliert ein Vorgehensmodell die systematische Erhebung und Strukturierung der fachlichen und technischen Anforderungen an das Projektergebnis beschreibt. Dadurch verbessert diese Disziplin beispielsweise die *EPD-BedDeck-Fähigk-Fachlich* (siehe Kapitel 2)*,* weil sie das Anreichern und Dokumentieren fachlichen Wissens unterstützt. Sie erhöht auch die *EPD-StaTrans-FachlicheAnforderungen* und die *EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien* (siehe Kapitel 2), da sie durch gründliche Analyse zu fundierteren und beständigeren Ergebnissen kommt, sowie durch systematisches Erheben von Anforderungen alle relevanten Aspekte abdeckt und somit Lücken in der Anforderungsdefinition vermindert.

Mit der Unterstützung Themenbereich Software Engineering (*EV-UnterstProjTheBer-SoftwareEngineering)* wird hingegen ausgedrückt, wie ausführlich und weitreichend in einem Vorgehensmodell die Entwicklung der sofwaretechnischen Projektergebnisse dargestellt ist. Durch diese Disziplin wird beispielsweise die *EPSWE-BedDeck-Fähigk-SoftwareEngineering* (siehe Kapitel 2) verbessert, weil fehlendes Wissen der Projektmitarbeiter durch Vorgehensmodellbeschreibungen ergänzt wird. Sie erhöht auch die

EPD-Einf-DurchUmfangFunktionen und die *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien* (siehe Kapitel 2), da sie einen systematischeren und übersichtlicheren Entwurf des Gesamtsystems unterstützt.
Ergänzend repräsentiert die Unterstützung Projektthemenbereich Software Management (*EV-UnterstProjTheBer-SoftwareManagement)*, wie ausführlich und vollständig ein Vorgehensmodell den Einsatz der Software Engineering Techniken im Rahmen einer geordneten Vorgehensweise unterstützt. Sie beeinflußt somit zum Beispiel die *EPSWE-Deck-Fähigk-SoftwareManagement* (siehe Kapitel 2).
Im Gegensatz zu den softwarelastigen Unterstützungsdisziplinen stellt die Unterstützung Projektthemenbereich Projektmanagement (*EV-UnterstProjTheBer-ProjektManagement)* dar, wie gut ein Vorgehensmodell die Behandlung der kaufmännisch-administrativen Belange des Projektes abdeckt. Diese Disziplin unterstützt beispielsweise die *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement,* da es fehlendes diesbezügliches Know-how der Projektmitarbeiter durch Vorgehensmodellbeschreibungen kompensiert. Sie erhöht ebenfalls die *EPP-Einf-DurchTeamgröße*, weil sie der Projektleitung Maßnahmen zur besseren Organisation und Koordination der Entwickler an die Hand gibt. Auch die *EPP-BedDeck-Kap-Personell,* die *EPP-BedDeck-Kap-KalendarischeZeit*, sowie die *EPP-BedDeck-Kap-Projektbudget* (siehe Kapitel 2) werden durch die *EV-UnterstProjTheBer-ProjektManagement* gesteigert, da sie einen effizienten Umgang mit diesen Ressourcen unterstützt.
Ergänzend zu diesen vergleichsweise rationalen Aspekten wird auch die Unterstützung von weichen Faktoren mit einbezogen. Die Berücksichtigung des Projektthemenbereichs Menschenführung ist ausschließlich wegen Mitarbeiter im Projekt erforderlich. Die Unterstützung der Menschenführung (*EV-UnterstProjTheBer-Menschenführung)* zeigt an, wie stark ein Vorgehensmodell den Umgang mit menschlichen Eigenschaften anspricht. Durch diese Disziplin werden die *EPM-BedDeck-MotivationImTeam*, die *EPM-BedDeck-KommunikationImTeam,* sowie die *EPM-BedDeck-Führungsfähigkeiten* (siehe Kapitel 2) adressiert.
Vorgehensmodelle unterstützen mehr oder weniger umfassend die Projektthemenbereiche durch Festlegungen von Abläufen und Arbeitsprodukten. Je weitreichender diese unterstützenden Festlegungen durch ein Vorgehensmodell erfolgen, desto mehr schränken sie den Entscheidungsspielraum der Projektbeteiligten ein. Diese Einschränkungen sind ist oftmals in Projekten nur begrenzt sinnvoll, weil auf fortwährende Veränderungen, wie beispielsweise von Anforderungen, reagiert werden muß. Daher betrachtet das Merkmal *EV-Flexib-DurchIterativitätUndInkrementalität,* welche Gestaltungsmöglichkeiten ein Vorgehensmodell hinsichtlich des mehrmaligen, Projektthemenbereiche übergreifenden Durchlaufens von Entwicklungsschritten, sowie des schrittweisen Erweiterns und Ergänzens von Entwicklungs-Arbeitsprodukten erlaubt.
Detaillierte Darstellungen der oben aufgezählten Vorgehensmodellmerkmale können in den folgenden Abschnitten nachgelesen werden.
Sie werden in der folgenden Abbildung 24 noch einmal im Überblick dargestellt.

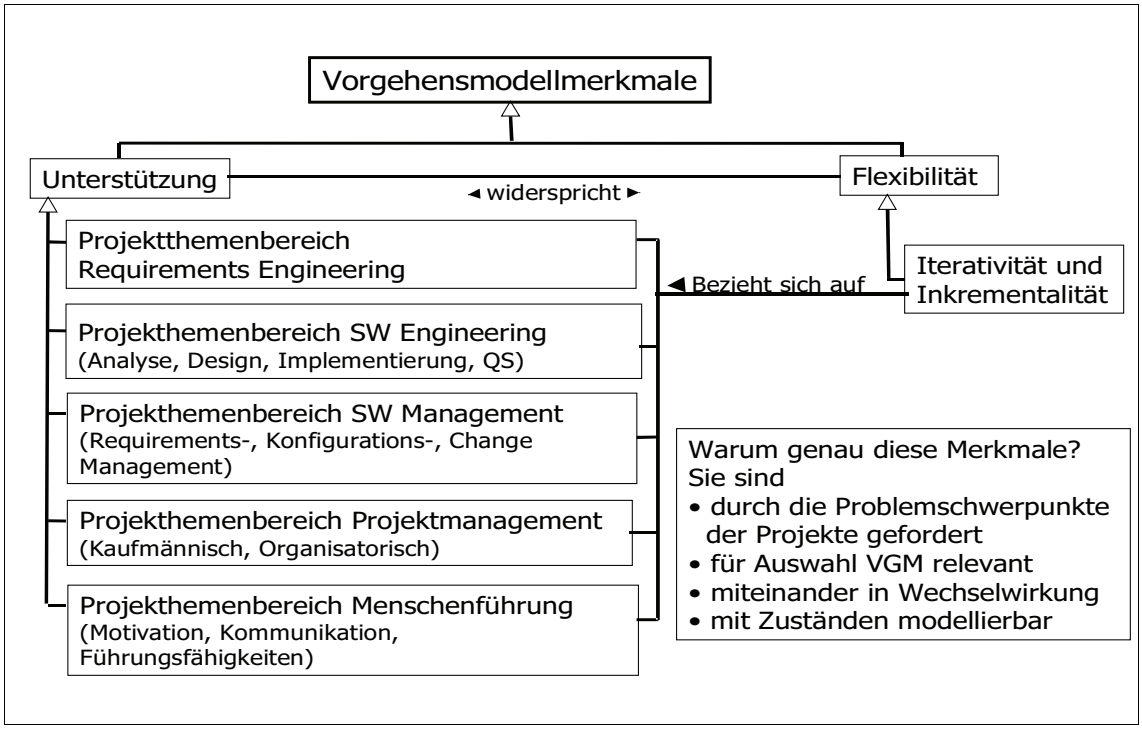

Abbildung 24 Vorgehensmodellmerkmale und Beeinflussungsbeziehungen

Inhalt des Kapitels

3.1 Die Definition der Vorgehensmodellmerkmale

Die Vorgehensmodellmerkmale und ihre vorgehensmodellspezifischen Bewertungen sind dem Risikobereich Entwicklungs-Prozeß zuzuordnen, welcher in Kapitel 2 motiviert wurde. Die Vorgehensmodellmerkmale decken die Risikobereiche und Projektthemenbereiche von Projekten aus Kapitel 2 ab. Um die Überdeckung des Themenbereiches der Vorgehensmodelle zu zeigen, sind sie in eine Systematik eingeordnet, welche die Projektthemenbereiche abdeckt.

3.1.1 Unterstützung Projektthemenbereiche in Vorgehensmodellen

Die Projektthemenbereiche heißen Domänenwissen, Softwareentwicklung, Projektmanagement und Menschenführung entsprechend der Definition in Kapitel 2.
Der Projektthemenbereich „Domänenwissen“ aus Kapitel 2 findet seine Entsprechung im Kapitel 3 durch die Unterstützung durch „Requirements Engineering“. Die Unterstützung der „Softwareentwicklung“ wird diversifiziert in „Software Engineering“ und „Software Management“. Die Unterstützung von „Projektmanagement“ und „Menschenführung“ haben identische Entsprechungen, wie nachfolgende Tabelle 3 noch einmal veranschaulicht.

Projektthemenbereich in der Systematik für Projekte	*Unterstützungen der Projektthemenbereiche durch Vorgehensmodelle*
Domänenwissen	Requirements Engineering
Softwareentwicklung	Software Engineering, Software Management
Projektmanagement	Projektmanagement
Menschenführung	Menschenführung

Tabelle 3 Entsprechungen der Projektthemenbereiche durch Vorgehensmodelle

Projektthemenbereiche repräsentieren die unterschiedlichen Themenfelder, die im Lebenszyklus eines Softwareprojektes abzudecken sind. Vorgehensmodelle unterscheiden sich nach ihrer Überdeckung von Projektthemenbereichen [Chr92].
Die Unterstützungen der Projektthemenbereiche aus Kapitel 2 werden folgendermaßen definiert:

- Requirements Engineering
- Software Engineering (Analyse, Design, Implementierung, Qualitätssicherung)
- Software Management (Requirements Management, Konfigurationsmanagement, Change Management)
- Projektmanagement (kaufmännisch, organisatorisch mit Planung, Kontrolle und Steuerung von Terminen, Arbeitsprodukten, Ressourcen, Aktivitäten, Budget, Risiken)
- Menschenführung (Motivation, Kommunikation, Führung)

Die genauen Definitionen können im Glossar im Anhang nachgelesen werden.
Das Qualitätsmanagement im Sinne des Glossars im Anhang ist projektübergreifend und daher nicht im Fokus der vorliegenden Arbeit.

3.1.2 Modellierbarkeit der Vorgehensmodellmerkmale

Die folgenden Vorgehensmodellmerkmale und ihre vorgehensmodellspezifischen Bewertungen der sind dazu gedacht, den Zustand projektgefährdend bewerteten Projektmerkmale aus Kapitel 2 positiv zu beeinflussen. Daher enthalten die Bezeichner der Merkmale die Bestandteile „Unterstützung“ oder als „Flexibilisierung“.
Unterstützung steht hierbei für die Bereitstellung von Hinweisen und Vorgaben bezüglich von Rollen, Aktivitäten, Arbeitsprodukten, Methoden und Prinzipien durch Vorgehensmodelle, die helfen das Softwareentwicklungsprojekt zu bewältigen. Unterstützung adressiert beispielsweise eine mangelnde „Bedarfsdeckung“ oder „Einfachheit“ aus Kapitel 2.
Flexibilität steht für die Eignung von Vorgehensmodellen, mit Unsicherheiten umzugehen. Flexibilität adressiert beispielsweise eine mangelnde „Stabilität und Transparenz“ aus Kapitel 2. Auch diese Systematisierung orientiert sich an den Problemschwerpunkten aus Kapitel 2.
In der Ontologie wird Unterstützung mit „Unterst“, Flexibilität mit „Flexib“ abgekürzt.
Der Herleitung der Vorgehensmodellmerkmale aus der Ontologie wird in der nachfolgenden Abbildung 25 verdeutlicht.

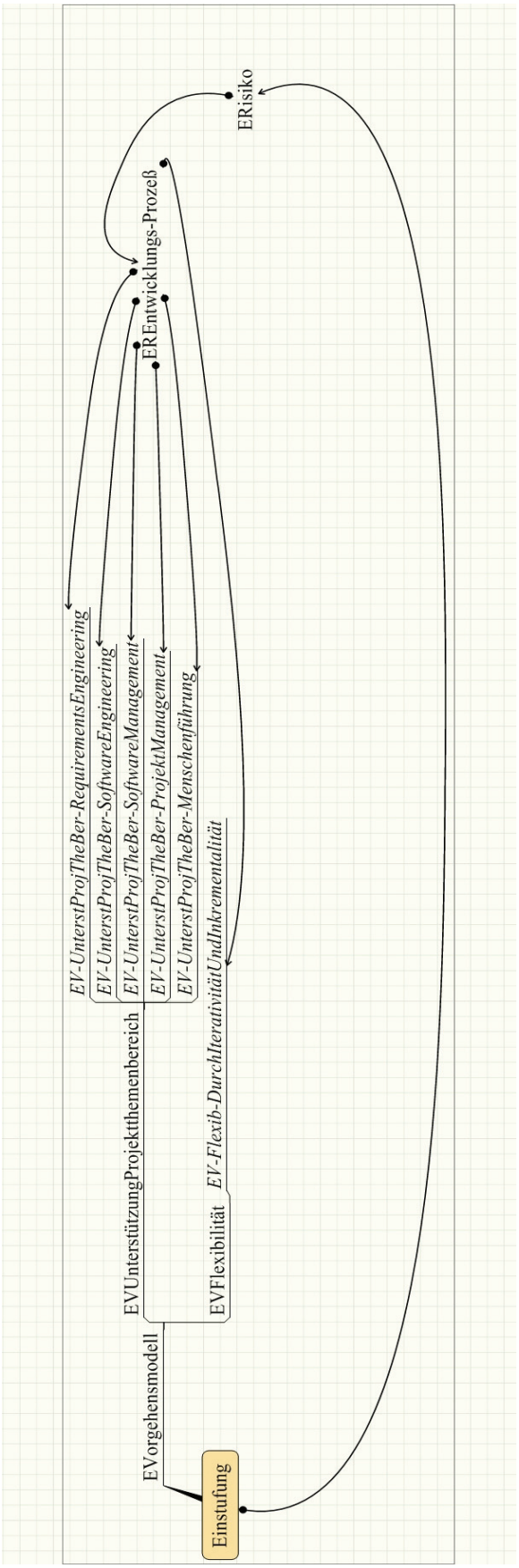

Abbildung 25 Vorgehensmodellmerkmale

3.2 Die Vorgehensmodellmerkmale

Im folgenden werden die Merkmale von Vorgehensmodellen beschrieben.

Beschreibung der Vorgehensmodellmerkmale Jedes Vorgehensmodellmerkmal wird definiert. Daraufhin wird es anhand seiner Bedeutung für Softwareprojekte, für die Eignungseinstufung von Vorgehensmodellen, sowie anhand seiner Beeinflussungsbeziehungen zu Projektmerkmalen motiviert. Die Beschreibungsstruktur der Vorgehensmodellmerkmale verdeutlicht die folgende Abbildung 26.

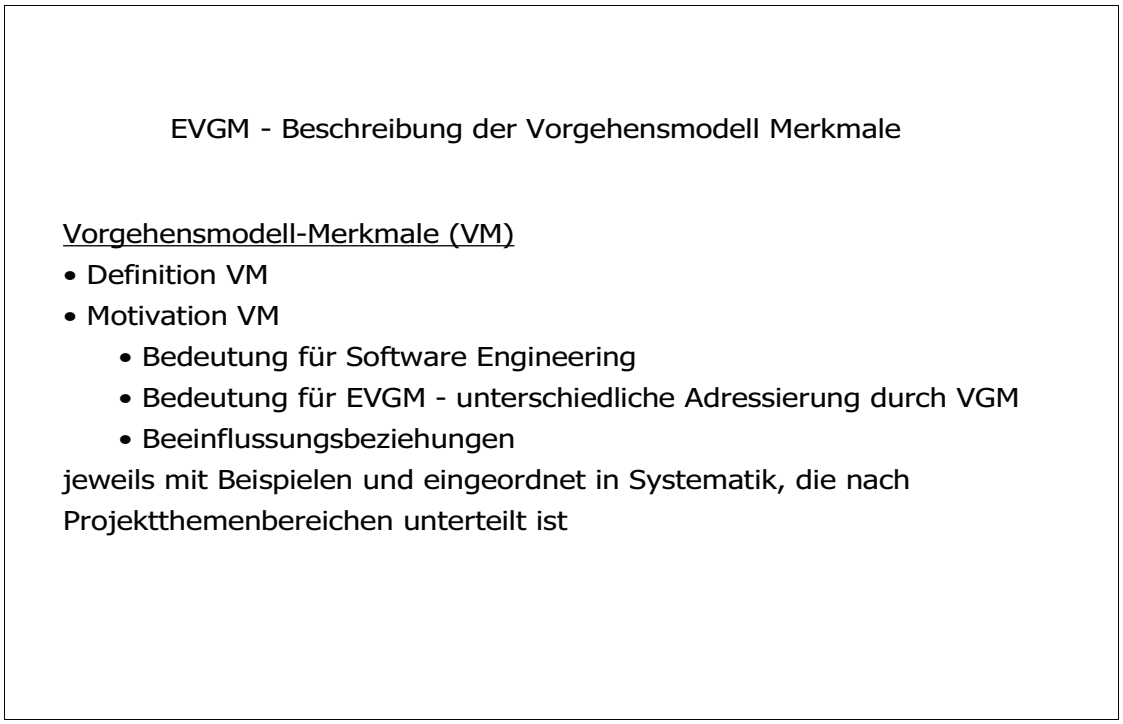

Abbildung 26 Beschreibung der Vorgehensmodellmerkmale

3.2.1 Unterstützung des Requirements Engineerings

Definition des Merkmals *EV-UnterstProjTheBer-RequirementsEngineering*:
Im Requirements Engineering wird erarbeitet, welche fachlichen Funktionen durch die Software unterstützt werden soll. Es wird weiter untersucht, welche nicht funktionalen Anforderungen wie zum Beispiel Antwortzeiten oder der Zugriffsschutz für Daten und Funktionen erfüllt sein müssen.
Beispiel: Eine Unterstützung für das Requirements Engineering bietet der *Vorgehensbaustein Anforderungsfestlegung* im V-Modell XT [VMX06] mit den entsprechenden Rollen, Aktivitäten und Arbeitsprodukten (hier: Produkten).

Motivation des Merkmals *EV-UnterstProjTheBer-RequirementsEngineering:*
Bedeutung für Softwareprojekte. Das Requirements Engineering wird benötigt, um zu erheben, welche fachlichen Erfordernisse durch die im Projekt zu entwickelnde Software genügen soll. Speziell instabile und intransparente fachliche Anforderungen sind mitunter nicht nur durch die fachliche Veränderungsdynamik gegeben, sondern widerspiegeln auch den Erkenntnisprozeß der Stakeholder während des Projektes. Fundiertes und systematisches Requirements Engineering kann die Stakeholder durch die richtigen Fragen unterstützen, eine fundierte und somit stabilere fachliche Basis zu erarbeiten. Oberflächliches und unsystematisches Requirements Engineering verschlechtert die Sta-

bilität und Transparenz fachlicher Anforderungen, weil dann fachliche Sachverhalte nicht hinreichend grundsätzlich durchdrungen sind, um eine stabile umfassende Darstellung finden zu können. Es gibt Projekte, in denen umfangreiche und komplizierte Anforderungen zu analysieren sind. Dies kann durch die Disziplin Requirements Engineering unterstützt werden.
Beispiel: Das Merkmal *EPD-BedDeck-Fähigk-Fachlich* aus Kapitel 2 wird durch die Disziplin Requirements Engineering beeinflußt.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Das Requirements Engineering wird von Vorgehensmodellen unterschiedlich umfassend und detailliert unterstützt. Das V-Modell XT [VMX06] und der Rational Unified Process [RUP06] beschreiben diese Disziplin sehr umfassend, Extreme Programming [Bec00] und Scrum [Sch02] nur informell.
Beispiel: Extreme Programming fordert den *On-site Customer* und *Story Cards*, das V-Modell XT [VMX06] bietet den Vorgehensbaustein *Anforderungsfestlegung* mit Rollen, Arbeitsprodukten und Aktivitäten (siehe Kapitel 3.3.1 beziehungsweise 3.3.10).

Beeinflussungsbeziehungen mit anderen Merkmalen
Die *EV-UnterstProjTheBer-RequirementsEngineering* verbessert die Verfügbarkeit fachlicher Fähigkeiten **(***EPD-BedDeck-Fähigk-Fachlich)* weil sie Hilfestellung bietet, fachliches Wissen methodisch zu erheben und zu dokumentieren.
Sie verbessert die Überschaubarkeit des geforderten Funktionsumfangs (*EPD-Einf-DurchUmfangFunktionen)*, weil sie unterstützt, die fachlichen Funktionen zu strukturieren, voneinander abzugrenzen und bestehende Abhängigkeiten explizit zu machen.
Auch die Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* wird durch fundiertes Requirements Engineering beeinflusst, wie auch die Überschaubarkeit der technischen Anforderungen und Qualitätskriterien (*EPSWE-Einf-Durch-UmfangTechnischeAnforderungenUndQualitätskriterien)* und die Stabilität und Transparenz der technischen Anforderungen und Qualitätskriterien (*EPS-WE-StaTrans-TechnischeAnforderungenUndQualitätskriterien)*. Die Stärke der Beeinflussungen hängt von dem eingesetzten Vorgehensmodell ab, wird hier aber zunächst mit „≈“ vorgeschlagen.

3.2.2 Unterstützung des Software Engineerings

Definition des Merkmals *EV-UnterstProjTheBer-SoftwareEngineering:*
Die Disziplin Software Engineering unterstützt die Erstellung von fachlichen und technischen Projekt-Arbeitsprodukten. Sie besteht aus den Teildisziplinen Analyse, Design, Implementierung und Qualitätssicherung im Sinne des Glossars im Anhang.
Beispiel: Unterstützung für das Software Engineering bietet der Vorgehensbaustein *Qualitätssicherung* im V-Modell XT [VMX06].

Motivation des Merkmals *EV-UnterstProjTheBer-SoftwareEngineering*:
Bedeutung für Softwareprojekte. Die Teildisziplinen des Software Engineering decken den Kern und die eigentliche Wertschöpfung in Softwareentwicklungsprojekten ab. Softwareentwicklungsprojekte unterscheiden sich nach dem bestehenden Anspruch an das Software Engineering. Dieser kann bei einer zu entwickelnden Software mit vielen, komplizierten fachlichen Funktionen und technischen Anforderungen hoch sein.
Beispiel: Das Merkmal *EPSWE-BedDeck-Fähigk-SoftwareEngineering* im Kapitel 2 wird durch den Projektthemenbereich Software Engineering beeinflusst.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen Die Teildisziplinen des Software Engineering werden von Vorgehensmodellen unterschiedlich umfassend und detailliert beschrieben. Das V-Modell XT [VMX06] und der Rational Unified Process [RUP06] unterstützen die Disziplin Software Engineering umfassend, Extreme Programming [Bec00] informell, Scrum [Sch02] gar nicht.
Beispiel: Der Rational Unified Process [RUP06] bietet die Disziplinen *Analysis & Design*, *Implementation* und *Test* mit Rollen, Aktivitäten und Arbeitsprodukten (siehe Kapitel 3.3.4).

Beeinflussungsbeziehungen mit anderen Merkmalen
Die *EV-UnterstProjTheBer-SoftwareEngineering* verbessert die Verfügbarkeit von Software Engineering Fähigkeiten (*EPSWE-BedDeck-Fähigk-SoftwareEngineering),* weil sie die Analyse, Design, Implementierung und Qualitätssicherung beschreibt.
Sie verbessert die Handhabbarkeit der technischen Anforderungen (*EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien)*, weil sie die Projektmitarbeiter methodisch anleitet, diese Anforderungen zu erfüllen.
Auch die Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* und der technische Anforderungen (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien)* wird verbessert, weil unterstützt wird, diese beispielsweise durch einen Prototypen zu erkunden. Die Stärke der Beeinflussungen hängt von dem eingesetzten Vorgehensmodell ab, wird hier aber zunächst mit „≈“ vorgeschlagen.

3.2.3 Unterstützung des Software Managements

Definition des Merkmals *EV-UnterstProjTheBer-SoftwareManagement*:
Die Disziplin Software Management unterstützt die Verwaltung der Arbeitsprodukte der Disziplin Software Engineering. Sie besteht aus den Teildisziplinen Requirements Management, Konfigurationsmanagement und Change Management im Sinne des Glossars im Anhang.
Beispiel: Der Vorgehensbaustein *Konfigurationsmanagement* im V-Modell XT [VMX06] unterstützt das Software Management.

Motivation des Merkmals *EV-UnterstProjTheBer-SoftwareManagement*:
Bedeutung für Softwareprojekte. Die Teildisziplinen des Software Management bieten den unterstützenden Rahmen für das Software Engineering. In großen Softwareentwicklungsprojekten für umfangreiche Softwaresysteme, aber auch bei Projekten mit sehr instabilen und intransparenten Anforderungen besteht ein großer Anspruch an das Software Management.
Beispiel: Das Merkmal *EPSWE-BedDeck-Fähigk-SoftwareManagement* aus Kapitel 2 wird durch den Projektthemenbereich Software Management beeinflusst.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen Das Software Management wird von Vorgehensmodellen unterschiedlich umfassend und detailliert beschrieben. Der Rational Unified Process [RUP06] unterstützt es sehr umfassend und detailliert, Extreme Programming sehr informell und implizit.
Beispiel: Der Rational Unified Process [RUP06] bietet die Disziplinen *Requirements* und *Configuration & Change Management* mit Rollen, Aktivitäten und Arbeitsprodukten (siehe Kapitel 3.3.4).

Beeinflussungsbeziehungen mit anderen Merkmalen
Eine gute *EV-UnterstProjTheBer-SoftwareManagement* verbessert den Zustand der Bedarfsdeckung der Software Management Fähigkeiten (*EPSWE-BedDeck-Fähigk-SoftwareManagement),* weil sie das Requirements Management, Konfigurationsmanagement und Change Management beschreibt.
Speziell über das Requirements Management und das Change Management kann sie auch die Stabilität und Transparenz der fachlichen Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* und der technischen Anforderungen (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien)* verbessern. Die Stärke der Beeinflussungen hängt von dem eingesetzten Vorgehensmodell ab, wird hier aber zunächst mit „≈“ vorgeschlagen.

3.2.4 Unterstützung des Projektmanagements

Definition des Merkmals *EV-UnterstProjTheBer-ProjektManagement*:
Das Projektmanagement unterstützt die Regelung der organisatorischen und kaufmännischen Belange im Softwareentwicklungsprojekt. Es beschreibt die Planung, Kontrolle und Steuerung des Projekts hinsichtlich von Aktivitäten, Terminen, Aufwänden, Ressourcen und Kosten und schließt das Risikomanagement ein.
Das Risikomanagement dient der systematischen Analyse von Projektrisiken, sowie deren Bewertung und Adressierung mit Gegenmaßnahmen. Es unterstützt das Projektmanagement somit bei Entscheidungen.
Beispiel: Das V-Modell XT [VMX06] mit dem Vorgehensbaustein *Projektmanagement* beschreibt Rollen, Aktivitäten und Arbeitsprodukte der Disziplin Projektmanagement.

Motivation des Merkmals *EV-UnterstProjTheBer-ProjektManagement*:
Bedeutung für Softwareprojekte Die Disziplinen des Projektmanagements unterstützen ein Projekt in kaufmännischer und organisatorischer Hinsicht. Sie sind dabei neutral bezüglich des Gegenstandes des Projektes. In großen Projekten mit umfangreichen Teams und hohen Budgets ist der Bedarf an Unterstützung durch das Projektmanagement groß.
Beispiel: Das Merkmal *EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement* aus Kapitel 2 wird durch die *EV-UnterstProjTheBer-Projekt-Management* beeinflusst.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen ihre Anwender unterschiedlich umfassend und detailliert im Projektmanagement. Der Rational Unified Process [RUP06] unterstützt das Projektmanagement sehr umfassend und detailliert, Scrum [Sch02] und XP [Bec00] sehr informell und nur implizit.
Beispiel: Der Rational Unified Process [RUP06] bietet die Disziplin *Project Management* mit Rollen, Aktivitäten und Arbeitsprodukten (siehe Kapitel 3.3.4).

Beeinflussungsbeziehungen mit anderen Merkmalen
Die *EV-UnterstProjTheBer-ProjektManagement* verbessert den Zustand der Bedarfsdeckung von Projektmanagementfähigkeiten **(***EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement)*, weil sie die Planung, Kontrolle und Steuerung von Aktivitäten, Terminen, Aufwänden, Ressourcen und Kosten, sowie das Risikomanagement beschreibt.

Sie verbessert hierdurch den Zustand der Terminsituation (*EPP-BedDeck-Kap-KalendarischeZeit),* der Budgetsituation *(EPP-BedDeck-Kap-Projektbudget),* der Überschaubarkeit der Teamgröße *(EPP-Einf-DurchTeamgröße)* und der *Bedarfsdeckung personeller Kapazitäten (EPP-BedDeck-Kap-Personell).* Die Stärke der Beeinflussungen hängt von dem eingesetzten Vorgehensmodell ab, wird hier aber zunächst mit „≈“ vorgeschlagen.

3.2.5 Unterstützung der Menschenführung

Definition des Merkmals *EV-UnterstProjTheBer-Menschenführung*:
Die Softwareentwicklung ist ein Prozeß, der existentiell von den Fähigkeiten abhängt, mit menschlichen Eigenschaften umzugehen. *EV-UnterstProjTheBer-Menschenführung* charakterisiert, ob und auf welche Weise ein Vorgehensmodell den Umgang mit dieser Tatsache unterstützt.
Beispiel: Extreme Programming [Bec00] beschreibt das Prinzip *Humanity.*

Motivation des Merkmals *EV-UnterstProjTheBer-Menschenführung*:
Bedeutung für Softwareprojekte. [DeM99] und [Kel94] fordern das Beachten von Soft Factors. Da Softwareentwicklung ein komplexer, hochkreativer Prozeß ist, sind menschliche Stärken wie Kreativität, Intuition oder die Fähigkeit, Schlußfolgerungen zu ziehen, für sie unerläßlich. Die Beachtung von Menschenführung ist wichtig, weil Menschen nicht nur ihre Fähigkeiten, sondern auch ihre menschlichen Schwächen und Fehler wie abweichende egoistische Individualziele, Konkurrenzdenken, Kultur- und Mentalitätsunterschiede, Bereichsanimositäten, Kommunikationsprobleme in ein Projekt einbringen. Menschen können erfahrungsgemäß ihre Fähigkeiten produktiver für das Projekt einsetzen können, wenn sie sich sozial und emotional wohl fühlen [Sch02], [Bec01], [Hig02], [Sta03]. Projekte tendieren dazu, eher an diesen Schwierigkeiten als an technischen Problemen zu scheitern [DeM99]. Auch [Sve05] unterstreicht die Bedeutsamkeit von Menschenführung in der Softwareentwicklung.
In einem Software Engineering Projekt ist es wichtig, welche Unterstützung ein Vorgehensmodell bietet, ein produktives soziales Klima zu erreichen und zu erhalten.
Beispiel: Die Merkmale *EPM-BedDeck-MotivationImTeam*, *EPM-BedDeck-KommunikationImTeam*, *EPM-BedDeck-Führungsfähigkeiten* aus Kapitel 2 werden durch *EV-UnterstProjTheBer-Menschenführung* beeinflusst.
Speziell in einem Off-Shoring Projekt bietet es sich an, die Teambildung und Teampflege, sowie die Überbrückung kultureller Gegensätze betont durch Maßnahmen wie Teambildungs-Workshops, institutionalisierte Kommunikation und Einweisungen in die kulturellen Eigenheiten der Projektpartner, beziehungsweise durch hierfür verantwortliche Rollen zu unterstützen (siehe auch Unterstützungsfrage bezüglich der räumlichen und geographischen Verteilung in Kapitel 2).
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Die eher menschengetriebenen Ansätze (siehe 5.4„Überdeckung des Vorgehensmodellspektrums“ wie die Agilen Methoden widmen der Menschenführung mehr Aufmerksamkeit, während die eher prozeßgetriebenen Ansätze diesen Aspekt weitgehend aussparen. Extreme Programming [Bec00] baut stark auf Menschenführung auf und beschreibt hierfür Prinzipien und Methoden. Das V-Modell XT [VMX06] verzichtet weitestgehend darauf, Menschenführung explizit durch Modellelemente zu adressieren.
Beispiel: Extreme Programming beschreibt die *Values Communication, Feedback, Courage* und *Respect,* die *Principles Rapid Feedback, Open Honest Communication, Work With People's Instinct Not Against Them, Accept Responsibility, Humanity* und

Accepted Responsibility, die *Techniques On-Site Customer, Planning Game, Metaphor, Collective Ownership* und *40 Hour Week* und die *Practices Sit together, Whole Team, Informative Workspace, Energized Work, Stories, Weekly Cycle, Quarterly Cycle, Real Customer Involvement, Team Continuity, Shrinking Teams, Daily Deployment, Negotiated Scope Contract* [Bec04] bezogen auf die Menschenführung (siehe auch Kapitel 3.3.1).

Beeinflussungsbeziehungen mit anderen Merkmalen
Die *EV-UnterstProjTheBer-Menschenführung* verbessert den Zustand der *EPM-BedDeck-MotivationImTeam* weil sie ein gesteigertes Bewußtsein für die Wichtigkeit unterstützt, motivationsfördernd mit den Projektmitarbeitern umzugehen.
Sie verbessert den Zustand der *EPM-BedDeck-KommunikationImTeam* weil sie Informationsaustausch im Projekt anregt und unterstützt.
Sie verbessert die *EPM-BedDeck-Führungsfähigkeiten* weil sie geeignetes Führungsverhalten von Managern anregt und beschreibt (s.o.). Die Stärke der Beeinflussungen hängt von dem eingesetzten Vorgehensmodell ab, wird hier aber zunächst mit „≈“ vorgeschlagen.

3.2.5.1 Flexibilität durch Iterativität und Inkrementalität

Definition des Merkmals *EV-Flexib-DurchIterativitätUndInkrementalität:*
EV-Flexib-DurchIterativitätUndInkrementalität stellt dar, wie stark ein Vorgehensmodell berücksichtigt, daß man in einem Projekt oft nicht alle wichtigen Informationen zur Verfügung hat, wodurch Planungen häufig mit starken Unsicherheiten behaftet sind und nur vorläufigen Charakter besitzen.
Sie charakterisiert, wie ein Vorgehensmodell mit der Tatsache umgeht, daß während der Projektabwicklung Veränderungen von Anforderungen und Rahmenbedingungen auftreten.
Flexibilität beschreibt, wie geeignet ein Vorgehensmodell anhand seiner Rollen, Aktivitäten, Arbeitsprodukte, Prinzipien und Methoden ist, den Umgang mit den durch mangelnde Stabilität und Transparenz bedingten Unsicherheiten in Projekten zu unterstützen.
Beispiel: Scrum [Sch02] schreibt eine Entwicklung in 30-Tages-Iterationen vor, die *Sprint* genannt werden. In diesem werden hierfür fixierte Anforderungen umgesetzt.

Motivation des Merkmals *EV-Flexib-DurchIterativitätUndInkrementalität*:
Bedeutung für Softwareprojekte. [Ket05] fordert, das Vorgehensmodelle unterstützen müssen, mit Intransparenz und Instabilität und umzugehen.
In Projekten sind nie alle wichtigen Sachverhalte rechtzeitig bekannt. Anforderungen werden Anwendern oft erst spät bewußt, technische Restriktionen von Werkzeugen oder Plattformen stellen sich mitunter erst unter realistischer Belastung heraus, wichtige Rahmenrichtlinien der Organisation sind dem Projektteam manchmal bis zur Übergabe eine Softwareproduktes unbekannt. Ein Beispiel hierfür sind technische Beschränkungen einer Middleware Plattform, die ihr Anbieter verschweigt. Dies erschwerte die Planung, Durchführung und Kontrolle von Projekten erheblich.
Außerdem verändern sich mitunter Anforderungen, technische Rahmenbedingungen, Termine, Teambesetzungen oder Budgetbeschränkungen während der Durchführung eines Softwareprojektes und produzieren hierbei noch Seiteneffekte, welche die Planung, Abwicklung und Kontrolle des Projektes erheblich verkomplizieren. Dies kann mitunter

Überarbeitungen oder Neuentwicklungen von Teilen der bereits geleisteten Arbeit erfordern, weil sie auf den alten Ständen der geänderten Anforderungen aufbauen. Ein Beispiel hierfür sind fortwährende Änderungen von Anforderungen und deren Auswirkungen auf von ihnen abhängige, bereits implementierte Anforderungen. Eine Voraussetzung für Flexibilität ist die Schlankheit [Con04], Kompaktheit und und Redundanzfreiheit eines Entwicklungsansatzes.
Softwareprojekte unterscheiden sich stark hinsichtlich der Stabilität und Transparenz der fachlichen und technischen Anforderungen.
Beispiel: Die Merkmale *EPD-StaTrans-FachlicheAnforderungen* und *EPSWE-Einf-EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien* aus Kapitel 2 fordern eine hohe *EV-Flexib-DurchIterativitätUndInkrementalität*.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Mit iterativem Vorgehen (siehe auch Glossar im Anhang) werden bestehende Projekt-Arbeitsprodukte immer wieder überarbeitet und können so an neue oder veränderte Erkenntnisse angepaßt werden. Dadurch ist es möglich, auf Instabilitäten oder Intransparenzen zu reagieren.
Mit inkrementellem Vorgehen (siehe auch Glossar im Anhang) wird immer nur ein fixierter Teil der Projekt-Arbeitsprodukte weiterentwickelt. Dadurch müssen bei neuen oder geänderten Erkenntnissen ein geringerer Arbeitsproduktumfang überarbeitet werden. Teilergebnisse können frühzeitig ausgeliefert und Feedback von den Stakeholdern eingeholt werden. Dies dient der Verminderung des Akzeptanz-, aber auch des Integrationsrisikos. Diese Sicht wird auch in [Oes+99] bestätigt. Vorgehensmodelle sind unterschiedlich flexibel ausgelegt. Extreme Programming [Bec00] und Srcum [Sch02] sind flexibel bis evolutionär (siehe Glossar im Anhang) ausgerichtet, der Rational Unified Process [RUP06] durch größere Iterationen gemäßigter, das V-Modell XT [VMX06] nicht primär dafür ausgelegt.
Beispiel: Extreme Programming minimiert den Umfang der erstellten Arbeitsproduktypen auf Testfälle und Code, was Redundanzen reduziert und daher zur besseren Änderbarkeit beiträgt. Es fordert kurze Iterationen, um nicht zu viele Anforderungen auf einmal umzusetzen. Dadurch wird das Risiko einer zwischenzeitlichen Änderung vermindert. Das V-Modell XT [VMX06] enthält ein viel größeres Arbeitsproduktmodell, durch dessen Redundanzen Änderungen sehr viel aufwendiger und fehleranfälliger werden (siehe Kapitel 3.3.1 und 3.3.10).

Beeinflussungsbeziehungen mit anderen Merkmalen

Eine gute *EV-Flexib-DurchIterativitätUndInkrementalität* kompensiert mangelnde Stabilität und Transparenz fachlicher Anforderungen (*EPD-StaTrans-FachlicheAnforderungen)* und technischer Anforderungen (*EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien)*, weil durch sie aufgrund kleiner Entwicklungspakete weniger fertige Arbeitsprodukte an Veränderungen angepaßt werden müssen. Sie erfordert aber ein über alle Projektthemenbereiche möglichst redundanzfreies Arbeitsproduktmodell, um bei Veränderungen der Gefahr von Inkonsistenzen aus dem Weg zu gehen, somit darf die Unterstützung durch das Arbeitsproduktmodell in keinem Projektthemenbereich hoch sein. Die Stärke der Beeinflussungen hängt von dem eingesetzten Vorgehensmodell ab, wird hier aber zunächst mit „≈“ vorgeschlagen.

3.2.6 Zuordnung Vorgehensmodellmerkmale zu Projektmerkmalen

Im folgenden werden die Beeinflussungsbeziehungen zwischen den Vorgehensmodell-

merkmalen aus dem letzten Abschnitt und den Projektmerkmalen aus Kapitel 2 anhand der Tabelle 4 im Überblick dargestellt. Die Vorgehensmodellmerkmale stehen dabei immer *kursiv* unter den Merkmalen aus Kapitel 2, die sie adressieren. Die Fragen zu Einflußfaktoren aus dem Projektumfeld werden nicht von Vorgehensmodellmerkmalen beeinflusst und werden nur zu Vereinheitlichung der Darstellung mit aufgeführt.

Risikobereich Projektthemen bereiche	***Entwicklungs-Produkt Merkmale***	***Entwicklungs-Team Merkmale***	***Entwicklungs-Umfeld Fragen zu Einflußfaktoren aus dem Projektumfeld***
Domänenwissen *Requirements Engineering*	• EPD-Einf-DurchUmfang-Funktionen *EV-UnterstProjTheBer-RequirementsEngineering* • EPD-StaTrans-FachlicheAnforderungen *EV-UnterstProjTheBer-RequirementsEngineering* *EV-UnterstProjTheBer-SoftwareEngineering* *EV-UnterstProjTheBer-SoftwareManagement* *EV-Flexib-DurchIterativitätUndInkrementalität*	• EPD-BedDeck-Fähigk-Fachlich *EV-UnterstProjTheBer-RequirementsEngineering*	• EPD-StaTrans-Projektziele
Softwareentwicklung *Software Engineering* *Software Management*	• EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien *EV-UnterstProjTheBer-RequirementsEngineering* *EV-UnterstProjTheBer-SoftwareEngineering* • EPSWE-StaTrans-TechnischeAnforderungenUnd-Qualitätskriterien *EV-UnterstProjTheBer-RequirementsEngineering* *EV-UnterstProjTheBer-SoftwareEngineering* *EV-UnterstProjTheBer-SoftwareManagement*	• EPSWE-BedDeck-Fähigk-SoftwareEngineering *EV-UnterstProjTheBer-RequirementsEngineering* *EV-UnterstProjTheBer-SoftwareEngineering* • EPSWE-BedDeck-Fähigk-SoftwareManagement *EV-UnterstProjTheBer-SoftwareManagement*	• EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUnd-Plattformen
Projektmanagement *Projektmanagement*	• EPP-BedDeck-Kap-Projektbudget *EV-UnterstProjTheBer-ProjektManagement* • EPP-BedDeck-Kap-KalendarischeZeit *EV-UnterstProjTheBer-ProjektManagement*	• EPP-BedDeck-Fähigk-OrganisatorischesUnd-KaufmännischesProjekt-management *EV-UnterstProjTheBer-ProjektManagement* • EPP-Einf-DurchTeamgröße *EV-UnterstProjTheBer-ProjektManagement* • EPP-BedDeck-Kap-Personell *EV-UnterstProjTheBer-ProjektManagement*	• EPP-Einf-Durch-WirtschaftlichJuristischeVerteilung • EPP-Einf-Durch-RäumlicheGeografischeVerteilung • EPP-Einf-Durch-Entwicklungstiefe • EPP-Einf-Durch-ArtDesVertraglichen Verhältnisses • EPP-Einf-Durch-UnternehmenspolitischeRahmenbedingungen • EPP-Einf-Durch-Organisatorische-Rahmenbedingungen • EPP-Einf-Durch-

Risikobereich Projektthemen bereiche	***Entwicklungs-Produkt Merkmale***	***Entwicklungs-Team Merkmale***	***Entwicklungs-Umfeld Fragen zu Einflußfaktoren aus dem Projektumfeld***
			Rahmenbedingungen DesZielmarktes • EPP-Einf-Durch-RisikenDurchAbhängigkeiten
Menschenführung *Menschenführung*		• EPM-BedDeck-Motivation-ImTeam *EV-UnterstProjTheBer-Menschenführung* • EPM-BedDeck-KommunikationImTeam *EV-UnterstProjTheBer-Menschenführung* • EPM-BedDeck-Führungsfähigkeiten *EV-UnterstProjTheBer-Menschenführung*	• EPM-Einf-Durch-KompatibilitätZuWertenUndUnternehmenskultur

Tabelle 4 Überblick über Beeinflussung Projektmerkmale durch Vorgehensmodell-Merkmale

3.3 Vorgehensmodellspezifische Bewertungen der Merkmale

Im folgenden werden die im letzten Abschnitt definierten Vorgehensmodellmerkmale jeweils spezifisch für die in der vorliegenden Arbeit berücksichtigten Vorgehensmodelle mit Bewertungen („--“, „-“, „0“, „+“, „++“) versehen. Jedes Vorgehensmodell bekommt für jedes Vorgehensmodellmerkmal einen Wert, der von „--“ für „stark verschlechternde Beeinflussung“ und „-“ für „verschlechternde Beeinflussung“ über „0“ für neutral und „+“ für „verbessernde Beeinflussung“ bis „++“ für „stark verbessernde Beeinflussung“ geht. Somit können Sie in der Modellierung im Rahmen der Eignungseinstufung verwendet werden.
Dieser Zusammenhang wird anhand der folgenden Abbildung 27 verdeutlicht.

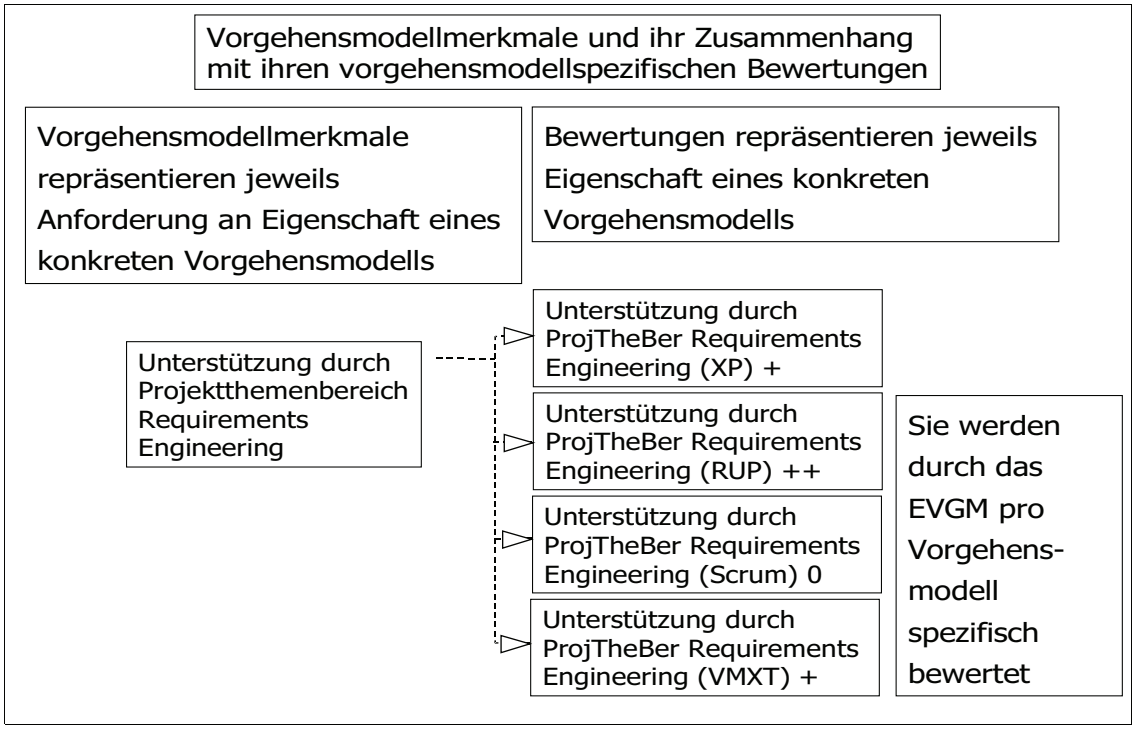

Abbildung 27 Vorgehensmodellmerkmale und spezifische Bewertungen

Nachfolgend wird dargestellt, warum die in der vorliegenden Arbeit berücksichtigten Vorgehensmodelle ausgewählt wurden.

Vorgehensmodellauswahl. Für die Auswahl der in der vorliegenden Arbeit berücksichtigten Vorgehensmodelle wurde bewertet, ob über diese praktische Erfahrungsberichte vorliegen, ob sie aktuell gepflegt werden, ob sie ausdrücklich die Arbeit von Teams adressieren und eine Darstellung aufweisen, welche die Zusammenhänge zwischen Bestandteilen erkennen läßt, also mehr als ein Methodenbaukasten ist. Die exakte Vorgehensweise bei der Auswahl der Vorgehensmodelle kann im Kapitel 5.3„Die Auswahl der Vorgehensmodelle“ nachgelesen werden. Das Ergebnis dieser Auswahl ist

- **Extreme Programming (XP)**
- **Rational Unified Process (RUP)**
- **Scrum**
- **V-Modell XT**

Diese Vorgehensmodellauswahl deckt große komplizierte Projekte (RUP, VMXT) und kleine, dynamische Projekte (Scrum, XP) ab, sie haben ihre Schwerpunktsetzung im Engineering (XP, RUP) beziehungsweise im Management (Scrum, V-Modell XT), wie nachfolgend verdeutlicht wird. Für die exakte Beschreibung dieser Systematik, die zeigt wie die Vorgehensmodellauswahl das Spektrum von Vorgehensmodellen überdeckt, wird auf Kapitel 5.4„Überdeckung des Vorgehensmodellspektrums“ verwiesen. Im Überblick wird sie in der folgenden Abbildung 28 dargestellt.

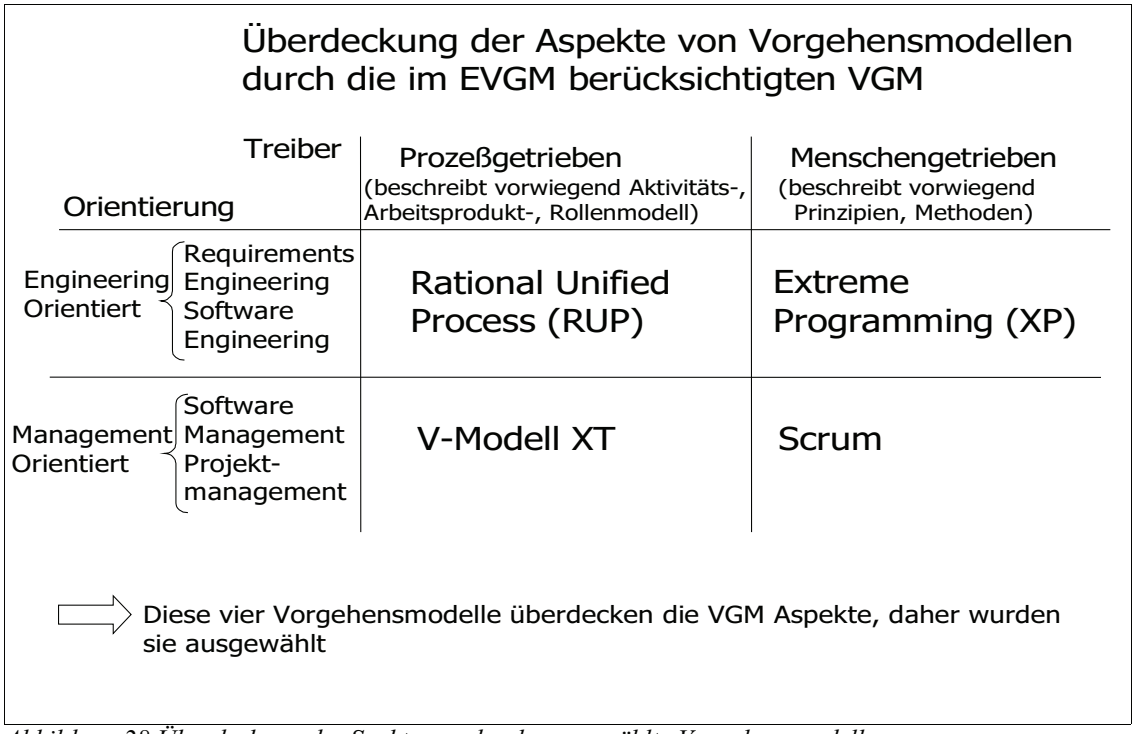

Abbildung 28 Überdeckung des Spektrums durch ausgewählte Vorgehensmodelle

Nachfolgend werden die im EVGM berücksichtigten Vorgehensmodelle entsprechend der in Kapitel 3.1.1 hergeleiteten Vorgehensmodellmerkmale charakterisiert. Hierbei werden ihre Herkunft und Quellen dargestellt und ein Überblick über ihre Modellelemente gegeben. Ihre Darstellung und ihr Detaillierungsgrad wird beurteilt. Ein Überblick über ihren Ablauf wird jeweils kurz dargestellt. Ihre Gesamtintention und Eignung, sowie ihre Nutzungsvoraussetzungen und -hindernisse werden diskutiert. Hierbei werden jeweils herausragende Aspekte aus der Literatur aufgegriffen und mit Hinblick auf die Fragen zu Einflußfaktoren aus dem Projektumfeld aus Kapitel 2 belegt und diskutiert. Damit wird jeweils ein Gesamteindruck jedes Vorgehensmodelles vermittelt. Dies erleichtert die Verständlichkeit ihrer darauffolgend dargestellten vorgehensmodellspezifischen Bewertungen der Vorgehensmodellmerkmale aus Kapitel 3.1.1. Die vorgehensmodellspezifischen Bewertungen der Vorgehensmodellmerkmale dienen zur Modellierung der vorgehensmodellspezifischen Lösungsmodelle, wie im Kapitel 4 beschrieben wird.
Die *Schlüsselwörter* aus den jeweiligen Prozeßhandbüchern sind *kursiv* geschrieben.

3.3.1 Charakterisierung von Extreme Programming (XP)

Extreme Programming ist ein Vorgehensmodell, das den Agilen Methoden [AGI06] zugerechnet wird. Es wurde von Kent Beck aus seiner Projektpraxis heraus entwickelt. Extreme Programming wird in vielen Publikationen beschrieben beziehungsweise kommentiert, beispielsweise in [Bec00], [Bec01], [Bec02], [Bec04], [Hig02], [Lip02], [Wol05], [Wak02], [Jef+01], [Boe04], [Lar04].

3.3.1.1 Modellelemente in Extreme Programming

Extreme Programming enthält *Values, Principles, Techniques, Practices* und *Roles.* Die *Principles* leiten sich aus den *Values* ab. Diese werden von den *Techniques* und *Practices* unterstützt.

Values. *Simplicity, Communication, Feedback, Courage* und *Respect.*
Principles. *Rapid Feedback, Assume Simplicity, Incremental Change, Embrace Change, Quality Work* (im Sinne von Qualitätsbewußtsein)*, Teach Learning, Small Initial Investment, Play To Win, Concrete Experiments, Open Honest Communication, Work With People's Instinct Not Against Them, Accept Responsibility, Local Adaptations* (projektspezifische Eigenschaften berücksichtigen)*, Travel Light* (reduzieren auf das nötigste) und *Honest Measurement, Humanity, Economics (*im Sinne von Wirtschaftlichkeitsbewußtsein)*, Mutual Benefit* (im Sinne von Teamdienlichkeit)*, Self-Similarity* (in Mustern denken)*, Improvement* (Verbesserungsbewußtsein)*, Diversity* (Unterschiedliche Teammitglieder akzeptieren)*, Reflection, Flow* (kontinuierliche Lieferung)*, Opportunity* (Probleme als Chancen betrachten)*, Redundancy (*unterschiedliche Lösungsoptionen versuchen)*, Failure* (aus Fehlern lernen)*, Quality, Baby Steps, Accepted Responsibility* [Bec04].
Techniques. *On-Site Customer, Planning Game, Metaphor, Short Releases, Testing, Simple Design, Refactoring, Pair Programming, Collective Ownership, Continuous Integration, Coding Standards* und *40 Hour Week.*
Practices. *Sit together, Whole Team, Informative Workspace* (Möglichkeit, sich über Projektstand zu informieren)*, Energized Work* (Zeitmanagement)*, Stories* (Detaillierung von Story Cards)*, Weekly Cycle* (Wochenplanung von Aufwänden)*, Quarterly Cycle* (Quartalsplanung von Aufwänden)*, Slack* (realistische Lieferzusagen machen)*, Ten-Minute-Build* (täglich komplett übersetzen und automatische Tests durchlaufen lassen)*, Test-first Programming* (Testfälle vor dem Code entwickeln)*, Incremental Design* (Das Design entwickelt sich mit dem Fortschreiten des Projekts)*, Real Customer Involvement* (tatsächlichen Endbenutzer einbeziehen)*, Incremental Deployment* (System in anwachsendem Umfang ausliefern)*, Team Continuity* (Fluktuation vermeiden)*, Shrinking Teams* (Team immer so klein wie möglich halten)*, Root-Cause-Analysis* (Fehlerursachen ermitteln)*, Shared Code* (jeder ist für den ganzen Code verantwortlich)*, Code and Tests* (Arbeitsproduktmodell minimieren)*, Single Code Base* (ca. halbtägliche Integration)*, Daily Deployment* (täglich lauffähige Software liefern)*, Negotiated Scope Contract* (kurze Vertragslaufzeiten heraus handeln)*, Pay-Per-Use* (Benutzungsabhängige Bezahlung der Software) [Bec04]
Roles. Die Organisation von großen Entwicklungsteams wird in Extreme Programming bedingt unterstützt. Es bietet einige Rollenbeschreibungen mit Fähigkeitsanforderungen. Extreme Programming beschreibt keine durchgängigen expliziten Verantwortlichkeiten von Rollen für Aktivitäten und Arbeitsprodukte. Es definiert die *Roles: Tester, Interaction Designer, Architect, Project Manager, Product Manager, Executive, Technical Writer, User*, *Programmer* und *Human Resources.* Durch seine Schwerpunktsetzung auf verbaler Kommunikation skaliert es nicht für große Entwicklungsteams: „Due to the communication overhead XP as is does not scale well“ [Mue01].
Aktivitäten. In Extreme Programming werden keine konkreten und zusammenhängenden Aktivitäten im Sinne der Definition im Glossar im Anhang beschrieben. *Practices* und *Techniques* leiten den Benutzer an, sind aber nicht in Form eines Prozesses in einen temporalen oder kausalen Zusammenhang gebracht, womit die Entscheidung, welche Practice und welche Technique unter welchen Bedingungen anzuwenden ist, weitestgehend dem Projektmitarbeiter überlassen bleibt. Dies entspricht dem Agilen Paradigma, erfordert aber entsprechende Software Engineering und Software Management Fähigkeiten bei seinen Benutzern.
Arbeitsprodukte. Umfangreiche Projektinformationen und deren Zusammenhänge

werden in Extreme Programming aufgrund der Tatsache, daß Testfälle und Code die einzigen persistenten Arbeitsprodukte sind, nur bedingt unterstützt. Dementsprechend gestaltet sich „Refactoring großer Systeme sehr schwierig und aufwendig“ [Lip04]. Hinsichtlich großer Funktionsumfänge skaliert Extreme Programming unter der Verletzung seiner Prinzipien: „Agile skaliert nur mit nicht Agilität, z.B. Architektur“ [Rea03].

3.3.1.2 Darstellung, Detaillierungsgrad in XP

In Extreme Programming werden nicht im herkömmlichen Sinne [Chr92] Aktivitäten und Arbeitsprodukte definiert und in zeitlichen und sachlichen Zusammenhang gebracht. Die Beschreibung von Extreme Programming hat einen sehr generischen, deklarativen und eher regelbasierten als imperativen Charakter und gleicht mehr einer vernetzten Gedanken- und Ideensammlung. Sie verteilt sich auf mehrere Bücher mit teilweise inkonsistenten Aussagen (zum Beispiel in Bezug auf die Eignung von Extreme Programming für verteilte Entwicklung [Bec00] schließt es kategorisch aus, [Bec04] stellt es als mögliche Option dar). Es ist erforderlich, so viel über einen schlanken Ansatz zu schreiben, weil er am Anfang [Bec00] nie wirklich konkret wird. Alleine für die vorliegende Arbeit wurden mehr als 1500 Buchseiten Methodenbeschreibung gesichtet, Sekundärliteratur und Erfahrungsberichte nicht mit eingerechnet. Auch dies stellt nur eine Teilmenge der aktuell verfügbaren Publikationen dar.

3.3.1.3 Überblick über XP

In Extreme Programming werden Releases in Iterationen von wenigen Wochen unterteilt. Vor einer Iteration werden im *Planning Game* von *User* und *Programmer* Anforderungen auf *Story Cards* geschrieben, geschätzt und priorisiert. Die *Metaphor* soll ein für alle verständliches einfaches Sinnbild der Anwendung bieten [Boe04]. *Coding Standards* regeln die Gestaltung des Codes. Dann wird mit Unterstützung des *On-Site Customers* die Entwicklung begonnen. Hierbei werden zuerst die Testklassen, dann der Code entwickelt und darauf geachtet, das Design so einfach wie möglich zu halten. Das *Pair Programming* ersetzt dabei den Code Review und sorgt dafür, das jeder soviel Code wie möglich kennt und die Verantwortung für ihn trägt. Neue Entwicklungsstände werden kontinuierlich getestet und integriert. Frühe Releases sorgen für frühes Feedback und vermindern das Projektrisiko. Mit *Refactoring* [Fow99] wird immer daran gearbeitet, möglichst schlanken und einfachen Code zu haben. Die *40-Hour-Week* darf nur punktuell überschritten werden.

3.3.1.4 Intention, Eignung, Nutzungsvoraussetzungen von XP

Projektgröße. Extreme Programming adressiert kleine Projekte [Boe04], in denen einfache Individual-Informationssysteme neu entwickelt werden. Für große Projekte sind Erweiterungen notwendig [Lip+03], beispielsweise eine Architektur [Cao04].

Kritikalität von Projekt und Produkt. Die Verwendbarkeit von Extreme Programming für sicherheitskritische Problemstellungen wird insgesamt eher skeptisch gesehen[Bec00], [Wäy+04]. [Gre01], [Dro+04] schildern die Abwicklung kritischer Projekte mit entsprechenden Anpassungen von Extreme Programming. Auch in Projekten in Verbindung mit Legacy Systemen werden Anpassungen von Extreme Programming geschildert [Col06]. Die Bewältigung restriktiver technischer Rahmenbedingungen zu unterstützen liegt nicht im Scope von Extreme Programming [Boe04].

Verteilung. Extreme Programming ist ausdrücklich nicht dafür gedacht, örtlich verteilte Projekte [Boe04] zu unterstützen, auch wenn es hierfür schon angewendet wurde

[Lip02]. Speziell die Unterbringung des Entwicklungsteams in einem Raum ist obligatorisch in Extreme Programming [Lar04], auch wenn in diversen Publikationen wie [Kir +01], [Ree04], [Mar04], [Poo04], [Bra05], [Han04] propagiert wird, daß man es auch in verteilten Projekten anwenden kann. In [Hol02], [Bra05], [Kar+00], [Xia+04], [XU05], [Yap05] werden Erfahrungsberichte über die Verwendung von Extreme Programming in geografisch verteilten Projekten präsentiert. Diese haben alle gemeinsam, das Extreme Programming jeweils entsprechend angepasst wurde, um den Erfordernissen eines geografisch verteilten Projektes zu genügen. Kommunikation ohne Arbeitsproduktdokumente und Modelle und ohne persönliches Gespräch birgt erhebliche Risiken für Mißverständnisse. Um diesen zu begegnen werden Modifikationen von Extreme Programming vorgeschlagen, was Kernaussagen von Extreme Programming (*On-site Customer, Face-To-Face-Communication, Pair Programming*) verletzt. Damit wird aber per Definition kein Extreme Programming mehr sondern nur noch Techniken hieraus verwendet, weil Extreme Programming nicht anpassbar ist [Lar04], [Sha04]. Auch [Kus+04] entdecken Widersprüche von verteilter Entwicklung zu agilen Prinzipien.

Flexibilität. Mit Extreme Programming können große Instabilitäten und Intransparenzen bewältigt werden, es ist ein evolutionärer Ansatz.

Wiederverwendbarkeit. Die Erzeugung von wiederverwendbarer Software zu unterstützen liegt nicht im Scope von Extreme Programming. Hiergegen spricht beispielsweise das *Simple Design,* welches kontextübergreifend benutzbare Lösungen ausschliesst.

Benutzungsanforderungen. Anwender von Extreme Programming erhalten keine umfassende, detaillierte und durchgängige Anleitung für die Entwicklungsarbeit. Es kann daher nur von qualifizierten und erfahrenen Entwicklern benutzt werden, wie auch [Boe04] unterstreicht. Sie müssen kommunikations- und entscheidungsstark, eigenverantwortlich und selbstorganisiert sein und brauchen die Fähigkeit, fast ausschließlich im Code denken zu können. Sie müssen weiterhin in der Objekttechnologie bewandert, team- und kritikfähig sein. Es sind insbesondere fachkompetente und entscheidungsstarke *User* gefordert.

Automatisierte Testunterstützung sind ein absolutes Muß, ein Refactoring Browser sehr empfehlenswert.

Extreme Programming benötigt mitunter signifikanten Einführungsaufwand [Lay04]. Seine Einführung bringt potentiell eine Veränderung der Entwicklungskultur mit sich [Boe04].

3.3.2 Merkmale für Extreme Programming

3.3.2.1 Unterstützung des Requirements Engineerings in XP

Das Requirements Engineering wird in Extreme Programing unterstützt durch *on-site customer* [Lar04], S*tory Cards* [Lar04], S*tories, Metaphor, Real Customer Involvement* und *Daily Deployment.* Es ist sehr kommunikationslastig und baut auf die Fähigkeit der Projektbeteiligten, sich viel fachliches Wissen zu merken. Eine Unterstützung, große Anforderungsmengen zu systematisieren, sowie nicht funktionale Anforderungen und innere Qualitätskriterien zu berücksichtigen, wird nicht geboten. Auch die Berücksichtigung der Abhängigkeiten zwischen Anforderungen ist mit *Story Cards* nicht zu leisten [Woi05]. Das Konzept des *On-site Customers* birgt das Risiko, daß dieser eine zu individuelle Sicht auf die Problemstellung vermittelt [Kos04]. [Boe04] unterstreicht die signifikanten Risiken, die durch ungeeignete O*n-site Customer* entstehen, weil es keine

weitere Qualitätssicherung für die Anforderungen gibt. Er fordert für den *on-site Customer* die Eigenschaften „Collaborative, Representative, Authorized, Committed, Knowledgeable“ [Boe04]. [Pat02] betont, daß Requirements Engineering in Extreme Programming weiterer Unterstützung bedarf.

Bewertung. In der vorliegenden Arbeit wird für Extreme Programming für die EV-UnterstProjTheBer-RequirementsEngineering die Bewertung „**+**“ vergeben.

3.3.2.2 Unterstützung des Software Engineerings in XP

Die **Analyse** wird nicht unterstützt.

Das **Design** wird unterstützt mit den Elementen *Simplicity, Assume Simplicity,Simple Design, Incremental Design*, wobei das Design bereits im Code erfolgt. Modelle oder Dokumente werden nicht angefertigt. Die *Roles Interaction Designer, Architect* und *Programmer* unterstützen das Design.

Die **Implementierung** wird unterstützt mit den Elementen *Simplicity, Assume Simplicity, Concrete Experiments, Refactoring* [Lar04], *Pair Programming* [Lar04], *Coding Standards* [Lar04] und *Redundancy*. Sie wird weiter unterstützt durch die *Role Programmer.*

Die **Qualitätssicherung** wird unterstützt mit den Elementen *Testing, Coding Standards, Quality Work, Honest Measurement, Refactoring, Pair Programming, Collective Ownership, Continuous Integration, Ten-Minute-Build, Test-first Programming* [Lar04]*, On-site Customer* [Lar04], *Root-Cause-Analysis, Shared Code.* [Bez05] kommt zu dem Schluß, das Extreme Programming den Anforderungen einer umfassenden, methodischen Qualitätssicherung, speziell bezogen auf innere Qualitätskriterien wie Security nicht gerecht wird. Extreme Programming eignet sich für Anwendungen mit einer grafischen Benutzerschnittstelle, da diese durch die Anwender beurteilt werden kann. Die Qualitätssicherung wird unterstützt durch die *Roles: Tester, User* und *Programmer.*

Bewertung. In der vorliegenden Arbeit wird für Extreme Programming für die EV-UnterstProjTheBer-SoftwareEngineering die Bewertung „**+**“ vergeben.

3.3.2.3 Unterstützung des Software Managements in XP

Das **Requirements Management** wird unterstützt mit den Elementen *Planning Event, Release Planning, Iteration Planning und Iteration Planning Meeting, Real Customer Involvement, Incremental Deployment, Daily Deployment,* sowie durch die *Roles Project Manager* und *Product Manager.*

Das **Change Management** wird unterstützt mit den Elementen *Incremental Change, Embrace Change, Collective Ownership* und *Real Customer Involvement,* sowie durch die *Roles Project Manager, Product Manager* und *Programmer.*

Das **Konfigurationsmanagement** wird unterstützt mit den Elementen *Continuous Integration, Ten-minute build, Single Code Base, Daily Deployment,* sowie durch die *Role Programmer.*

Bewertung. In der vorliegenden Arbeit wird für Extreme Programming für die EV-UnterstProjTheBer-SoftwareManagement die Bewertung „**+**“ vergeben.

3.3.2.4 Unterstützung des Projektmanagements in XP

Das kaufmännische und organisatorische Projektmanagement wird unterstützt mit den Elementen *Planning Game* [Lar04]*, Economics, Negotiated Scope Contract, Pay-Per-Use, Energized Work, Weekly Cycle, Quarterly Cycle, Slack, Shrinking Teams.*

Das Risikomanagement wird unterstützt mit den Elementen *Small Initial Investment,*

Concrete Experiments, Travel Light, Baby Steps, Short Releases, Incremental Deployment, Rapid Feedback, Flow. Es unterstützen die *Roles Project Manager* und *Human Resources.*

Bewertung. In der vorliegenden Arbeit wird für Extreme Programming für die EV-UnterstProjTheBer-ProjektManagement die Bewertung „**+**“ vergeben.

3.3.2.5 Unterstützung der Menschenführung in XP

In Extreme Programming steht Menschenführung stark im Fokus, diese werden in den *Elementen Communication, Feedback, Courage* und *Respect* sowie den *Principles Rapid Feedback, Open Honest Communication, Work with people's instinct, not against them, Accept Responsibility, Humanity, Mutual Benefit,* den *Practices Sit Together, Whole Team, Teach Learning, Diversity, Reflection, Opportunity, Failure, Quality, Baby Steps, On-site Customer, Collective Ownership, 40 Hour Week, Sit Together, Informative Workspace, Real Customer Involvement, Team Continuity, Shrinking Teams* gefordert. Ein so sehr auf informelle verbale Kommunikation bauender Ansatz wie Extreme Programming stellt hohe Anforderungen an die kommunikativen Fähigkeiten seiner Anwender: „XP Team Members müssen in der Lage sein, ein soziales Klima mit Vertrauen, Respekt und Verantwortlichkeit aufzubauen und zu erhalten. Dafür braucht man die richtigen Leute“ [Rob04]. Extreme Programming fordert und fördert Kommunikationsbereitschaft im Projekt [Hol+06], sowie die Berücksichtigung von Social Factors [Law05]. Die hohe Abhängigkeit von verbaler Kommunikation und Teamorientierung in Extreme Programming wird auch von [Lar04] bestätigt. Die Disziplin Menschenführung wird nicht dezidiert durch eine Rolle unterstützt.

Bewertung. In der vorliegenden Arbeit wird für Extreme Programming für die EV-UnterstProjTheBer-Menschenführung die Bewertung „**++**“ vergeben.

3.3.2.6 Flexibilität durch Iterativität und Inkrementalität in XP

Extreme Programming ist an der Optimierung von Flexibilität ausgerichtet und geht gezielt mit Unsicherheiten im Projekt um. Intransparenzen bei fachlichen Anforderungen wird durch den Ansatz des *On-site Customer* adressiert. Instabilitäten fachlicher Anforderungen werden durch den *On-site Customer* und die *Short Releases* adressiert. Extreme Programming hat insgesamt evolutionären Charakter, der durch Iterationen über kleinen Inkrementen ausgedrückt wird. Das *Principle Embrace Change* steht für die gesamte Geisteshaltung, die Extreme Programming fordert.

Weitere Elemente, welche die Flexibilität unterstützen sind *Rapid Feedback, Incremental Change, Weekly Cycle, Real Customer Involvement, Incremental Deployment, Negotiated Scope Contract.* Insbesondere das minimierte Arbeitsproduktmodell ist geeignet, flexibel auf Veränderungen zu reagieren. Keine Rolle unterstützt dezidiert die Flexibilität.

Bewertung. In der vorliegenden Arbeit wird für Extreme Programming für die EV-Flexib-DurchIterativitätUndInkrementalität die Bewertung „**++**“ vergeben.

3.3.3 Zusammenfassende Bewertung von Extreme Programming

Vorgehensmodellmerkmal in Extreme Programming	Bewertung im EVGM
EV-UnterstProjTheBer-RequirementsEngineering	+
EV-UnterstProjTheBer-SoftwareEngineering	+
EV-UnterstProjTheBer-SoftwareManagement	+
EV-UnterstProjTheBer-ProjektManagement	+
EV-UnterstProjTheBer-Menschenführung	++
EV-Flexib-DurchIterativitätUndInkrementalität	++

Tabelle 5 Bewertung von Extreme Programming

3.3.4 Charakterisierung des Rational Unified Process

Der Rational Unified Process (RUP) wurde von Phillipe Kruchten [Kru99] entwickelt. Er ist eine Konkretisierung des Unified Software Development Process [Lar04], der von Ivar Jacobson, Grady Booch und James Rumbaugh als Konsolidierung und Weiterentwicklung ihrer eigenen objektorientierten Methoden beschrieben wurde [Jac+99]. Der Rational Unified Process (RUP) wird in diversen Publikationen beschrieben [Ber04], [Ess03], [Ess07], [Lar04], [Kro03], [Kru99], [Ver00] [Jac+99]. Er ist elektronisch verfügbar [RUP06].

3.3.4.1 Modellelemente im Rational Unified Process

Der Rational Unified Process enthält die Modellelemente *Phase, Discipline, Role, Task* und *Work Product,* der Bezeichnung für Arbeitsprodukte im Rational Unified Process.

Phases. Sie stellen Schwerpunkte von *Disciplines* dar. Der Rational Unified Process ist auf oberster Ebene strukturiert in die *Phases Inception, Elaboration, Construction* und *Transition. Phases* stehen für Tätigkeitsschwerpunkte. Sie sind sequentiell und überschneidungsfrei. *Phases* enthalten jeweils Iterationen und führen auf Meilensteine.

Disciplines. Sie sind die nächst tiefere Ordnungsstruktur im Rational Unified Process. Sie stehen für Themenfelder in Softwareentwicklungsprojekten. Der Rational Unified Process definiert folgende *Disciplines*:

Das ***Business Modeling*** unterstützt die Geschäftsprozeßmodellierung. Es arbeitet mit *Business Modeling Guidelines*. Es werden die Organisation analysiert, die Geschäftsprozeßziele abgestimmt, ein fachliches Glossar entwickelt, Geschäftsregeln dokumentiert und *Business Use Cases* und die *Business Object Modells* modelliert. Es wird festgelegt, welche Teile der Geschäftsprozesse durch das zu entwickelnde Softwaresystem unterstützt werden sollen. Dabei entstehen die *Work Products Glossar, Business Rules, Business Use Case Model und Business Object Model*. Diese Arbeitsprodukte sind die Grundlage für die *Requirements* [Ess03].

Die *Discipline* ***Requirements*** dient dem kontinuierlichen Erfassen, Dokumentieren, Organisieren und Verfolgen von fachlichen und technischen Anforderungen. Es wird der Leistungsumfang des zu entwickelnden Softwaresystems definiert und begrenzt, die Basis für ein gemeinsames Verständnis für das Projekt und für die technische und kaufmännische Planung geschaffen. Das fachliche Problem wird analysiert, die Bedürfnisse der Stakeholder erfasst, das System definiert und hinsichtlich seines Umfanges begrenzt. Sich ändernde Anforderungen werden berücksichtigt. Hierfür wird ein Anforderungsmanagementplan und eine Vision entwickelt und relevante Akteure und Uses Cases identifiziert. Die Use Cases werden detailliert und priorisiert und ein Oberflächenprototyp

wird erstellt. Dabei entstehen die *Work Products Stakeholder Requests, Vision, Use Cases und Use Case Model, Supplementary Specification und Software Requirements Specification* [Ess03].

Durch die *Discipline* ***Analysis&Design*** wird der Übergang zur modellhaften Darstellungen vollzogen. Hierbei wird zunächst plattformunabhängig und danach unter Einbeziehung der Plattformeigenschaften modelliert. Es wird mit *Design Guidelines*, der *Software Requirements Specification*, dem *Use Case Story Board* und dem *Business Object Model* gearbeitet. Zentrale Funktionalitäten wie Fehlerbehandlung werden entwickelt. Subsysteme, Klassen und die Datenbankstrukturen werden entworfen. Hierbei entstehen die *Work Products Software Architecture Document, Design Model* und *Data Model*.

Die *Discipline* ***Implementation*** unterstützt die Programmierung der Komponenten und deren Integration. Dies schließt Entwicklertest, Bugfixing und Code Reviews ein. Ein lauffähiges System entsteht. Hierbei werden die *Work Products* aus den *Disciplines Analysis&Design*, sowie *Test* verwendet. Es entstehen die Komponenten und Teilsysteme des zu entwickelnden Softwaresystems, sowie der *Integration Build Plan*. Am Ende steht der *Build*, die erste Komplettversion.

Die *Discipline* ***Test*** unterstützt die Qualitätssicherung im Sinne des Glossars im Anhang. Die Funktionsweise und die Erfüllung der Anforderungen der Software wird überprüft. Hier wird mit *Test Guidelines*, *Software Requirements Specification, Use Case Model, Use Case Story Board, Software Architecture Document, Design Model* und *Build* gearbeitet. Das Testmanagement, die Testanalyse, -Design und Durchführung sind hier enthalten. Es entstehen die *Test Evaluation Summary* und die *Test Results*.

Die *Discipline* ***Deployment*** wickelt die Auslieferung des Softwaresystems ab. Auch Benutzertests und -Schulungen sind hier enthalten. Es enthält die definierten Verteilungs- und Freigabeaspekte.

Die *Discipline* ***Configuration&Change Management*** unterstützt die Konsistenzerhaltung und kontrollierte Änderung der Projekt-Arbeitsprodukte.

Durch die *Discipline* ***Project Management*** werden die Softwareentwicklungsdisziplinen planerisch und kaufmännisch unterstützt. Hierzu gehört auch das Risikomanagement. Es arbeitet mit den *Work Products Development Case, Development Infrastructure, Software Requirements Specification, Software Architecture Document, Test Evaluation Summary*. Es werden die wirtschaftlichen Randbedingungen des Projektes festgelegt. Ressourcen werden geplant und die Risikoliste gepflegt. Der *Software Development Plan* und die *Iteration Plans* werden erstellt und aktualisiert. Der Projektstatus wird geprüft und Meilensteinreviews abgewickelt.

Die *Discipline* ***Environment*** passt den Rational Unified Process an das aktuelle Projekt an. Hierbei entsteht der *Development Case*. Hierbei wird bestimmt, welche *Work Products* wann erstellt werden sollen. Dabei entstehen projektspezifische Templates. Das Ineinandergreifen von Entwicklungsprozess und -Tools wird ermöglicht.

Roles. Der Rational Unified Process bietet ein umfassendes Rollenmodell. Die Rollen heißen *Role,* stehen für Verantwortungsbereiche und sind detailliert beschrieben. Die Organisation großer Teams wird unterstützt.

Es gibt im ***Role Set Analysts*** die Rollen *Business-Process Analyst, Business Designer, System Analyst, Requirements Specifier***,**

im ***Role Set Developers*** *Software Architect, Designer, User-Interface Designer, Capsule Designer, Database Designer, Implementer, Integrator***,**

im ***Role Set Managers*** *Project Manager, Change Control Manager, Configuration*

Manager, Test Manager, Deployment Manager, Process Engineer, Management Reviewer,

im ***Role Set Production and Support*** *Technical Writer, System Administrator, Tool Specialist, Course Developer, Graphic Artist,*

im ***Role Set Testers*** *Tester, Test Analyst, Test Designer,*

und die ***Additional Roles*** *Reviewer, Review Coordinator, Technical Reviewer und Stakeholder.*

Tasks. Sie stehen für Tätigkeiten. Das Aktivitätsmodell des Rational Unified Process ist umfangreich. Die Abwicklung komplizierter Folgen von Entwicklungsschritten wird unterstützt.

Aufgrund ihrer Anzahl ist es nicht zielführend, sie hier einzeln aufzulisten. Da Aktivitäten in *Disciplines* organisiert sind, werden diese hier aufgeführt (siehe oben).

Das Arbeitsproduktmodell des Rational Unified Process unterstützt die strukturierte Erfassung vielfältiger Informationen.

Work Products. Sie stehen für Arbeitsprodukte und und werden spezialisiert in *Artifacts*, die für verwaltete Arbeitsergebnisse stehen, *Deliverables*, die Liefergegenstände repräsentieren, und *Outcomes*, die nicht greifbare, formlose Arbeitsergebnisse symbolisieren [Ess07]. Diese Unterscheidung ist für die Eignungseinstufung von Vorgehensmodellen jedoch nicht relevant. Aufgrund ihrer Anzahl ist es nicht zielführend, die *Work Products* einzeln aufzulisten.

Principles. Folgende Prinzipien definiert der Rational Unified Process: *Attack risks early and continuously – or they will attack you* bei [Kro03] zitiert nach [Gil88].

Practices. Der Rational Unified Process beschreibt die Best Practices: *Develop Iteratively, Manage Requirements, Use Component Architectures, Model Visually* (mit UML)*, Continuously Verify Quality, Manage Change*.

3.3.4.2 Darstellung, Detaillierungsgrad des RUP

Der Rational Unified Process ist ist umfassend und detailliert beschrieben. Er ist im HTML-Format erstellt, was Navigation und Anpassung erleichtert.

3.3.4.3 Überblick über den RUP

In der ***Inception*** wird eine gemeinsame Vision skizziert und etabliert. Sie dauert wenige Tage und hat potentiell nur eine Iteration. Es werden Requirements Workshops abgehalten und die wichtigsten zehn Anforderungen priorisiert. Der Business Case für das Projekt wird erarbeitet [Lar04]. Sie führt auf den Meilenstein *Lifecycle Objective* [Ess03].

In der ***Elaboration*** wird die Kernarchitektur entwickelt und versucht, die größten Risiken abzusichern. Sie wird in mehreren Iterationen in kurzen Time Boxes abgewickelt. Weitgehend fundierte Planungen und Schätzungen werden erstellt. Es finden *Requirements Workshops* statt und das Design wird modelliert. Das Kernsystem und die meisten Anforderungen sollen nach ihr weitestgehend stabil sein [Lar04]. Sie führt auf den Meilenstein *Lifecycle Architecture* [Ess03].

Die ***Construction*** unterstützt die Entwicklung und den Test der restlichen Funktionalität. Sie wird in kurzen Iterationen abgewickelt und baut auf die Arbeitsprodukte der *Elaboration* auf. Es werden Anforderungsänderungen im kleineren Umfang berücksichtigt. Die entstehende Alphaversion des zu entwickelnden Softwaresystems wird getestet. Durchsatzoptimierung und Dokumentation werden durchgeführt [Lar04]. Sie führt auf den Meilenstein *Initial Operational Capability* [Ess03].

Die ***Transition*** dient dem Roll-Out des Softwaresystems. Testversionen werden gelie-

fert, Reviews und Feedback entgegengenommen. Parallelbetrieb mit Altsystemen wird im Bedarfsfall durchgeführt. Benutzerschulungen und Datenmigrationen werden abgewickelt [Lar04]. Sie führt auf den Meilenstein *Product Release* [Ess03].

3.3.4.4 Intention, Eignung, Nutzungsvoraussetzungen des RUP

Der Rational Unified Process ist iterativ, inkrementell, risikogetrieben, architekturzentriert und anwendungsfallgetrieben [Ess03]. Er ist geeignet, die Entwicklung von betriebswirtschaftlichen Systemen zu unterstützen.

Projektgröße. Der Rational Unified Process eignet sich für kleine und große Teams [Kro03]. Speziell seine Anwendbarkeit für kleine Projekte wird von [Hir02], [Kim05] bestätigt.

Kritikalität von Projekt und Produkt. Sicherheitskritische Software kann mit dem Rational Unified Process entwickelt werden [Lar04].

Verteilung. Der Rational Unified Process ist für verteilte Projekte geeignet [Ber04]. Gefordert durch sein umfassendes Arbeitsproduktmodell liegt viel Information elektronisch vor und kann geografisch verteilt werden.

Flexibilität. Der Rational Unified Process unterstützt den Umgang mit gemässigten Instabilitäten und Intransparenzen, ist aber kein evolutionärer Ansatz.

Wiederverwendbarkeit. Sie steht nicht im Fokus des Rational Unified Process. Wiederverwendung wird zwar punktuell angeregt, aber nicht umfassend und systematisch unterstützt.

Benutzungsanforderungen. Der Rational Unified Process bietet umfassende Beschreibungen und ist daher auch von Benutzern mit weniger umfassenden Kenntnissen anwendbar. Das Tailoring des Rational Unified Process ist schwierig [Boe04] und sollte professionell begleitet werden [Han+05]. Es wird von diversen Werkzeugen unterstützt [Lar04], [Ess03], was aber nicht im Fokus der vorliegenden Arbeit liegt. Das EVGM liefert Vorgaben für ein bedarfsgerechtes Tailoring, weil es die Vorgehensmodellmerkmale vorschlägt, die in einem spezifischen Projekt als besonders wichtig beurteilt werden. Die Einführung des Rational Unified Process wird in [Ber04] umfassend und präzise beschrieben. [Wes05] betont, daß die Einführung durch Management Support, Überzeugung der Entwickler und Training unterstützt werden muß.

3.3.5 Merkmale für den Rational Unified Process

3.3.5.1 Unterstützung des Requirements Engineerings im RUP

Die *Disciplines Business Modelling* und *Requirements* mit ihren *Roles*, *Work Products* und *Tasks* unterstützen umfassend und detailliert die Disziplin Requirements Engineering. Die *Roles* sind im *Role Set Analysts* und bei den *Additional Roles* definiert.

Bewertung. In der vorliegenden Arbeit wird für den Rational Unified Process für die EV-UnterstProjTheBer-RequirementsEngineering die Bewertung „**++**“ vergeben.

3.3.5.2 Unterstützung des Software Engineerings im RUP

Die *Disciplines Analysis&Design*, *Implementation* und *Test* unterstützen mit ihren *Roles*, *Work Products* und *Tasks* umfassend und detailliert die Disziplin Software Engineering. Die Roles sind im *Role Set Developers, im Role Set Testers* und bei den *Additional Roles* definiert.

Bewertung. In der vorliegenden Arbeit wird für den Rational Unified Process für die EV-UnterstProjTheBer-SoftwareEngineering die Bewertung „**++**“ vergeben.

3.3.5.3 Unterstützung des Software Managements im RUP

Die *Disciplines Requirements*, sowie *Configuration&Change Management* unterstützen umfassend und detailliert das Software Management. Die *Roles* sind im *Role Set Developers* und im *Role Set Managers* definiert.

Bewertung. In der vorliegenden Arbeit wird für den Rational Unified Process für die EV-UnterstProjTheBer-SoftwareManagement die Bewertung „**++**“ vergeben.

3.3.5.4 Unterstützung des Projektmanagements im RUP

Die *Disciplines Project Management* mit ihren *Roles*, *Work Products* und *Tasks* unterstützt umfassend und detailliert das Projektmanagement. Die *Roles* sind im *Role Set Managers* definiert.

Bewertung. In der vorliegenden Arbeit wird für den Rational Unified Process für die Unterstützung durch Disziplin Projektmanagement die Bewertung „**++**“ vergeben.

3.3.5.5 Unterstützung der Menschenführung im RUP

Das Prinzip *Work together as one team* unterstützt die Disziplin Menschenführung. Es sind keine *Roles* zur expliziten Unterstützung der Menschenführung definiert.

Bewertung. In der vorliegenden Arbeit wird für den Rational Unified Process für die EV-UnterstProjTheBer-Menschenführung die Bewertung „**0**“ vergeben.

3.3.5.6 Flexibilität durch Iterativität und Inkrementalität im RUP

Die EV-Flexib-DurchIterativitätUndInkrementalität unterstützt der Rational Unified Process über seine Best Practice, Software iterativ zu entwickeln, sowie Anforderungen inkrementell umzusetzen. Die *Role Process Engineer* kann der expliziten Unterstützung von Flexibilität dienlich sein. Die Flexibilität des Rational Unified Process wird eingeschränkt durch sein großes Arbeitsproduktmodell, weil dieses bei Veränderungen konsistent gehalten werden muß.

Bewertung. In der vorliegenden Arbeit wird für den Rational Unified Process für die EV-Flexib-DurchIterativitätUndInkrementalität die Bewertung „**+**“ vergeben.

3.3.6 Zusammenfassende Bewertung des Rational Unified Process

Vorgehensmodellmerkmal in Rational Unified Process	Bewertung im EVGM
EV-UnterstProjTheBer-RequirementsEngineering	++
EV-UnterstProjTheBer-SoftwareEngineering	++
EV-UnterstProjTheBer-SoftwareManagement	++
EV-UnterstProjTheBer-ProjektManagement	++
EV-UnterstProjTheBer-Menschenführung	0
EV-Flexib-DurchIterativitätUndInkrementalität	+

Tabelle 6 Bewertung des Rational Unified Process

3.3.7 Charakterisierung von Scrum

Scrum ist ein Ansatz, der sich den Agilen Methoden [AGI06] zuordnet. Er stammt von Ken Schwaber, Mike Beedle und Jeff Sutherland. Es wird in [Sch02], [Sch04] aus der Sicht seiner Autoren und in [Hig02], [Lar04], [Boe04] aus der Sicht Dritter beschrieben. Scrum ist ein iterativer, inkrementeller Ansatz. Es ist gedacht und geeignet für Neuentwicklung von Individualsoftware und betrachtet vorwiegend Aspekte des Software Ma-

nagements im Sinne des Glossars im Anhang. Scrum kann ergänzt werden. Es adressiert keine Software Engineering Aspekte [Abr+03] und macht keine explizite Aussage bezüglich eines Entwicklungsparadigmas. Scrum fordert und fördert bei den Projektmitarbeitern die Haltung, direkt zu kommunizieren [Hol+06].

3.3.7.1 Modellelemente in Scrum

Scrum beschreibt *Roles*, die *Work Products* bearbeiten und dabei *Practices* benutzen.

Roles. Scrum beschreibt den *Scrum Master,* den *Product Owner,* das *Scrum Team Member* und die *Chickens* [sic].

Der *Scrum Master* ist Projektcoach und Prozeßpromotor. Er moderiert die Meetings und sorgt für geeignete Arbeitsbedingungen. Der *Scrum Master* ist für die Einführung und das Coaching von Scrum verantwortlich.

Der *Product Owner* ist eine Art Produkt Manager. Er stellt die Anforderungen, priorisiert sie und entscheidet, welche für die nächste Iteration zur Umsetzung vorgeschlagen werden.

Das *Scrum Team* sagt einen Iterationsumfang zu und ist für dessen Erfüllung gemeinschaftlich verantwortlich. Vom *Scrum Team* wird erwartet, daß es sich selbst organisiert, daß es selbstverantwortlich, heterogen, erfahren, qualifiziert und kommunikativ ist. *Scrum Teams* dürfen nicht verteilt sein und sollen aus fünf bis neun Personen bestehen.

Die *Chickens* sind Personen im Unternehmen, die am Projekt nicht beteiligt sind, aber sich im Rahmen von Meetings informieren wollen. Sie haben kein Mitspracherecht.

Die Anforderungen an die *Roles* und ihre Verantwortlichkeiten werden rudimentär beschrieben. Große Teams können mit den Rollen von Scrum nicht unterstützt werden, weil keine Diversifizierung der *Scrum Team Members* erfolgt.

Artifacts. Scrum beschreibt den *Product Backlog*, den *Iteration Backlog,* den *Backlog Graph,* den *Release Backlog* und das *Increment of Potentially Shippable Product Functionality*.

Der *Product Backlog* enthält priorisierte funktionale und nicht funktionale Anforderungen mit vorläufigen, vagen Aufwandsschätzungen. Er ist eine schlagwortartige Liste, für die der *Product Owner* verantwortlich ist.

Der *Iteration Backlog* enthält die Anforderungen, die Gegenstand einer Iteration sind. Diese sind mit tagesgenauen Aufwandsschätzungen versehen. Der *Iteration Backlog* darf während eines Sprints nur durch *Scrum Team Members* aktualisiert werden.

Der *Backlog Graph* stellt das tagesaktuelle Verhältnis zwischen noch zu realisierenden Funktionen und verbleibender Zeit in einem Sprint dar [Lar04].

Das *Increment of Potentially Shippable Product Functionality* ist die einsatzfähige Software am Ende einer Iteration.

Große zu verarbeitenden Mengen an Projektinformationen werden in Scrum nicht unterstützt.

Aktivitäten. Scrum bietet kein Aktivitätsmodell.

Practices. Sie sind das *Sprint Planning Meeting, Daily Scrum Meeting,* der *Sprint,* das *Sprint Review Meeting* und das *Sprint Retrospective Meeting.*

Das *Sprint Planning Meeting* ist ein maximal achtstündiger Workshop, in dem der Sprint Backlog zusammengestellt wird.

Im *Daily Scrum Meeting* wird in insgesamt maximal fünfzehn Minuten von jedem *Scrum Team Member* dargestellt, was es seit dem letzten *Daily Scrum* erledigt hat, was es bis zum nächsten *Daily Scrum* zu erledigen plant und was es dabei behindern könnte.

Es werden Informationen über den Projektstatus ausgetauscht aber keine Diskussionen geführt.
Der *Sprint* ist eine genau dreißig Kalendertage lange Iteration nach der Art einer Time Box, die das *Scrum Team* eigenverantwortlich und selbst organisiert durchführt. Im Sprint werden die hierfür im *Sprint Backlog* fixierten Anforderungen umgesetzt. Der *Product Owner* und der *Scrum Master* begleiten den *Sprint*.
Das *Sprint Review Meeting* ist eine maximal vierstündige Präsentation der *Sprint* Ergebnisse durch das *Scrum Team*.
Das *Sprint Retrospective Meeting* ist ein maximal dreistündiges Review über den vergangenen Sprint.
Prinzipien. Scrum formuliert die *Values Commitment, Focus, Openness, Respect* und *Courage. Commitment s*teht für die Identifizierung der Mitarbeiter mit den Projektzeilen und den Willen, diese zu erreichen. *Focus* fordert die Konzentration auf die Projektarbeit. *Openness* steht für eine Informationspolitik der freien Verfügbarkeit und Sichtbarkeit von Projektinformationen für jeden, der daran interessiert ist. *Respect* fordert, die unterschiedlichen Hintergründer der Projektmitarbeiter zu respektieren. *Courage* ermutigt, die eben genannten *Values* im Projekt auch zu leben.

3.3.7.2 Darstellung, Detaillierungsgrad von Scrum

Der Detaillierunggrad der Beschreibungen ist niedrig. Die Darstellung von Scrum ist über mehrere, zum Teil widersprüchliche Publikationen verteilt.

3.3.7.3 Überblick über Scrum

Am Beginn jedes Sprints werden in einem *Sprint Planning Meeting* vom *Product Owner* eine Auswahl der am höchsten priorisierten Anforderungen aus dem *Product Backlog* zusammengestellt. Das *Scrum Team* selektiert hieraus einen als realisierbar beurteilten Funktionsumfang. Das *Scrum Team* plant den *Sprint* während dem die zugesagten Anforderungen umgesetzt werden. Das Arbeitsprodukt des *Sprint Planning Meeting* ist der *Sprint Backlog*.
Während des Sprints findet täglich ein *Daily Scrum Meeting* statt, in dem das *Scrum Team* über erledigte Aufgaben, als nächstes geplante Aufgaben und Probleme berichtet und entscheidet. Inhaltliche Diskussionen werden auf ein *Follow-Up-Meeting* im kleinen Kreis verschoben. Der *Sprint* führt auf ein *Sprint Goal*, einen Meilenstein an dem im Rahmen eines *Sprint Review Meetings* ein ausführbares Inkrement den Anwendern präsentiert wird. Der *Product Owner* kann Feedback geben und Änderungswünsche einbringen. Ein *Sprint* kann im Extremfall abgebrochen werden, wenn er sich als nicht realisierbar oder nicht mehr relevant herausstellt. Hierüber entscheidet das *Scrum Team* beziehungsweise der *Scrum Master*.
Am Ende des Sprints wird im *Sprint Retrospective Meeting* diskutiert, was gut funktioniert hat und was beim nächsten Sprint anders laufen sollte. Das *Scrum Team* priorisiert die Themen, der *Scrum Master* moderiert. Die Beteiligung des *Product Owners* ist hier optional.

3.3.7.4 Intention, Eignung, Nutzungsvoraussetzungen von Scrum

Projektgröße. Scrum adressiert kleine Projekte [Boe04] mit instabilen fachlicher Anforderungen [Abr+03]. Größere Projekt sind als „Scrum of Scrums“ grundsätzlich möglich, wofür Scrum keinerlei Unterstützung bietet. Beispielsweise eine hierfür erforderliche Architektur [Eck04] wäre im *Product* oder *Sprint Backlog*, in dem der *Product*

Owner seine Features fordert, fehl am Platz. Für größere Projekte werden hierarchische *Product Backlogs* vorgeschlagen [Sch04], aber keine Vorschläge zu deren Organisation, Synchronisation und Konsistenzsicherung gemacht.

Kritikalität von Projekt und Produkt. Für kritische Projekte bietet Scrum keine Unterstützung. Da es sich aber komplementär zu Engineering Prozessen verhält, können diese entsprechende Unterstützung ergänzen. Daher sind kritische Projekte mit Scrum nicht völlig ausgeschlossen.

Verteilung. Da Scrum sehr stark auf verbale Kommunikation setzt und nur minimale Arbeitsproduktdokumente fordert, sind verteilte Projekte ohne Ergänzungen von Scrum bedenklich.

Flexibilität. Aufgrund seiner kurzen Iterationen und minimaler Arbeitsproduktdokumente kann man mit Scrum flexibel auf Unsicherheiten und Instabilitäten reagieren.

Wiederverwendbarkeit. Scrum bietet keine Unterstützung für wiederverwendbare Software.

Benutzungsanforderungen. Scrum ist aufgrund seiner Beschreibung für das Software Management im Bezug auf das Requirements Management gut benutzbar. Es bietet eine eingeschränkte Abdeckung der Projektdisziplinen. Speziell beim Software Engineering erhält der Anwender von Scrum keine Unterstützung.

3.3.8 Merkmale für Scrum

3.3.8.1 Unterstützung des Requirements Engineerings in Scrum

Das Requirements Engineering im Sinne der Definition im Glossar (siehe Anhang) wird durch Scrum nicht unterstützt.

Bewertung. In der vorliegenden Arbeit wird für Scrum für die EV-UnterstProjTheBer-RequirementsEngineering die Bewertung „**0**“ vergeben.

3.3.8.2 Unterstützung des Software Engineerings in Scrum

Das Software Engineering mit seinen Disziplinen Analyse, Design, Implementierung und Qualitätssicherung im Sinne der Definition im Glossar wird in Scrum nicht unterstützt. Speziell die Qualitätssicherung erfolgt nur durch *Sprint Review Meetings*, in denen der *Owner* Oberflächenfunktionalität bewerten kann. Auch [Bez05] betont, das Scrum den Anforderungen einer umfassenden, methodischen Qualitätssicherung, am Beispiel von inneren Qualitätskriterien wie Security nicht gerecht wird.

Bewertung. In der vorliegenden Arbeit wird für Scrum für die EV-UnterstProjTheBer-SoftwareEngineering die Bewertung „**0**“ vergeben.

3.3.8.3 Unterstützung des Software Managements in Scrum

Der *Product Backlog* unterstützt das Requirements Management [Lar04]. Auch das *Sprint Planning Meeting* adressiert das Requirements Management. Hier unterstützen auch die *Roles Scrum Master, Product Owner* und *Scrum Team Member.*

Das Konfigurationsmanagement, das Qualitätsmanagement und das Change Management im Sinne der Definition des Glossars im Anhang werden in Scrum nicht unterstützt.

Bewertung. In der vorliegenden Arbeit wird für Scrum für die EV-UnterstProjTheBer-SoftwareManagement die Bewertung „**+**“ vergeben.

3.3.8.4 Unterstützung des Projektmanagements in Scrum

Das kaufmännische Projektmanagement im Sinne der Definition im Glossar (siehe Anhang) wird nicht unterstützt, das organisatorische rudimentär. Der *Product Backlog*, *der Release Backlog* und der *Backlog Graph* unterstützen des Projektmanagement eingeschränkt. Hier unterstützt die *Role Scrum Team Member.* Auch das *Sprint Planning Meeting* unterstützt das Projektmanagement.

Bewertung. In der vorliegenden Arbeit wird für Scrum für die EV-UnterstProjTheBer-ProjektManagement die Bewertung „**+**“ vergeben.

3.3.8.5 Unterstützung der Menschenführung in Scrum

Scrum enthält keine konkreten Maßnahmen bezüglich der Motivation von Mitarbeitern. Es fordert und fördert aber die Kommunikation durch die regelmäßigen Meetings, was auch [Man05] bestätigt. Beim *Sprint Planning Meeting, Daily Scrum Meeting, Sprint Review Meeting* und *Sprint Retrospective Meeting* wird Kommunikation gefordert und gefördert. Scrum forciert die Berücksichtigung von Social Factors [Law05]. Somit unterstützt es den Projekthemenbereich „Menschenführung“ im Sinne der Definition im Glossar (siehe Anhang) teilweise. Es unterstützen auch die *Roles Scrum Master* und *Scrum Team Member*. Die *Values Commitment, Focus, Openness, Respect* und *Courage* fördern die Disziplin der Social Factors.

Bewertung. In der vorliegenden Arbeit wird für Scrum für die EV-UnterstProjTheBer-Menschenführung die Bewertung „**+**“ vergeben.

3.3.8.6 Flexibilität durch Iterativität und Inkrementalität in Scrum

Der Umgang mit Unsicherheiten bezüglich der Anforderungen wird in Scrum unterstützt. Um intransparente Anforderungen zu ermitteln, gibt es *Scrum Meetings* mit Benutzern. Instabile fachliche Anforderungen werden mit dreissig-Tages-*Sprints* adressiert. Hier unterstützt die *Role Scrum Team Member.* Das minimale Arbeitsproduktmodell ermöglicht flexible Reaktion auf Veränderungen, weil nur wenige persistente Arbeitsprodukte aktualisiert werden müssen.

Bewertung. In der vorliegenden Arbeit wird für Scrum für die EV-Flexib-DurchIterativitätUndInkrementalität die Bewertung „**++**“ vergeben.

3.3.9 Zusammenfassende Bewertung von Scrum

Vorgehensmodellmerkmal in Scrum	Bewertung im EVGM
EV-UnterstProjTheBer-RequirementsEngineering	0
EV-UnterstProjTheBer-SoftwareEngineering	0
EV-UnterstProjTheBer-SoftwareManagement	+
EV-UnterstProjTheBer-ProjektManagement	+
EV-UnterstProjTheBer-Menschenführung	+
EV-Flexib-DurchIterativitätUndInkrementalität	++

Tabelle 7 Bewertung von Scrum

3.3.10 Charakterisierung des V-Modell XT (VM XT)

Das V-Modell XT wurde im Rahmen der durch das Bundesministerium für Bildung und Forschung (BMBF) geförderten Projekte WEIT I-III durch die Technische Universität München, Institut für Informatik am Lehrstuhl von Prof. Dr. Dr. h.c. Manfred Broy in

Zusammenarbeit mit Stakeholdern aus Industrie und öffentlicher Hand entwickelt. Es stellt den aktuellen Entwicklungsstandard für IT-Systeme des Bundes dar. Das V-Modell XT ist Online unter [VMX06] frei verfügbar.

3.3.10.1 Modellelemente im V-Modell XT

Abgrenzung Projektvorbereitung durch Tailoring. Das V-Modell XT enthält einen Tailoring-Mechanismus zur Vorbereitung eines Projektes. Dieser ist charakterisierend für das V-Modell XT, aber nicht Gegenstand der vorliegenden Arbeit. Das EVGM ermittelt Anforderungen an Vorgehensmodelle, die in einem spezifischen Projekt bestehen. Dies erfolgt unabhängig vom Tailoring-Mechanismus des V-Modell XT. Die Anforderungen an Vorgehensmodelle können als Input für den Tailoring-Mechanismus des V-Modell XT benutzt werden.
In der vorliegenden Arbeit werden die V-Modell XT Modellelemente V*orgehensbaustein, Rolle, Produkt* und *Aktivität* betrachtet. Das EVGM betrachtet das V-Modell XT vorwiegend auf der Basis seiner *Vorgehensbausteine.*

Die ***Vorgehensbausteine*** des V-Modell XT sind
Projektmanagement, Qualitätssicherung, Konfigurationsmanagement, Problem- und Änderungsmanagement, Lieferung und Abnahme (AG), Vertragsschluß (AG), Anforderungsfestlegung, Evaluierung von Fertigprodukten, Systemsicherheit, Kaufmännisches Projektmanagement, Messung und Analyse, Lieferung und Abnahme (AN), Vertragsschluß (AN), Systemerstellung, HW-Entwicklung, SW-Entwicklung, Logistikkonzeption, Benutzbarkeit und Ergonomie, Weiterentwicklung und Migration von Altsystemen, Einführung und Pflege eines organisationsspezifischen Vorgehensmodells, Multi-Projektmanagement.

Die *Rollen, Aktivitäten* und *Produkte* des V-Modell XT sind zu umfangreich, um in diesem Überblick sinnvoll dargestellt werden zu können. Hierfür wird auf die Dokumentation verwiesen [VMX06].
Mit seinem umfangreichen Rollenmodell kann das V-Modell XT die Organisation von großen Teams unterstützen. Durch sein umfangreiches Aktivitätsmodell kann das V-Modell XT die Beherrschung von komplizierten Tätigkeitsstrukturen unterstützen. Es werden keine konkreten Bearbeitungsschritte beschrieben. Durch sein umfangreiches Produktmodell kann das V-Modell XT die Verwaltung und Strukturierung von großen Informationsmengen und deren Abhängigkeiten unterstützen. Das V-Modell XT beschreibt keine Prinzipien und Methoden.
Da die Vorgehensbausteine im V-Modell XT thematisch zusammenhängende Rollen, Aktivitäten und Arbeitsprodukte zusammenfassen, die gut zu den Projektthemenbereichen der vorliegenden Arbeit passen, weisen sie exakt den geeigneten inhaltlichen Zuschnitt auf, um sie Merkmalen zuzuordnen.

3.3.10.2 Darstellung, Detaillierungsgrad des VM XT

Das V-Modell XT ist sehr strukturiert aufgebaut. Seine Modellelemente sind explizit miteinander vernetzt. Ihre Beschreibungen sind umfassend und detailliert. Das V-Modell XT ist in deutsch und englisch verfügbar. Es liegt in elektronischer Form vor.

3.3.10.3 Überblick über das VM XT

Ein beispielhafter Durchlauf durch das V-Modell XT startet mit der Genehmigung eines

Projektes. Hierauf wird die Projektdefinition ausgearbeitet. Es werden Anforderungen festgelegt, und das Projekt ausgeschrieben. Nach der Beauftragung des Projektes werden Iterationen geplant und der Projektfortschritt überprüft. Nach erfolgter Abnahme wird das Projekt abgeschlossen. Diese Darstellung zeigt einen repräsentativen Durchlauf durch das V-Modell XT um einen überblickshaften Eindruck zu vermitteln. Er ist der V-Modell XT Dokumentation entnommen. Weitere Varianten von Projektdurchläufen sind möglich.
Das V-Modell XT unterstützt iteratives Vorgehen. Es ist methodenneutral und paradigmenunabhängig, und dadurch organisationsunabhängig einsetzbar.

3.3.10.4 Intention, Eignung, Nutzungsvoraussetzungen des VM XT

Projektgröße. Das V-Modell XT ist für umfangreiche Projekte [VMX06] entwickelt.
Kritikalität von Projekt und Produkt. Das V-Modell XT ist für sicherheitskritische Projekte geeignet. Die Entwicklung sicherheitskritischer Systeme wird im V-Modell XT beispielsweise durch das Requirements Tracking unterstützt [VMX06].
Verteilung. Es ist an der Auftraggeber-Auftragnehmer-Schnittstelle ausgerichtet, was sich durch viele unterschiedliche Modellelemente für Auftraggeber und Auftragnehmer zeigt [VMX06]. Viele definierte Arbeitsprodukte erhöhen seine Eignung für verteilte Projekte.
Flexibilität. Das V-Modell XT ist durch iteratives Vorgehen geeignet, die Bewältigung von Unsicherheiten, Instabilitäten und Intransparenzen zu unterstützen. Aufgrund des Umfanges seiner obligatorischen Arbeitsprodukte und Aktivitäten ist das flexible reagieren auf umfassende Änderung von Anforderungen jedoch eingeschränkt möglich. Das V-Modell XT ist kein evolutionärer Ansatz.
Wiederverwendbarkeit. Die Entwicklung wiederverwendbarer Software wird auf der Ebene von Methoden des Software Engineering unterstützt. Da das V-Modell XT methodenneutral ist, unterstützt es nicht explizit Wiederverwendbarkeit, beispielsweise durch entsprechende methodische Ansätze oder Verwendung des objektorientierten Paradigmas.
Benutzungsanforderungen. Die Beschreibungstiefe ist geeignet, einen Überblick über die Inhalte von Modellelementen zu geben. Produkte werden durch kommentierte Templates weiter detailliert, Aktivitätsbeschreibungen enthalten keine konkreten Angaben, wie sie durchgeführt werden können.

3.3.11 Merkmale für das V-Modell XT

3.3.11.1 Unterstützung des Requirements Engineerings im V-Modell XT

Das V-Modell XT unterstützt das Requirements Engineering im Sinne der Definition im Glossar im Anhang durch den *Vorgehensbaustein Anforderungsfestlegung.*
Bewertung. In der vorliegenden Arbeit wird für das V-Modell XT für die EV-UnterstProjTheBer-RequirementsEngineering die Bewertung „**+**“ vergeben.

3.3.11.2 Unterstützung des Software Engineerings im V-Modell XT

Das V-Modell XT unterstützt das Software Engineering im Sinne der Definition im Glossar im Anhang durch seine *Vorgehensbausteine Qualitätssicherung, Systemsicherheit, Messung und Analyse, Systemerstellung, SW-Entwicklung, Benutzbarkeit und Ergonomie, Weiterentwicklung und Migration von Altsystemen.*

Bewertung. In der vorliegenden Arbeit wird für das V-Modell XT für die EV-UnterstProjTheBer-SoftwareEngineering die Bewertung „**++**“ vergeben.

3.3.11.3 Unterstützung des Software Managements im V-Modell XT

Das V-Modell XT unterstützt das Software Management im Sinne der Definition im Glossar durch die *Vorgehensbausteine Konfigurationsmanagement, Problem- und Änderungsmanagement* und *Anforderungsfestlegung.*

Bewertung. In der vorliegenden Arbeit wird für das V-Modell XT für die EV-UnterstProjTheBer-SoftwareManagement die Bewertung „**++**“ vergeben.

3.3.11.4 Unterstützung des Projektmanagements im V-Modell XT

Das V-Modell XT unterstützt das Projektmanagement im Sinne der Definition im Glossar im Anhang durch die Vorgehensbausteine *Projektmanagement, Lieferung und Abnahme (AG), Vertragsschluß (AG), Kaufmännisches Projektmanagement, Lieferung und Abnahme (AN), Vertragsschluß (AN), Multi-Projektmanagement.*

Bewertung. In der vorliegenden Arbeit wird für das V-Modell XT für die EV-UnterstProjTheBer-ProjektManagement die Bewertung „**++**“ vergeben.

3.3.11.5 Unterstützung der Menschenführung im V-Modell XT

Menschenführung wird im V-Modell XT nicht explizit adressiert.

Bewertung. In der vorliegenden Arbeit wird für das V-Modell XT für die EV-UnterstProjTheBer-Menschenführung die Bewertung „**0**“ vergeben.

3.3.11.6 Flexibilität durch Iterativität und Inkrementalität im V-Modell XT

Fachliche Intransparenzen und Instabilitäten adressiert das V-Modell XT durch die Unterstützung iterativen und inkrementellen Vorgehens. Aufgrund des umfangreichen verbindlichen Produktmodells ist das V-Modell XT jedoch kein evolutionärer Ansatz.

Bewertung. In der vorliegenden Arbeit wird für das V-Modell XT für die EV-UnterstProjTheBer-RequirementsEngineering die Bewertung „**-**“ vergeben.

3.3.12 Zusammenfassende Bewertung des V-Modell XT

Vorgehensmodellmerkmal in V-Modell XT	Bewertung im EVGM
EV-UnterstProjTheBer-RequirementsEngineering	+
EV-UnterstProjTheBer-SoftwareEngineering	++
EV-UnterstProjTheBer-SoftwareManagement	++
EV-UnterstProjTheBer-ProjektManagement	++
EV-UnterstProjTheBer-Menschenführung	0
EV-Flexib-DurchIterativitätUndInkrementalität	-

Tabelle 8 Bewertung des V-Modell XT

3.3.13 Vorgehensmodellspezifische Bewertungen

Die in den letzten Abschnitten hergeleiteten vorgehensmodellspezifischen Bewertungen der Vorgehensmodellmerkmale wird in der folgenden Tabelle 9 noch einmal im Überblick dargestellt. Die einzelnen Zellen stehen hierbei jeweils für eine spezifische Bewertung des Vorgehensmodellmerkmals im Zeilenkopf für das Vorgehensmodell im Spaltenkopf. In der Zelle links oben steht also die Bewertung des Merkmals EV-UnterstProj-

TheBer-RequirementsEngineering für Extreme Programming, also ein „+“.

Vorgehensmodell *Vorgehensmodell merkmal*	*Vorgehensmodells-pezifische Bewer-tung für Extreme Programming*	*Vorgehensmodells-pezifische Bewer-tung für den Rational Unified Process*	*Vorgehensmodells-pezifische Bewer-tung für Scrum*	*Vorgehensmodells-pezifische Bewer-tung für das V-Modell XT*
EV-UnterstProjThe-Ber-RequirementsEn-gineering	+	++	0	+
EV-UnterstProjThe-Ber-SoftwareEngi-neering	+	++	0	++
EV-UnterstProjThe-Ber-SoftwareMana-gement	+	++	+	++
EV-UnterstProjThe-Ber-ProjektManage-ment	+	++	+	++
EV-UnterstProjThe-Ber-Menschenfüh-rung	++	0	+	0
EV-Flexib-DurchIte-rativitätUndInkre-mentalität	++	+	++	-

Tabelle 9 Zusammenfassung der Vorgehensmodellbewertungen

Zusammenfassung der Charakterisierung von Vorgehensmodellen. Ein Kriterienschema zur Charakterisierung von Vorgehensmodellen bezüglich ihrer Eignungseinstufung wurde definiert und motiviert. Dann wurden die für das EVGM relevanten Vorgehensmodelle nach diesen Kriterien charakterisiert, um für die eigentliche Eignungseinstufung aufbereitet zu sein.

Bezug zum EVGM. Im diesem Kapitel wurden Vorgehensmodelle mit Bezug auf ihre Eignungseinstufung charakterisiert. Hierbei stand im Vordergrund, wie Vorgehensmodelle klassischen Problemstellungen in Software Engineering Projekten adressieren, wie sie in den Projektmerkmalen in Kapitel 2„Charakterisierung von Projekten“ dargestellt werden. Hierdurch wurden die Vorgehensmodelle aufbereitet, um im nächsten Kapitel anhand ihrer Eigenschaften im EVGM verwendet zu werden.

4 Das Modellierungsverfahren im EVGM

In diesem Kapitel werden die Projektmerkmale aus Kapitel 2 und die Vorgehensmodellmerkmale aus Kapitel 3 in einem methodischen Modellierungsverfahren verwendet.
Das Kapitel hat folgende Struktur:
Im Abschnitt 4.1 „Überblick über das EVGM“ werden die Modellelementtypen des EVGM vorgestellt und ihre Benutzung skizziert. Im Abschnitt 4.2 „Prinzipien des EVGM“ werden die Grundsätze beschrieben, an denen sich das EVGM ausrichtet. Arbeiten mit ähnlichen Zielsetzungen oder Ansätzen werden im Abschnitt 4.3 „Verwandte Arbeiten“ dargestellt. Der Abschnitt 4.4 „Ursache-Wirkungs-Diagramme als Notation des EVGM“ führt in die Sprache ein, mit der die Modelle des EVGM dargestellt werden. In Abschnitt 4.5 „Rollen im EVGM“ werden Verantwortungsbereiche definiert, seine Arbeitsprodukte in Abschnitt 4.6 „Arbeitsprodukte im EVGM“ und seine Vorgehensweise in Abschnitt 4.7 „Aktivitäten im EVGM“. Der Abschnitt 4.8 „Exemplarischer Durchlauf durch das EVGM“ verdeutlicht die Anwendung des EVGM anhand eines Durchlaufs mit einem fiktiven Projekt.

Zusammenfassung. Das EVGM ist ein Verfahren, welches ein Vorhaben zur Verbesserung von Softwareentwicklungsprozessen methodisch unterstützt. Es systematisiert umfassend die Analyse der wesentlichen Merkmale eines Softwareentwicklungsprojektes und deren wechselseitige Beeinflussungsbeziehungen. Das EVGM leitet seine Benutzer methodisch an, durch Vorgehensmodelle vorhandene Lösungsmöglichkeiten für die Probleme in einem Projekt zu identifizieren. Durch Schritt-für-Schritt-Anweisungen wird die Eignungseinstufung von Vorgehensmodellen für die Zielgruppe besser beherrschbar, sowie für Außenstehende nachvollziehbar und überprüfbar. Das EVGM betrachtet dabei fachliche, Softwareentwicklungs-, Projektmanagement- und Menschenführungs-Aspekte.
Es definiert Rollen, Arbeitsprodukte, Aktivitäten und eine grafische Notation. Die Rollen werden von den Anwendern des EVGM besetzt. Sie beschreiben die Verantwortlichkeiten der Anwender des EVGM (dem EVGM-Team) bezüglich der Arbeitsprodukte, sowie die hierfür wünschenswerten Fähigkeiten.
Es gibt im EVGM zwei Arten von Arbeitsprodukten. Im EVGM fertig vorgegeben sind Merkmale zur Beschreibung von Projekten und von Vorgehensmodellen, sowie Beeinflussungsbeziehungen zwischen Merkmalen (siehe Kapitel 2 und 3). Nur bezüglich ihrer Eigenschaften beschrieben und durch die Anwendung des EVGM jeweils entwickelt werden die Modelle des zu untersuchenden Projektes, die mit Ursache-Wirkungs-Diagrammen dargestellt werden.
Die Aktivitäten beschreiben, wie das EVGM angewendet wird. Aktivitäten veranschaulichen, wie die im Rahmen des EVGM bereits vorgefertigten Arbeitsprodukte als Unterstützung und Anregung benutzt werden, um die bei der Durchführung des EVGM noch anzufertigenden Arbeitsprodukte zu entwickeln. Aktivitäten beschreiben, wie die vorgegebenen Merkmale und die Beeinflussungsbeziehungen verwendet werden, um die neu zu entwickelnden Modelle zu erstellen. Die grafische Notation des EVGM ist die Grundlage für die Modellierung der Ursache-Wirkungs-Diagramme.
Im Fließtext namentlich referenzierte Modellelemente werden *kursiv* geschrieben.

Inhalt des Kapitels

4.1 Überblick über das EVGM

Zunächst werden die Merkmale und Beeinflussungsbeziehungen als Modellelemente im EVGM vorgestellt. Hierauf wird die Benutzung der Modellelemente beim Ablauf des EVGM veranschaulicht.

4.1.1 Merkmale und Beeinflussungsbeziehungen

Die Modellierung im EVGM erfolgt durch Modellelemente. **Merkmale** sind Modellelemente, die wesentliche Einflußfaktoren im Projekt repräsentieren. Merkmale werden, wenn sie als gefährdend für den Projekterfolg eingeschätzt werden, in der Modellierung mit einem negativen Zustand versehen. Wenn sie als förderlich für den Projekterfolg eingeschätzt werden, erhalten sie einen positiven Zustand. Die Merkmale von Projekten wurden in Kapitel 2 vorgestellt, die von Vorgehensmodellen in Kapitel 3.
Zwischen Merkmalen werden **Beeinflussungsbeziehungen** modelliert, durch die sie gegenseitig ihren Zustand beeinflussen. Merkmale verändern je nach Beeinflussung durch andere Merkmale ihren Zustand. Ziel ist es hierbei, jedes Merkmal, das sich in einem projektgefährdenden Zustand befindet, durch geeignete Vorgehensmodellmerkmale zu beeinflussen und so in einen projektförderlichen Zustand zu bringen. Die Beeinflussungsbeziehungen zwischen Merkmalen wurden in Kapitel 2 und 3 eingeführt.

4.1.2 Benutzung der Modellelemente beim Ablauf des EVGM

Der Ablauf des EVGM wurde bereits im Kapitel 1.5 in der Abbildung 1 grafisch veranschaulicht.

1. Mit Hilfe der **Fragen zu Einflußfaktoren aus dem Projektumfeld** aus Kapitel 2.4 untersuchen die Anwender des EVGM das projektspezifische Entwicklungs-Umfeld. Die Fragen zu Einflußfaktoren aus dem Projektumfeld führen auf eine erste Auswahl von wesentlichen Projektmerkmalen und auf Ansätze für deren projektspezifische Bewertungen.
2. Diese Auswahl ist die Basis für die Modellierung EVGM, bei der nun die von den Projektbeteiligten als wichtig beurteilten **Projektmerkmale** in Bezug auf das Entwicklungs-Produkt (siehe Kapitel 2) identifiziert und in einem Ursache-Wirkungs-Diagramm, dem *Modell mit Merkmalen Entwicklungs-Produkt* modelliert werden.
3. Auch bei der nun folgenden Identifikation von personellen Projektmerkmalen können die Benutzer des EVGM die Fragen zu Einflußfaktoren aus dem Projektumfeld aus

Kapitel 2, sowie die vom EVGM angebotenen Projektmerkmale aus dem Risikobereich Entwicklungs-Team als Hilfe heranziehen. Diese Merkmale werden durch die Benutzer des EVGM mit den Projektmerkmalen aus Schritt 2 durch Beeinflussungsbeziehungen vernetzt und ihre Zustände somit beeinflußt. Das *Modell mit Merkmalen Entwicklungs-Produkt* wird durch die personellen Merkmale ergänzt. Dieser Modellierungsschritt wird durch im EVGM vorgegebene Beeinflussungsbeziehungen zwischen Projektmerkmalen unterstützt (siehe Kapitel 2). Das Arbeitsprodukt dieses Modellierungsschritts ist das *Modell mit Merkmalen Entwicklungs-Team*, eine Erweiterung von dem *Modell mit Merkmalen Entwicklungs-Produkt*.
Somit wird ein umfassendes Modell mit allen erfolgsbestimmenden positiven und negativen Einflüssen durch das Entwicklungs-Produkt und das Entwicklungs-Team (siehe Kapitel 2) dargestellt und vernetzt. Das *Modell mit Merkmalen Entwicklungs-Team* repräsentiert den Ist-Zustand im Projekt.

4. Im nächsten Schritt adressieren die Benutzer des EVGM alle Merkmale im *Modell mit Merkmalen Entwicklungs-Team*, deren Zustand als gefährdend für den Projekterfolg eingeschätzt wird. Dies erfolgt durch Hinzufügen von für die Verbesserung erforderlichen Vorgehensmodellmerkmalen. Diese repräsentieren die Anforderungen an die Eigenschaften eines projektspezifisch ideal geeigneten Vorgehensmodells. Dieser Modellierungsschritt hat zum Ziel, zunächst einen von den konkreten, in der vorliegenden Arbeit berücksichtigten Vorgehensmodellen noch unabhängigen Lösungsansatz zu formulieren. Auch dieser Modellierungsschritt wird durch im EVGM vorgegebene Beeinflussungsbeziehungen zwischen Projektmerkmalen und Vorgehensmodellmerkmalen unterstützt (siehe Kapitel 3). Das Arbeitsprodukt des Modellierungsschritts ist das *Modell mit Anforderungen an ideales Vorgehensmodell*, eine Erweiterung des *Modells mit Merkmalen Entwicklungs-Team*.
5. Dann wird durch die Benutzer des EVGM eine Vorauswahl aus den im EVGM berücksichtigen Vorgehensmodelle getroffen. Hierbei wird das EVGM-Team durch die Bewertungen der Vorgehensmodelle in Kapitel 3 unterstützt. Jeweils pro in Betracht gezogenen Vorgehensmodell wird durch die Benutzer des EVGM aus dem *Modell mit Anforderungen an ideales Vorgehensmodell* ein V*orgehensmodellspezifisches Lösungsmodell* erstellt. Hierbei ersetzen die Benutzer des EVGM jeweils die geforderten Bewertungen der Vorgehensmodellmerkmale durch die vom EVGM im Abschnitt 3.3 „Vorgehensmodellspezifische Bewertungen der Merkmale“ vorgeschlagenen Bewertungen der Vorgehensmodellmerkmale für das jeweilige Vorgehensmodell. Auf diese Weise ist es möglich, die Wirkweise der zur Auswahl stehenden Vorgehensmodelle dezidiert darzustellen und zu vergleichen.
6. Auf der Basis des Vergleichs der V*orgehensmodellspezifischen Lösungsmodelle* wird von den Benutzern des EVGM eine Entscheidung für ein Vorgehensmodell getroffen. Das Tailoring von Vorgehensmodellen für ein spezifisches Projekt ist nicht Gegenstand des EVGM. Das EVGM liefert aber im *Modell mit Anforderungen an ideales Vorgehensmodell* Anforderungen an das ausgewählte Vorgehensmodell als Hinweis für dessen Tailoring oder Customizing.

4.2 Prinzipien des EVGM

Das EVGM stützt sich auf die folgenden Prinzipien (umfassende Darstellung siehe Kapitel 5.5), die seine Anwender bei ihren Überlegungen und Entscheidungen zu Rate ziehen können.

Vollständige Betrachtung. Es müssen alle wesentlichen Merkmale im Projekt be-

rücksichtigt, alle projektgefährdend eingestuften durch Maßnahmen adressiert werden.
Geeigneter Potentialeinsatz. Die Fähigkeiten des Projektteams und die Unterstützung durch Vorgehensmodelle werden im Lösungsansatz als sich ergänzend betrachtet.
Übergreifende Betrachtung. Auch die Einflüsse aus dem Projektumfeld werden berücksichtigt.
Fokussierte Betrachtung. Es werden nur nachweislich wesentliche Merkmale vorgeschlagen.

4.3 Verwandte Arbeiten

Die Modellierungsmethode im EVGM orientiert sich an der Methode von Gomez und Probst [Gom99], welche auf dem System Dynamics Ansatz von Jay Forrester [For80] aufbaut. Beide Ansätze werden im Kapitel 5.2 kurz skizziert. Weitere Ansätze, mit System Dynamics Probleme zu modellieren, die der Themenstellung der vorliegenden Arbeit ähnlich sind, werden dort kurz vorgestellt.

4.4 Ursache-Wirkungs-Diagramme als Notation des EVGM

Ursache-Wirkungs-Diagramme [Gom99] sind die Notation der EVGM-Modelle. Die Modellierung mit Ursache-Wirkungs-Diagrammen stellt den methodischen Kern des EVGM dar. Sie bietet während der Problemanalyse und des Lösungsentwurfs einen visualisierten Überblick über die vernetzten Zusammenhänge von Merkmalen in Projektgefährdendem Zustand und Merkmalen in projektförderlichem Zustand. Die Notation der Ursache-Wirkungs-Diagramme vermeidet, daß einzelne Merkmale isoliert betrachtet und ohne die Beachtung von Seiteneffekten verändert oder eingefügt werden. Bei der Modellierung der Ursache-Wirkungs-Diagramme steht immer die Fokussierung auf die erfolgsbestimmenden Merkmale des aktuellen Projektes im Vordergrund. Die Modellierung mit Ursache-Wirkungs-Diagrammen ermöglicht weiterhin das Darstellen, Untersuchen und Vergleichen unterschiedlicher Lösungsoptionen.
Das EVGM benutzt eine grafische Notation, die in nachfolgender Abbildung 29 veranschaulicht und darauffolgend beschrieben wird.

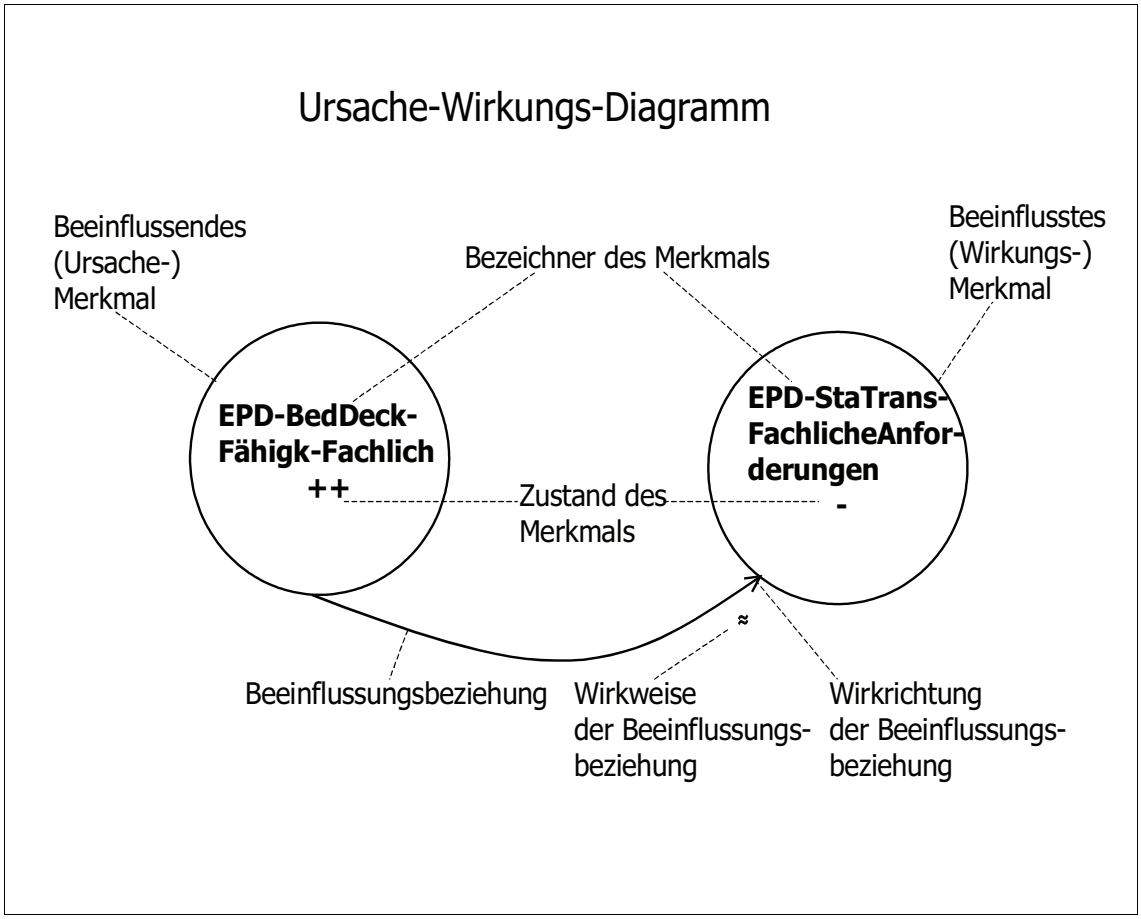

Abbildung 29 Ursache-Wirkungs-Diagramm

Merkmale stellen die beeinflussenden und die beeinflußten Faktoren in Ursache-Wirkungs-Diagrammen dar. Jedes Merkmal hat einen Zustand, der von „--“ für „sehr gefährdend für den Projekterfolg“ und „-“ für „gefährdend für den Projekterfolg“ über „0“ für neutral und „+“ für „förderlich für den Projekterfolg“ bis „++“ für „sehr förderlich für den Projekterfolg“ geht. Der Zustand eines Merkmals bewegt sich zwischen einer unteren („--“) und oberen („++“) Grenze. Der Zustandsverlauf zwischen diesen Grenzen wird optisch etwa in der Form einer Tangens hyperbolicus Funktion angenommen. Hin zu den Extremwerten benötigt es immer stärkere Einflüsse, um den Zustand noch zu verändern.
Merkmale werden in den Ursache-Wirkungs-Diagrammen grafisch als Kreis dargestellt und mit ihrem Bezeichner, sowie ihrem Zustandswert versehen. Jedes Merkmal kann Ursache und Wirkung zugleich sein. Dies ist nur abhängig von den Beeinflussungsbeziehungen, in denen es steht.
Beispiel: Merkmale sind *EPD-BedDeck-Fähigk-Fachlich* oder *EPD-StaTrans-FachlicheAnforderungen* (siehe Abbildung oben)

Beeinflussungsbeziehungen stellen die Abhängigkeiten zwischen Merkmalen dar. Beeinflussungsbeziehungen stellen dar, wie Merkmale in Wechselwirkung stehen und dabei ihren Zustand verändern.
Beeinflussungsbeziehungen haben eine Wirkrichtung, die besagt, welches das beeinflusste und welches das beeinflussende Merkmal ist.
Ihre Wirkweise legt fest, ob die Beeinflussung kausal nebenläufig („≈“) oder gegenläu-

fig („1/≈“) (siehe auch Glossar im Anhang) ist. Gegenläufig besagt, daß bei einer Zustandsverbesserung des beeinflussenden Merkmals eine Zustandsverschlechterung des beeinflußten Merkmals eintritt und umgekehrt. Nebenläufig heißt, daß mit einer Zustandsverbesserung eines beeinflussenden Merkmals auch eine Zustandsverbesserung des beeinflussten Merkmals eintritt und umgekehrt.

Beispiel: Eine Verbesserung des Zustandes der *EPD-BedDeck-Fähigk-Fachlich* bewirkt über eine kausal nebenläufige Beeinflussungsbeziehung („≈“) eine Verbesserung des Zustandes der *EPD-StaTrans-FachlicheAnforderungen* (siehe Kapitel 2 beziehungsweise Abbildung oben).

Beeinflussungsbeziehungen werden grafisch als Pfeil dargestellt, der von dem beeinflussenden zum beeinflussten Merkmal zeigt. Ihre Wirkweise wird von „1/≈≈“ für „stark gegenläufig“ über „1/≈“ für „gegenläufig“ und „≈“ für „nebenläufig“ bis „≈≈“ für „stark nebenläufig“ dargestellt. „0“ würde für „kein Einfluss“ stehen, dies wird allerdings nicht modelliert. Mit dieser Notation kann auch ausgedrückt werden, daß ein Ursache-Merkmal ein Wirkungs-Merkmal nebenläufig, ein anderes aber gegenläufig beeinflusst, da dies durch unterschiedliche Beeinflussungsbeziehungen ausgedrückt wird.

Anmerkung zu dieser Notation. Es muß verdeutlicht werden, daß im Diagramm eine Kennzeichnung „≈“ einer Beeinflussungsbeziehung bei einer Verschlechterung des Zustandes des Ursache-Merkmals auch eine Verschlechterung des Zustandes des Wirkungs-Merkmals bedeutet.

Beispiel: Eine Verschlechterung des Zustandes der *EPD-BedDeck-Fähigk-Fachlich* bewirkt über eine kausal nebenläufige Beeinflussungsbeziehung eine Verschlechterung des Zustandes der *EPD-StaTrans-FachlicheAnforderungen* (siehe Kapitel 2 beziehungsweise Abbildung oben).

Die Kennzeichnung „1/≈“ an einer Beeinflussungsbeziehung bedeutet hingegen bei einer Verschlechterung des Zustandes des Ursache-Merkmals eine Verbesserung des Wirkungs-Merkmals. Auf den Zustands- und Beeinflussungswerten wird nicht gerechnet, sondern sie werden durch die Benutzer des EVGM eingeschätzt.

Begrenzung der Ausdrucksmöglichkeiten. Gegen- beziehungsweise Nebenläufigkeit einer Beeinflussungsbeziehung sind in der Realität nicht gänzlich unabhängig vom Zustandsverlauf des Ursache-Merkmals. Beispielsweise wirkt sich eine Verringerung der *EPP-BedDeck-Kap-KalendarischeZeit* ab einem gewissen Ausmaß vermutlich vermindernd auf die *EPM-BedDeck-MotivationImTeam* aus, wirkt also nebenläufig. Eine Erhöhung der *EPP-BedDeck-Kap-KalendarischeZeit* hat aber tendenziell keine Erhöhung der *EPM-BedDeck-MotivationImTeam* zur Folge. Zusammenhänge wie diese werden im EVGM nicht als Beeinflussungsbeziehung vorgeschlagen weil sie nicht projektübergreifend gültig sind. Die Modellierungsnotation des EVGM bietet für solche Spezialfälle kein Ausdrucksmittel, weil hierfür die Verhältnismäßigkeit nicht vorhanden ist und weil dadurch die Modellierungssprache möglichst einfach und verständlich gehalten wird. Es sind jedoch im Bedarfsfall spezifische informelle Ergänzungen in den Ursache-Wirkungs-Diagrammen möglich, wenn das EVGM-Team einen wesentlichen Sachverhalt formulieren möchte. Die Entscheidung über die Notwendigkeit solcher Ergänzungen obliegt der Rollenverantwortung der Team-Mitglieder.

4.5 Rollen im EVGM

Rollen sind im EVGM Verantwortungsbereiche, die einer definierten Intention entsprechen und für die festgelegte Fähigkeiten erforderlich sind. Rollen werden von Personen besetzt.

Die **Intention** einer Rolle beschreibt, warum es diese Rolle gibt und welcher Nutzen ihr für das EVGM zugedacht ist. Die Intention einer Rolle bestimmt und rechtfertigt die für sie geforderten Fähigkeiten.
Die **Fähigkeiten** einer Rolle beschreiben, welche Kenntnisse, Erfahrungen und Fertigkeiten eine Person besitzen muß, um sie ausfüllen zu können.
Im EVGM werden **keine differenzierten Verantwortlichkeiten von Rollen** für Arbeitsprodukte vergeben, da dies aufgrund seiner Kompaktheit nicht erforderlich ist. Im EVGM sind alle Rollen im Rahmen ihrer Möglichkeiten für alle Arbeitsprodukte verantwortlich.
Im EVGM gibt es die Rollen **Prozeß-Berater** und **Prozeß-Beteiligter**, die nachfolgend definiert werden. Die Verantwortlichkeiten werden also zwischen einer beratenden und einer beratenen Rolle aufgeteilt. Das Rollenmodell im EVGM ist in Anlehnung an die Rollen „Prozeßentwickler“ beziehungsweise „Stakeholder“ in [Zie05] gestaltet, wo ein Verfahren zur Softwareprozeßverbesserung beschrieben wird.

4.5.1 Die Rolle Prozeß-Berater

Intention für den Prozeß-Berater

Der *Prozeß-Berater* ist dafür vorgesehen, von der betroffenen Organisation unabhängiges, externes Wissen und Erfahrung über Softwareentwicklungsmanagement in das EVGM-Verfahren einzubringen. Er steuert einen objektiven Blickwinkel auf die projektspezifische Problemstellung bei und bewirkt, daß die Prozeß-Beteiligten während der Anwendung des EVGM über die gewohnten Bahnen hinausdenken, um Lösungsräume nicht zu früh einzuengen. Der *Prozeß-Berater* soll darum nicht direkt aus der zu beratenden Organisation stammen.

Fähigkeiten des Prozeß-Beraters

Der *Prozeß-Berater* braucht Basiswissen über Softwareentwicklung und Softwaremanagement. Er muß kommunikative Fähigkeiten haben, Kreativität wecken und moderieren können. Der Prozeßberater muß kein spezielles Wissen über Vorgehensmodelle besitzen und auch nicht notwendigerweise im Vorfeld detailliert mit dem EVGM vertraut sein.

4.5.2 Die Rolle Prozeß-Beteiligter

Intention für den Prozeß-Beteiligten

Die *Prozeß-Beteiligten* repräsentieren die späteren Anwender des mit Hilfe des EVGM auszuwählenden Vorgehensmodells. Dies sind beispielsweise Anforderungsanalytiker, Softwareentwickler, Projektmanager oder Qualitätssicherer. Es können auch Mitglieder der Prozeß- und Methodenabteilung beteiligt sein. Die *Prozeß-Beteiligten* sollten so weit wie möglich Mitglieder des Projektteams sein, welches später mit dem auszuwählenden Vorgehensmodell arbeiten muß. Dadurch haben sie ein persönliches Interesse an der gewissenhaften und bedarfsgerechten Auswahl eines Vorgehensmodells, da seine Eignung bedeutsam für den Erfolg des betreffenden Softwareentwicklungsprojektes ist. Die *Prozeß-Beteiligten* bringen die Kenntnis und interne Expertise über das anstehende Softwareprojekt in das EVGM-Verfahren ein und können dessen wesentliche projektspezifische Problemstellungen einschätzen. Die Rolle *Prozeß-Beteiligter* trägt der Erkenntnis Rechnung, daß Prozeßverbesserungen wie sie die Einführung eines Vorgehensmodells darstellen, immer von innen auf breiter Basis mitgetragen werden müssen wie [Hum90] mit „everyone must be involved“ darstellt. Auch [Ber04] betont, daß nicht nur externe Berater an der Einführung eines Vorgehensmodells beteiligt sein dürfen. Einer der *Prozeß-Beteiligten* sollte nach der Entscheidung für ein Vorgehensmodell der Pro-

zeßverantwortliche dafür werden.

Fähigkeiten des Prozeß-Beteiligten

Die *Prozeß-Beteiligten* müssen gute Kenntnisse bezüglich des betreffenden spezifischen Projekts besitzen. Sie müssen Wissen und Erfahrung in Bezug auf Softwareentwicklung besitzen und die Fähigkeit haben, in Softwareentwicklungs-Prozessen zu denken. Sie brauchen dafür prozessuales Denk- und Vorstellungsvermögen, sowie Gestaltungspotential.

Abgrenzung denkbarer Rollen. Da sich das EVGM auf die Modellierung der Projekt- und Vorgehensmodell-Eigenschaften fokussiert und dadurch die Belange eines übergeordneten Prozeßverbesserungsvorhabens und des Veränderungsmanagements mit ihren unternehmenspolitischen Facetten nur tangiert, beschreibt es keine Rollen wie beispielsweise [Ber04] mit “Process Owner”, “Process Accelerator”, “Process Mäzen“ oder „Process Mentor“.

4.6 Arbeitsprodukte im EVGM

Mit Hilfe der Arbeitsprodukte wird der gesamte Verlauf der Eignungseinstufung während der Anwendung des EVGM dokumentiert, es werden Aktivitäten unterstützt und erarbeitete Erkenntnisse über das spezifische Projekt festgehalten. Bereits als Bestandteil des EVGM vorgefertigte Arbeitsprodukte in Form von Merkmalen und Beeinflussungsbeziehungen wurden in den Kapitel 2 und 3 hergeleitet. Auf den Arbeitsprodukten des EVGM operieren die Rollen des EVGM mit den Aktivitäten des EVGM.
Jedes der hier aufgeführten Arbeitsprodukte ist bei der Anwendung des EVGM verbindlich zu benutzen beziehungsweise zu erstellen.
Im EVGM sind vorgefertigte und anzufertigende Arbeitsprodukte definiert. Vorgefertigte Arbeitsprodukte sind Merkmale und Beeinflussungsbeziehungen. Anzufertigende Arbeitsprodukte sind die zu modellierenden Ursache-Wirkungs-Diagramme.

Das Kapitel beschreibt die Arbeitsprodukte
- Risikoliste und Fragen zu Einflußfaktoren aus dem Projektumfeld
- Projektmerkmale
- Modell mit Merkmalen Entwicklungs-Produkt
- Modell mit Merkmalen Entwicklungs-Team
- Vorgehensmodellmerkmale
- Modell mit Anforderungen an ideales Vorgehensmodell
- Vorgehensmodellspezifisches Lösungsmodell

Die Arbeitsprodukte sind in der Reihenfolge ihrer Verwendung im EVGM aufgeführt.

4.6.1 Risikoliste, Fragen zu Einflußfaktoren aus dem Projektumfeld

Intention

Die *Risikoliste* in Tabelle 10 macht klassische Risiken für Vorhaben zur Softwareprozeßverbesserung und Softwareentwicklung für das EVGM-Team nutzbar.
Die *Fragen zu Einflußfaktoren aus dem Projektumfeld* bieten dem EVGM-Team Hinweise für die Selektion und Bewertung von Projektmerkmalen.

Inhalte und Eigenschaften

Die *Risikoliste* ist eine Aufstellung aus der Literatur bekannter Risiken für Vorhaben zur Softwareprozeßverbesserung und zum Veränderungsmanagement. Sie enthält für jedes aufgeführte Risiko Vorschläge für Gegenmaßnahmen. Die während der Aktivität *Risi-*

komanagement betreiben vom EVGM-Team identifizierten projektspezifischen Risiken und Gegenmaßnahmen werden der Risikoliste hinzugefügt. Die Risiken sind nach „Ziele“, „Know-How“, „Management und Migration“und „Human Factors und Unternehmenspolitik“ kategorisiert.

SPI-Risiko	Maßnahme
Risiken bezogen auf die SPI-Ziele	
Die Ziele eines Prozeßverbesserungsprogramms sind von den Beteiligten oft nicht verstanden.	„SPI-Goals well understood“ bei [Gol95] zitiert nach [Ema+99]: SPI-Ziele in Workshop mit EVGM-Team erarbeiten und abstimmen
Die Ziele eines Prozeßverbesserungsprogramms adressieren oft nicht den Kern der Probleme und sind nicht realistisch.	„Setting relevant and realistic objectives“ [Ste98]: Ziele kritisch überprüfen, aus vielen Sichten des EVGM-Teams betrachten
Risiken bezogen auf Know-how für das SPI	
Der aktuelle Prozeß ist den Beteiligten oft unbekannt.	„Effective Change requires a goal and knowledge of the current process“ [Hum90]: SPI-Ziele in Workshop mit EVGM-Team erarbeiten und abstimmen, sowie Anforderungen an Rolle Prozeß-Beteiligter und ihr Staffing
Die Zusammenhänge von Prozessen und Mechanismen der Prozeßverbesserung sind den Beteiligten oft nicht vertraut.	„Providing enhanced understanding“ [Ste98]: Anforderungen an Rolle Prozeß-Beteiligter und ihr Staffing
Risiken bezogen auf Management des SPI und die Migration	
Prozeßverbesserungsvorhaben werden von den Beteiligten oft temporäre Erscheinung gesehen.	„Change is continuous“ [Hum90]: EVGM zyklisch wiederholen, VGM optimieren wie bei CMMI-Level 5 gefordert
Prozeßverbesserungsvorhaben werden oft ohne fachkundige Unterstützung durchgeführt.	„Need Guidance to Improve“ bei [Gol95] zitiert nach [Ema+99] und „Need more Monitoring and Assistance“ bei [Gol95] zitiert nach [Ema+99]: Anforderungen des EVGM an die Rolle des Prozeß-Beraters
Prozeßverbesserungsmaßnahmen werden oft nicht aktiv und nachhaltig in die Softwareentwicklungsprojekte hineingetragen.	„Software process changes will not be retained without conscious effort and periodic reinforcement“ [Hum90]: Längerfristiges Prozeß-Coaching durchführen
Prozeßverbesserungsvorhaben werden oft fälschlicherweise so eingeschätzt, daß man sie ohne Ressourcenbereitstellung nebenher erledigen kann	„Software process improvement requires investment“ [Hum90]: Business Case rechnen, Studien heranziehen, zusätzlichen Ressourcenbedarf einplanen
Prozeßverbesserungsvorhaben werden oft nicht als Projekt organisiert und managed.	„Managing the improvement project“ [Ste98]: Anforderungen an Rolle Prozeß-Berater; Prozeßverbesserungsprogramm als Projekt planen, kontrollieren, steuern
Risiken bezogen auf Human Factors und Unternehmenspolitik	
Bei Prozeßverbesserungsvorhaben werden oft die Individualziele von Beteiligten außer acht gelassen: „Existence of organizational politics“ bei [Gol95] zitiert nach [Ema +99]	Beim Staffing Meinungsführer und führende Manager mit einbeziehen
Bei Prozeßverbesserungsvorhaben erfolgt oft keine aktive Unterstützung durch das gehobene Management	„Major changes to the software Process must start at the top“ [Hum90], „Management commitment and support“ [Ste98], „Senior Management Involved in SPI“ bei [Gol95] zitiert nach [Ema+99], “Senior Management Involvement” [Hum90]:

SPI-Risiko	Maßnahme
	Das gehobene Management muß durch Vorbild, Mitwirkung, Ressourcenbereitstellung und eindeutige Signale an die Beteiligten das Verbesserungsvorhaben konsequent und nachhaltig unterstützen
Bei Prozeßverbesserungsvorhaben herrscht oft engstirniges Bereichsdenken vor.	„Encouraging communication and collaboration“ [Ste98]: Anforderungen an Rolle Prozeß-Berater
Bei Prozeßverbesserungsvorhaben werden oft Gruppen von Betroffenen übergangen.	„Technical Staff involved in SPI“ bei [Gol95] zitiert nach [Ema+99] und „Staff involvement“ [Ste98], „Ultimately, everyone must be involved“ [Hum90]: Anforderungen an Rolle Prozeß-Beteiligter und ihr Staffing; Stakeholderanalyse durchführen
Bei Prozeßverbesserungsmaßnahmen werden oft in der Organisation anerkannte Meinungsführer nicht hinreichend eingebunden.	„Change agents and opinion leaders“ [Ste98]: Anforderungen an Rolle Prozeß-Beteiligter und ihr Staffing
Bei Prozeßverbesserungsmaßnahmen werden oft die Anregungen und Wünsche der Betroffenen nicht ernstgenommen	“Attitude of respect for the views of the people in the organisation being assessed” [Hum90]: Anforderungen an Rolle Prozeß-Beteiligter und ihr Staffing
Bei Prozeßverbesserungsmaßnahmen werden oft kritische Aussagen von Beteiligten an deren Vorgesetzte weitergegeben.	“requirement of confidentiality” [Hum90]: Den Prozeß-Beteiligten Vertraulichkeit zusichern, Aussagen anonymisieren

Tabelle 10 SPI-Risikoliste mit Maßnahmen

Die *Fragen zu Einflußfaktoren aus dem Projektumfeld* zur Untersuchung des weiteren Entwicklungs-Umfeldes, in dem das mit Hilfe des EVGM ausgewählte Vorgehensmodell eingesetzt werden soll, stehen in Kapitel 2 beschrieben. In der nachfolgenden Tabelle 11 sind sie noch einmal im Überblick:

Risikobereich ***Projektthemenbereiche***	***Entwicklungs-Umfeld*** ***Fragen zu Einflußfaktoren aus dem Projektumfeld***
Domänenwissen	• EPD-StaTrans-Projektziele
Softwareentwicklung	• EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen
Projektmanagement	• EPP-Einf-DurchWirtschaftlichJuristischeVerteilung • EPP-Einf-DurchRäumlicheGeografischeVerteilung • EPP-Einf-DurchEntwicklungstiefe • EPP-Einf-DurchArtDesVertraglichenVerhältnisses • EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen • EPP-Einf-DurchOrganisatorischeRahmenbedingungen • EPP-Einf-EPP-Einf-DurchRahmenbedingungenDesZielmarktes • EPP-Einf-EPP-Einf-EPP-Einf-DurchRisikenDurchAbhängigkeiten
Menschenführung	• EPM-Einf-DurchKompatibilitätZuWertenUndUnternehmenskultur

Tabelle 11 Fragen zu Einflußfaktoren aus dem Projektumfeld

Die *Risikoliste* und die *Fragen zu Einflußfaktoren aus dem Projektumfeld* sind im EVGM vorgefertigt. Sie können durch die Aktivität *Risikomanagement betreiben* modifiziert werden.

4.6.2 Projektmerkmale

Intention

Die *Projektmerkmale* stellen den Benutzern des EVGM die belegbar häufig erfolgsbestimmenden Einflußfaktoren in Softwareentwicklungsprojekten komprimiert und syste-

matisiert zur Verfügung. Die *Projektmerkmale* dienen ihnen als gemeinsame Sprache.

Inhalte und Eigenschaften

Die *Projektmerkmale* repräsentieren die vom EVGM vorgeschlagenen Merkmale für die Modellierung eines Softwareentwicklungsprojektes in Bezug auf das Entwicklungs-Produkt und das Entwicklungs-Team.

Die *Projektmerkmale* wurden im Kapitel 2 hergeleitet. In der nachfolgenden Tabelle 12 sind sie noch einmal im Überblick:

Risikobereich ***Projektthemenbereiche***	***Entwicklungs-Team*** ***Merkmale***	***Entwicklungs-Produkt*** ***Merkmale***
Domänenwissen	• EPD-BedDeck-Fähigk-Fachlich	• EPD-Einf-DurchUmfangFunktionen • EPD-StaTrans-FachlicheAnforderungen
Softwareentwicklung	• EPSWE-BedDeck-Fähigk-SoftwareEngineering • EPSWE-BedDeck-Fähigk-Software-Management	• EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien • EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien
Projektmanagement	• EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement • EPP-Einf-DurchTeamgröße • EPP-BedDeck-Kap-Personell	• EPP-BedDeck-Kap-Projektbudget • EPP-BedDeck-Kap-KalendarischeZeit
Menschenführung	• EPM-BedDeck-MotivationImTeam • EPM-BedDeck-KommunikationImTeam • EPM-BedDeck-Führungsfähigkeiten	

Tabelle 12Projektmerkmale

Die *Projektmerkmale* sind im EVGM vorgefertigt. Sie können durch die Aktivitäten *Modell mit Merkmalen Entwicklungs-Produkt entwickeln* und *Modell mit Merkmalen Entwicklungs-Team entwickeln* modifiziert werden.

4.6.3 Modell mit Merkmalen Entwicklungs-Produkt

Intention

Dem Team dient das *Modell mit Merkmalen Entwicklungs-Produkt* als visualisiertes und übersichtliches Ausdrucksmittel für dessen projektspezifische Einschätzung und Expertise bezüglich wesentlicher Problemschwerpunkte.

Inhalte und Eigenschaften

Das *Modell mit Merkmalen Entwicklungs-Produkt* enthält die dezidierte Darstellung der erfolgsgefährdend bewerteten Projektmerkmale aus dem Risikobereich Entwicklungs-Produkt mit ihren Beeinflussungsbeziehungen und ihren projektspezifischen Bewertungen. Es ist die Dokumentation von umfassendem Problemverständnis und -durchdringung.

Das *Modell mit Merkmalen Entwicklungs-Produkt* ist das Ergebnis einer umfassenden, methodischen Analyse der Ist-Situation mit Hilfe der *Projektmerkmale* und der *Fragen zu Einflußfaktoren aus dem Projektumfeld.*

Es setzt die wichtigsten Merkmale in projektgefährdendem Zustand des betrachteten Projektes zueinander in Beziehung. Das *Modell mit Merkmalen Entwicklungs-Produkt* ermöglicht einen Gesamtüberblick über die Merkmale in projektgefährdendem Zustand

und ihre Abhängigkeiten voneinander. Dieses Arbeitsprodukt des EVGM stellt spezifisch für das Projekt umfassend und vernetzt die als wesentlich eingeschätzten Merkmale dar. Das Hauptaugenmerk liegt hierbei auf den als projektgefährdend eingeschätzten Merkmalen. Es repräsentiert einen Teil des Ist-Zustandes im Projekt.
Das *Modell mit Merkmalen Entwicklungs-Produkt* muß im Rahmen der Aktivität *Modell mit Merkmalen Entwicklungs-Produkt entwickeln* angefertigt werden. Es wird durch die Aktivität *Modell mit Merkmalen Entwicklungs-Team entwickeln* weiterentwickelt.

4.6.4 Modell mit Merkmalen Entwicklungs-Team

Intention

Das EVGM-Team erhält durch das *Modell mit Merkmalen Entwicklungs-Team* ein Ausdrucksmittel für die erfolgsbestimmenden personellen Einflußfaktoren im Softwareprojekt und eine vervollständigte, systematische Darstellung des Ist-Zustandes.

Inhalte und Eigenschaften

Das *Modell mit Merkmalen Entwicklungs-Team* ist das Ergebnis einer umfassenden, methodischen Analyse des Ist-Zustandes. Es ergänzt das *Modell mit Merkmalen Entwicklungs-Produkt* um die erfolgsbestimmenden Merkmale der am Projekt beteiligten Personen.
Hierbei können gezielt projektgefährdende Zustände von Merkmalen durch menschliche Eigenschaften verbessert werden. Es zeigt insgesamt die verbessernden Auswirkungen der Merkmale in projektförderlichem Zustand auf den Zustand der Merkmale in projektgefährdendem Zustand.
Das *Modell mit Merkmalen Entwicklungs-Team* muß im Rahmen der Aktivität *Modell mit Merkmalen Entwicklungs-Team entwickeln* angefertigt werden. Es wird durch die Aktivität *Modell mit Anforderungen an ideales Vorgehensmodell entwickeln* weiterentwickelt.

4.6.5 Vorgehensmodellmerkmale

Intention

Die *Vorgehensmodellmerkmale* stellen den Benutzern des EVGM die komprimierten Eigenschaften von Vorgehensmodellen zur Verfügung, welche die häufigsten Problemschwerpunkte in Softwareentwicklungsprojekten adressieren. Die *Vorgehensmodellmerkmale* unterstützten die Anwender des EVGM, systematisch die Nutzenpotentiale von Vorgehensmodellen bedarfsgerecht einzusetzen. Das EVGM-Team erhält durch die *Vorgehensmodellmerkmale* eine Basis für den strukturierten, methodischen Vergleich von Vorgehensmodellen. Das erleichtert die Nachvollziehbarkeit des Entscheidungsprozesses und der Auswahl eines Vorgehensmodells.

Inhalte und Eigenschaften

Die *Vorgehensmodellmerkmale* repräsentieren die vom EVGM vorgeschlagenen Merkmale für die Modellierung der Eigenschaften von Vorgehensmodellen. Sie decken den Risikobereich Entwicklungs-Prozeß ab.
Die *Vorgehensmodellmerkmale* wurden im Kapitel 3 hergeleitet. In der nachfolgenden Tabelle 13 sind sie noch einmal im Überblick dargestellt. Dabei sind sie den Projektmerkmalen, welche sie adressieren, bereits zugeordnet: Jedes *Vorgehensmodellmerkmal* steht *kursiv* jeweils unter dem Projektmerkmal, welches es adressiert.

Risikobereich Projektthemen bereiche	***Entwicklungs-Produkt Merkmale***	***Entwicklungs-Team Merkmale***
Domänenwissen *Requirements Engineering*	• EPD-Einf-DurchUmfangFunktionen *EV-UnterstProjTheBer-Requirements-Engineering* • EPD-StaTrans-FachlicheAnforderungen *EV-UnterstProjTheBer-Requirements-Engineering* *EV-UnterstProjTheBer-SoftwareEngineering* *EV-UnterstProjTheBer-SoftwareManagement* *EV-Flexib-DurchIterativität-UndInkrementalität*	• EPD-BedDeck-Fähigk-Fachlich *EV-UnterstProjTheBer-RequirementsEngineering*
Softwareentwicklung *Software Engineering* *Software Management*	• EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien *EV-UnterstProjTheBer-Requirements-Engineering* *EV-UnterstProjTheBer-SoftwareEngineering* • EPSWE-StaTrans-TechnischeAnforderungenUndQualitätskriterien *EV-UnterstProjTheBer-RequirementsEngineering* *EV-UnterstProjTheBer-SoftwareEngineering* *EV-UnterstProjTheBer-SoftwareManagement*	• EPSWE-BedDeck-Fähigk-SoftwareEngineering *EV-UnterstProjTheBerRequirements-Engineering* *EV-UnterstProjTheBer-SoftwareEngineering* • EPSWE-BedDeck-Fähigk-SoftwareManagement *EV-UnterstProjTheBer-SoftwareManagement*
Projektmanagement *Projektmanagement*	• EPP-BedDeck-Kap-Projektbudget *EV-UnterstProjTheBer-ProjektManagement* • EPP-BedDeck-Kap-KalendarischeZeit *EV-UnterstProjTheBer-ProjektManagement*	• EPP-BedDeck-Fähigk-OrganisatorischesUndKaufmännischesProjektmanagement *EV-UnterstProjTheBer-ProjektManagement* • EPP-Einf-DurchTeamgröße *EV-UnterstProjTheBer-ProjektManagement* • EPP-BedDeck-Kap-Personell *EV-UnterstProjTheBer-ProjektManagement*
Menschenführung *Menschenführung*		• EPM-BedDeck-MotivationImTeam *EV-UnterstProjTheBer-Menschenführung* • EPM-BedDeck-KommunikationImTeam *EV-UnterstProjTheBer-Menschenführung* • EPM-BedDeck-Führungsfähigkeiten *EV-UnterstProjTheBer-Menschenführung*

Tabelle 13 Projekt- und Vorgehensmodellmerkmale

Die *Vorgehensmodellmerkmale* sind im EVGM vorgefertigt. Sie können durch die Aktivität *Lösungsmodell mit Eigenschaften ideales Vorgehensmodell entwickeln* modifiziert werden.

4.6.6 Modell mit Anforderungen an ideales Vorgehensmodell

Intention

Das EVGM-Team erhält durch das *Modell mit Anforderungen an ideales Vorgehensmo-*

dell ein Ausdrucksmittel für die Anforderungen an ein projektspezifisch optimales Vorgehensmodell. Es wird durch die lösungsneutrale Darstellung nicht durch die Beschränkungen reeller Vorgehensmodellen in seinem Denken über einen Idealzustand eingeschränkt.

Inhalte und Eigenschaften

Das *Modell mit Anforderungen an ideales Vorgehensmodell* ist eine Weiterentwicklung des *Modells mit Merkmalen Entwicklungs-Team.* In ihm werden die Projektmerkmale in projektgefährdendem Zustand aus dem *Modell mit Merkmalen Entwicklungs-Team* mit Vorgehensmodellmerkmalen adressiert, welche die Anforderungen an ein projektspezifisch ideales Vorgehensmodell formulieren. Das *Modell mit Anforderungen an ideales Vorgehensmodell* zeigt insgesamt die verbessernden Auswirkungen der Vorgehensmodellmerkmale auf den Zustand der Projektmerkmale in projektgefährdendem Zustand. Das *Modell mit Anforderungen an ideales Vorgehensmodell* repräsentiert einen umfassenden, methodischen, aber noch lösungsneutralen Lösungsansatz. Dieser besteht aus Merkmalen, die Vorgehensmodelle charakterisieren, ist jedoch von existierenden Vorgehensmodellen noch unabhängig. Es enthält Anforderungen an die Werte von Vorgehensmodellmerkmalen und gibt so Anregungen für Ergänzungen des später ausgewählten konkreten Vorgehensmodells.

Das *Modell mit Anforderungen an ideales Vorgehensmodell* muß im Rahmen der Aktivität *Modell mit Anforderungen an ideales Vorgehensmodell entwickeln* angefertigt werden. Es wird durch die Aktivität *Vorgehensmodellspezifisches Lösungsmodell entwickeln* weiterentwickelt.

4.6.7 Vorgehensmodellspezifisches Lösungsmodell

Intention

Das EVGM-Team erhält durch das V*orgehensmodellspezifische Lösungsmodell* ein Ausdrucksmittel für den abgeschätzten Gesamtzustand, der durch den Einsatz von einem der im EVGM berücksichtigten konkreten Vorgehensmodelle entsteht. Es dient dem EVGM-Team als Basis für den Vergleich von entstehenden Zielzuständen durch den Einsatz verschiedener Vorgehensmodelle und damit für die Eignung der Vorgehensmodelle. Damit ist das V*orgehensmodellspezifische Lösungsmodell* Grundlage für die nachvollziehbare Entscheidung bezüglich eines Vorgehensmodells.

Inhalte und Eigenschaften

Es simuliert modellhaft die projektförderlichen und projektgefährdenden Beeinflussungen von Projektmerkmalen durch den Einsatzes von jeweils einem Vorgehensmodell. *Vorgehensmodellspezifische Lösungsmodelle* zeigen jeweils spezifisch für ein Vorgehensmodell die durch dessen Einsatz entstehende Situation im Projekt und bilden somit Varianten des *Modells mit Anforderungen an ideales Vorgehensmodell*. Sie zeigen insgesamt die verbessernden Auswirkungen der Merkmale in projektförderlichem Zustand auf den Zustand der projektgefährdenden Merkmale. *Vorgehensmodellspezifische Lösungsmodelle* repräsentieren einen realistischen, konkreten und umfassenden Entwurf einer durch den Einsatz eines Vorgehensmodells geprägten Soll-Situation. Sie stellen daher nicht notwendigerweise einen Idealzustand dar, da sie auch projektgefährdende Bestandteile eines Vorgehensmodells übernehmen müssen.

Die V*orgehensmodellspezifischen Lösungsmodelle* müssen im Rahmen der Aktivität V*orgehensmodellspezifisches Lösungsmodell entwickeln* angefertigt werden. Es wird durch die Aktivität *Entscheidung für Vorgehensmodell treffen* weiterverwendet.

Dieses Kapitel stellte die Arbeitsprodukte vor, die im EVGM definiert sind.
Im nächsten Kapitel werden die Aktivitäten definiert, mit deren Hilfe diese Arbeitsprodukte erstellt und benutzt werden.

4.7 Aktivitäten im EVGM

Die Aktivitäten beschreiben die Tätigkeiten, die von den am EVGM Verfahren Beteiligten durchgeführt werden sollen, um sie bei der richtigen Durchführung der Arbeitsabläufe zu unterstützen. Es werden Schritt-für-Schritt-Anweisungen gegeben.
Jede Aktivität im EVGM ist verbindlich durchzuführen.

Die Darstellung in diesem Kapitel gibt eine **geeignete Abfolge** vor, über die iteriert wird. Im Bedarfsfall kann von der Abfolge abgewichen werden. Die folgenden Aktivitäten werden beschrieben:

- Risikomanagement betreiben
- Modell mit Merkmalen Entwicklungs-Produkt entwickeln
- Modell mit Merkmalen Entwicklungs-Team entwickeln
- Modell mit Anforderungen an ideales Vorgehensmodell entwickeln
- Vorgehensmodellspezifische Lösungsmodelle entwickeln
- Entscheidung für ein Vorgehensmodell treffen

Die Aktivitäten des EVGM werden in der Abbildung 30 im Überblick veranschaulicht.

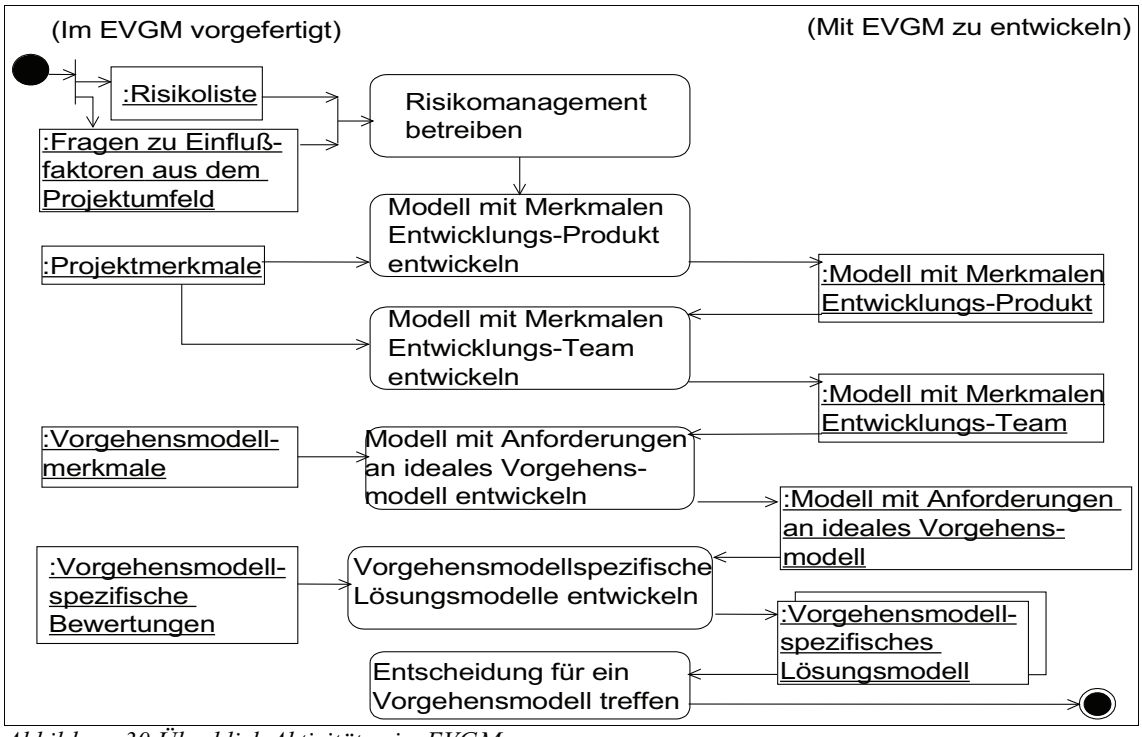

Abbildung 30 Überblick Aktivitäten im EVGM

Die vom Autor der vorliegenden Arbeit für das EVGM entworfene Vorgehensweise und die in [Wal04] generischer skizzierte Vorgehensweise für das Risikomanagement „Identifikation von Risiken"; „Analyse, Bewertung, Filterung und Priorisierung von Risiken" und „Risikoplanung" (im Sinne von Bemaßnahmung von Risiken) weisen auf abstrakter Ebene grundsätzliche Ähnlichkeiten auf, obwohl sie unabhängig voneinander entstan-

den sind. Dies kann als zusätzliches Indiz für die Sinnhaftigkeit der in der vorliegenden Arbeit gewählten Vorgehensweise gewertet werden.
Die jeweils in einem Abschnitt beschriebene Aktivität und die mit ihr in Verbindung stehenden Arbeitsprodukte sind in der zugehörigen Abbildung jeweils kursiv hervorgehoben. Am Ende jeder Aktivitätsbeschreibung sind Prüffragen angefügt, die helfen, das Ergebnis zu evaluieren.

4.7.1 Risikomanagement betreiben

Intention

Die Aktivität *Risikomanagement betreiben* benutzt die Arbeitsprodukte 4.6.1 Risikoliste, Fragen zu Einflußfaktoren aus dem Projektumfeld (Intention siehe dort).
Die Aktivität *Risikomanagement betreiben*, ihre Eingabe- und ihre Ausgabe-Arbeitsprodukte (kursiv hervorgehoben) zeigt die folgende Abbildung 31.

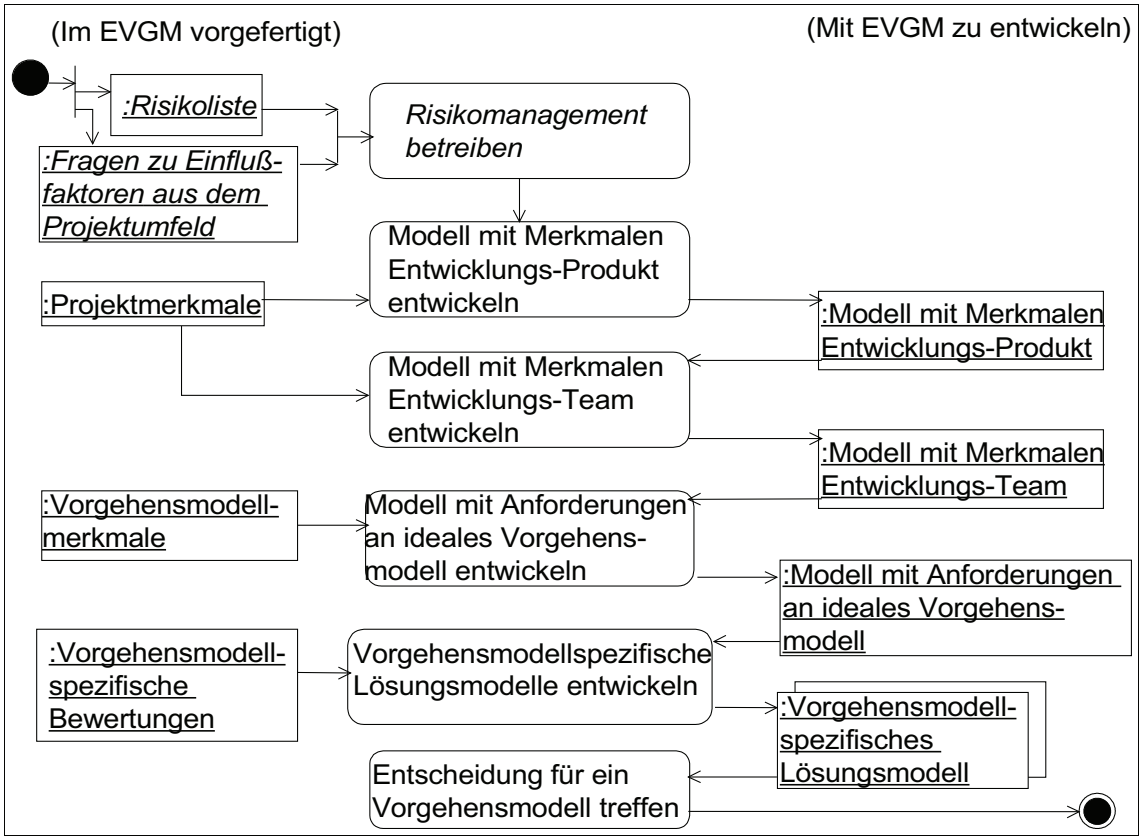

Abbildung 31 Aktivität Risikomanagement betreiben im Zusammenhang

Inhalt

Rollen besetzen. Risiken für das Prozeßverbesserungsvorhaben und das Softwareentwicklungsvorhaben werden auch durch die Anforderungen an die Rolle des Prozeß-Beraters beziehungsweise durch die Anforderungen an die Rolle Prozeß-Beteiligter und deren Besetzung adressiert (siehe Kapitel 4.5 Rollen im EVGM).
Softwareprozeßverbesserungs- und Veränderungsmanagementrisiken behandeln. Im Rahmen der Aktivität *Risikomanagement betreiben* werden Risiken für das EVGM und das damit verbundene Vorhaben zur Verbesserung des Softwareprozesses identifiziert. Sie hilft, das Umfeld zu untersuchen, in dem das EVGM angewendet wird. Die Risiken bezüglich der Softwareprozeßverbesserung und des Veränderungsmanage-

ments aus dem Umfeld des EVGM werden dokumentiert und bewertet. Für die Risiken werden Gegenmaßnahmen beschrieben. Dies geschieht mit Hilfe der *Risikoliste,* für deren Einträge jeweils untersucht wird, ob sie für das aktuelle Projekt relevant sind. Falls dies der Fall ist, wird untersucht, ob die in der Risikoliste vorgeschlagene Gegenmaßnahme durch das EVGM-Team als zielführend beurteilt wird und versucht diese zu initiieren.

Softwareentwicklungsrisiken behandeln. Im Rahmen der Aktivität *Risikomanagement betreiben* werden auch die Risiken für das Softwareentwicklungsprojekt analysiert, für das mit dem EVGM ein Vorgehensmodell ausgewählt wird. Diese werden durch die Fragen zu Einflußfaktoren aus dem Projektumfeld aus Kapitel 2 untersucht, welche über Beeinflussungsbeziehungen Hinweise auf wesentliche Merkmale geben. Die Merkmale werden dann in der Modellierung berücksichtigt.

Die Aktivität *Risikomanagement betreiben* wird bei der Durchführung des EVGM als erstes ausgeführt und im Bedarfsfall iterativ wiederholt.

Prüffragen

- Welche über die Vorschläge im EVGM hinausgehenden Risiken und Gegenmaßnahmen gibt es im Projekt, wie wesentlich sind diese?
- Welche weiteren über die Vorschläge im EVGM hinausgehende Risikoaspekte sind denkbar?

4.7.2 Modell mit Merkmalen Entwicklungs-Produkt entwickeln

Intention

Die Aktivität *Modell mit Merkmalen Entwicklungs-Produkt entwickeln* beschreibt, wie das Arbeitsprodukt 4.6.3 „Modell mit Merkmalen Entwicklungs-Produkt“ aus den Arbeitsprodukten 4.6.1 „Risikoliste, Fragen zu Einflußfaktoren aus dem Projektumfeld“ und 4.6.2 „Projektmerkmale“ abgeleitet und entwickelt wird.

Die Aktivität *Modell mit Merkmalen Entwicklungs-Produkt entwickeln*, ihre Eingabe- und ihre Ausgabe-Arbeitsprodukte (kursiv hervorgehoben) zeigt die folgende Abbildung 32.

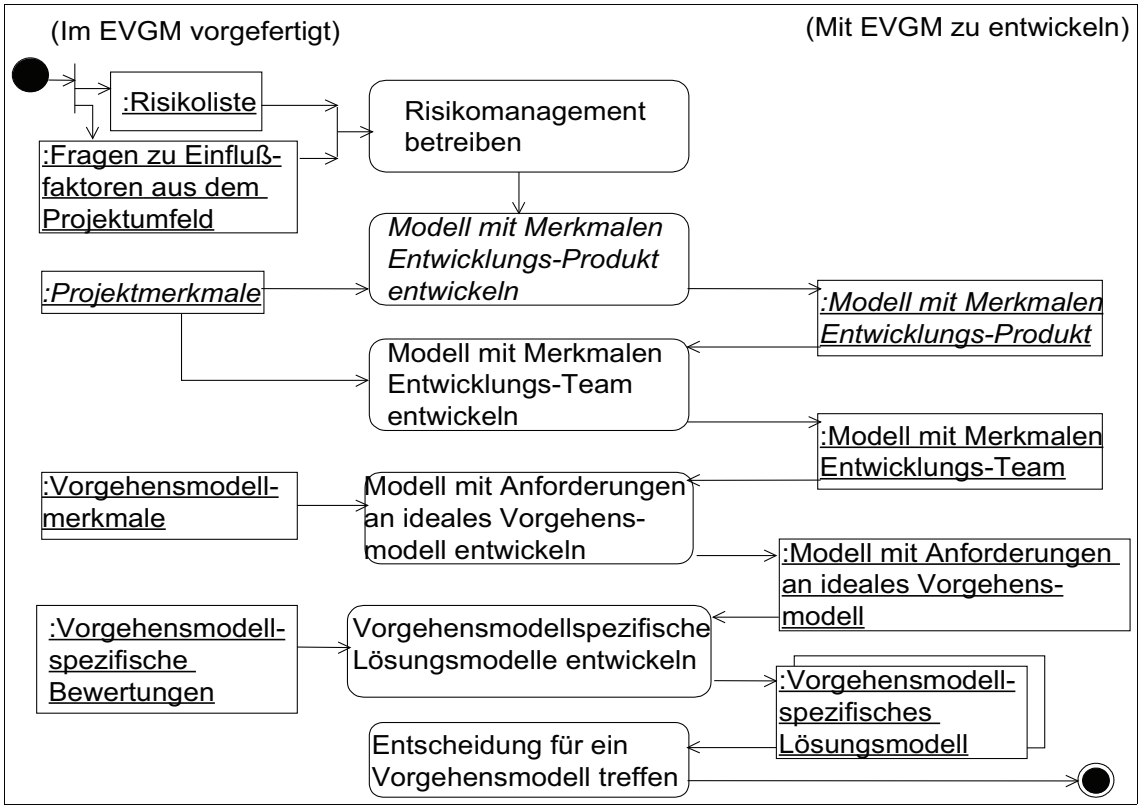

Abbildung 32 Aktivität Modell mit Merkmalen Entwicklungs-Produkt entwickeln im Zusammenhang

Inhalt

Erfolgsbestimmende Projektmerkmale erheben. Das Projekt wird hinsichtlich seiner wesentlichen erfolgsbestimmenden *Projektmerkmale* aus dem Risikobereich Entwicklungs-Produkt durch das EVGM-Team analysiert. Hierbei werden speziell die als projektgefährdend beurteilte *Projektmerkmale* herausgegriffen. Wesentliche *Projektmerkmale*, welche nicht in einem projektgefährdenden Zustand aber erfolgsbestimmend sind, werden ebenfalls berücksichtigt. Hinweise auf projektspezifisch wesentliche *Projektmerkmale* haben die Fragen zu Einflußfaktoren aus dem Projektumfeld aus Kapitel 2 gegeben (siehe letzter Abschnitt).

Erfolgsbestimmende Projektmerkmale auswählen. Erfolgsbestimmende *Projektmerkmale* werden vom EVGM vorgeschlagen (siehe Kapitel 2) und können vom EVGM-Team ausgewählt werden. Das EVGM-Team kann im Bedarfsfall eigene *Projektmerkmale* hinzufügen.

Projektmerkmale in Ursache-Wirkungs-Diagramm modellieren. Die Projektmerkmale werden im *Modell mit Merkmalen Entwicklungs-Produkt* modelliert.

Beeinflussungsbeziehungen zwischen Merkmalen erheben. Das EVGM-Team kann durch eigene Expertise Beeinflussungsbeziehungen ergänzen.

Beeinflussungsbeziehungen zwischen Merkmalen auswählen. Beeinflussungsbeziehungen werden durch das EVGM in Kapitel 2 vorgeschlagen und können dort ausgewählt werden.

Beeinflussungsbeziehungen in Ursache-Wirkungs-Diagramm modellieren. Dann werden die *Projektmerkmale* durch ihre Beeinflussungsbeziehungen zueinander in Wechselwirkung gesetzt. Hier dienen die Beschreibungen der Beeinflussungsbeziehungen aus Kapitel 2 als Leitlinie. Die Beeinflussungsbeziehungen werden im *Modell mit Merkmalen Entwicklungs-Produkt* modelliert.

Merkmale und Beeinflussungsbeziehungen in Modell bewerten. Die *Projektmerkmale* werden hinsichtlich ihres Zustands im konkreten Projekt durch die Einschätzung des EVGM-Teams mit Werten von „--“ für einen sehr projektgefährdenden Zustand bis „++“ für einen sehr projektförderlichen Zustand bewertet. Die Beeinflussungsbeziehungen werden mit „1/≈≈“ bis „≈≈“ bewertet. Hinweise auf Bewertungen von *Projektmerkmalen* und Beeinflussungsbeziehungen geben die Fragen zu Einflußfaktoren aus dem Projektumfeld aus Kapitel 2.

Ursache-Wirkungs-Diagramm reduzieren. Der erste Entwurf des Ursache-Wirkungs-Diagramms, der mit Hilfe der Aktivität *Modell mit Merkmalen Entwicklungsprodukt entwickeln* modelliert wurde, wird nun wieder reduziert um das Diagramm kompakt und beherrschbar zu halten. Hierbei werden Projektmerkmale und Beeinflussungsbeziehungen entfernt, die durch das EVGM-Team als weniger wichtig beurteilt werden. Dies ist bei Projektmerkmalen an einer vergleichsweise geringen Vernetzung und an einem unkritischen Zustand erkennbar. Bei Beeinflussungsbeziehungen ist eine geringe Einflußstärke ein Hinweis darauf das sie möglicherweise entfernt werden können. Es werden keine Merkmale entfernt, die erst durch eine Beeinflussung durch ein Projektmerkmal mit projektförderlichem Zustand einen unkritischen Zustand erreicht haben.

Prüffragen

Folgende Fragen unterstützen die Modellierung des *Modells mit Merkmalen Entwicklungs-Produkts*:

- Was sind die wesentlichen Merkmale in projektgefährdendem Zustand im gegebenen Projekt aus dem Risikobereich Entwicklungs-Produkt?
- Wie hängen die Projektmerkmale zusammen, wie beeinflussen sie sich gegenseitig?
- In welchem Ist-Zustand sind die Projektmerkmale und deren Beeinflussungsbeziehungen im Ursache-Wirkungs-Diagramm?
- Was wirkt der Verbesserung von deren Zuständen entgegen?
- Wie beeinflussbar sind die Projektmerkmale und ihre Zustände?
- Welche Auswirkungen haben die Veränderungen von Projektmerkmalen auf den Zustand anderer Projektmerkmale?
- Wurde das *Modell mit Merkmalen Entwicklungs-Produkt* ausreichend hinterfragt?
- Besteht ein ausgewogenes umfassendes Bild der Merkmale des Entwicklungs-Produktes?
- Welche über die Vorschläge im EVGM hinausgehenden Merkmale und Beeinflussungsbeziehungen gibt es im Projekt, wie wesentlich sind diese?

4.7.3 Modell mit Merkmalen Entwicklungs-Team entwickeln

Intention

Die Aktivität *Modell mit Merkmalen Entwicklungs-Team entwickeln* beschreibt, wie das Arbeitsprodukt 4.6.4 „Modell mit Merkmalen Entwicklungs-Team“ aus den Arbeitsprodukten 4.6.1 „Risikoliste, Fragen zu Einflußfaktoren aus dem Projektumfeld“, 4.6.2 „Projektmerkmale“ und 4.6.3 „Modell mit Merkmalen Entwicklungs-Produkt“ entwickelt wird.

Die Aktivität *Modell mit Merkmalen Entwicklungs-Team entwickeln*, ihre Eingabe- und ihre Ausgabe-Arbeitsprodukte (kursiv hervorgehoben) zeigt die folgende Abbildung 33.

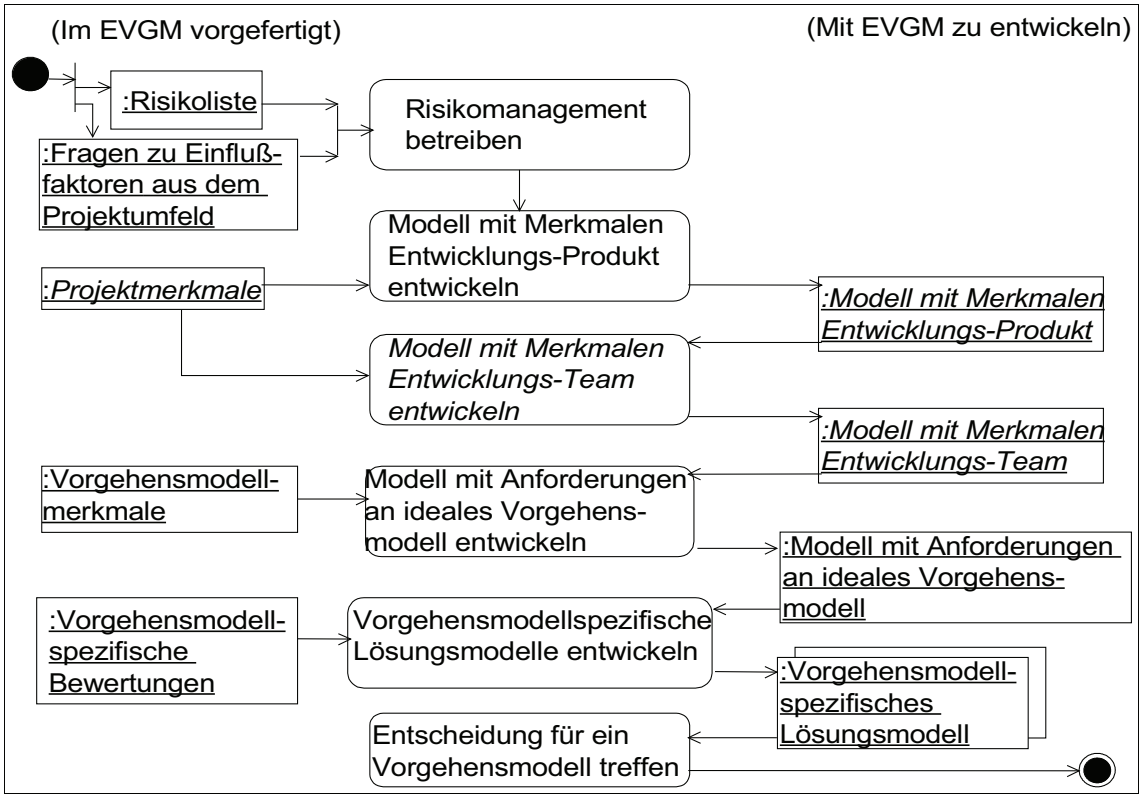

Abbildung 33 Aktivität Modell mit Merkmalen Entwicklungs-Team entwickeln im Zusammenhang

Inhalt

Erfolgsbestimmende Projektmerkmale erheben. Das Projekt wird hinsichtlich seiner wesentlichen *Projektmerkmale* bezüglich des Risikobereiches Entwicklungs-Team durch das EVGM-Team analysiert, es beurteilt potentiell seine eigenen Stärken und Schwächen. Es verspricht eine realistischere und fundiertere Einschätzung, wenn anstatt des Managements die Projektbeteiligten diese Einschätzung für sich selbst durchführen, weil sie eine unmittelbare Kenntnis der Materie besitzen. Hierbei werden die wesentlichen, als projektförderlich und projekthinderlich beurteilten *Projektmerkmale* untersucht und berücksichtigt. *Projektmerkmale* können auch vom EVGM-Team ergänzt werden. Dies soll aus möglichst vielen unterschiedlichen Sichten der einzelnen Teammitglieder, ihrer Software-Engineering-Rollen und -Hintergründe geschehen.

Erfolgsbestimmende Projektmerkmale auswählen. *Projektmerkmale*, werden vom EVGM vorgeschlagen und können vom EVGM-Team ausgewählt werden. Als Leitlinie dienen *Projektmerkmale* aus Kapitel 2. Hinweise auf personelle *Projektmerkmale* geben die *Fragen zu Einflußfaktoren aus dem Projektumfeld* aus Kapitel 2.

Projektmerkmale in Ursache-Wirkungs-Diagramm modellieren. Die personellen *Projektmerkmale* werden im *Modell mit Merkmalen Entwicklungs-Team* modelliert.

Beeinflussungsbeziehungen zwischen Projektmerkmalen erheben. Das EVGM-Team kann über die im EVGM vorgeschlagenen, eigene Beeinflussungsbeziehungen zwischen den modellierten *Projektmerkmalen* ergänzen.

Beeinflussungsbeziehungen zwischen Projektmerkmalen auswählen. Das EVGM schlägt in Kapitel 2 Beeinflussungsbeziehungen zwischen den *Projektmerkmalen* vor, aus denen das EVGM-Team auswählen kann.

Beeinflussungsbeziehungen in Ursache-Wirkungs-Diagramm modellieren. Wesentliche Beeinflussungsbeziehungen zwischen *Projektmerkmalen* werden dem *Mo-*

dell mit Merkmalen Entwicklungs-Team hinzugefügt.

Projektmerkmale und Beeinflussungsbeziehungen in Ursache-Wirkungs-Diagram bewerten. Die personellen *Projektmerkmale* aus dem Risikobereich Entwicklungs-Team werden hinsichtlich ihres Zustands im konkreten Projekt bewertet. Hierbei gehen die Werte von „--“ für einen sehr projektgefährdenden Zustand“ bis „++“ für einen sehr projektförderlichen Zustand. Hinweise geben hier die *Fragen zu Einflußfaktoren aus dem Projektumfeld* aus Kapitel 2. Die Beeinflussungsbeziehungen werden mit „1/≈≈“ bis „≈≈“ bewertet.

Auswirkungen modellieren. Das Einfügen der neuen *Projektmerkmale* hat zum Ziel, den Zustand bestehender projektgefährdend bewerteter *Projektmerkmale* zu verbessern. Diese Zustandsveränderungen werden nun entlang der Beeinflussungsbeziehung und gemäß der Einschätzung des EVGM-Teams modelliert.

Ursache-Wirkungs-Diagram reduzieren. Nach der Erweiterung des Ursache-Wirkungs-Diagramms durch die Aktivität *Modell mit Merkmalen Entwicklungs-Team entwickeln* wird es wieder reduziert, um es kompakt und beherrschbar zu halten. Hierbei werden *Projektmerkmale* und Beeinflussungsbeziehungen entfernt, die durch das EVGM-Team als nicht wesentlich beurteilt werden. Hinweis hierauf sind vergleichsweise geringe Vernetzung und unkritischer Zustand. Es werden keine *Projektmerkmale* entfernt, die erst durch eine Beeinflussung durch ein Projektmerkmal in projektförderlichem Zustand einen unkritischen Zustand erreicht haben.

Prüffragen

Folgende Fragen unterstützen die Modellierung des *Modells mit Merkmalen Entwicklungs-Team*:

- Was sind die wesentlichen projektförderlichen Merkmale des Entwicklungs-Teams im gegebenen Projekt?
- Wie hängen die personellen Merkmale zusammen, wie bedingen und beeinflussen sie sich gegenseitig?
- In welchem Ist-Zustand sind die personellen Merkmale und deren Beeinflussungsbeziehungen?
- Was können die personellen Merkmale in projektförderlichem Zustand zum Erreichen des Soll-Zustandes des Projektes beitragen?
- Wie kann eine Zustandsverbesserung durch personelle Merkmale erreicht und erhalten werden?
- Wie stehen die personelle Merkmale in Beeinflussungsbeziehung zu den Merkmalen in projektgefährdendem Zustand?
- Wie sollen die personelle Merkmale in Beeinflussungsbeziehung stehen, wie können sie die Merkmale in projektgefährdendem Zustand positiv beeinflussen?
- Was wirkt der Verbesserung des Zustandes projektgefährdender Merkmale entgegen? Wie kann das durch personelle Merkmale kontrolliert oder kompensiert werden?
- Wie beeinflussbar sind die Projektmerkmale und ihr Zustand?
- Paradoxiemethode: Was müßte man tun, um das Softwareprojekt sicher zum Scheitern zu bringen? Wie können solche Maßnahmen invertiert werden?
- Welche Auswirkungen haben Zustandsveränderungen projektgefährdender Merkmale auf den Zustand anderer Merkmale?
- Wurde das *Modell mit Merkmalen Entwicklungs-Team* ausreichend hinterfragt?
- Besteht ein ausgewogenes umfassendes Bild der Merkmale durch das Entwicklungs-

Produkt und das Entwicklungs-Team?

- Welche über die Vorschläge im EVGM hinausgehenden Projektmerkmale und Beeinflussungsbeziehungen gibt es im Projekt, wie wesentlich sind diese?

4.7.4 Modell mit Anforderungen an ideales Vorgehensmodell entwickeln

Intention

Die Aktivität *Modell mit Anforderungen an ideales Vorgehensmodell entwickeln* beschreibt, wie das Arbeitsprodukt 4.6.6 „Modell mit Anforderungen an ideales Vorgehensmodell“ aus den Arbeitsprodukten 4.6.4 „Modell mit Merkmalen Entwicklungs-Team“ und 4.6.5 „Vorgehensmodellmerkmale“ entwickelt wird. Hierbei wird die „Tabelle 4 Überblick über Beeinflussung Projektmerkmale durch Vorgehensmodell-Merkmale“ aus Kapitel 3.2.6 „Zuordnung Vorgehensmodellmerkmale zu Projektmerkmalen“ auf Seite 114 benutzt.
Die Aktivität *Modell mit Anforderungen an ideales Vorgehensmodell entwickeln*, ihre Eingabe- und ihre Ausgabe-Arbeitsprodukte (kursiv hervorgehoben) zeigt die folgende Abbildung 34.

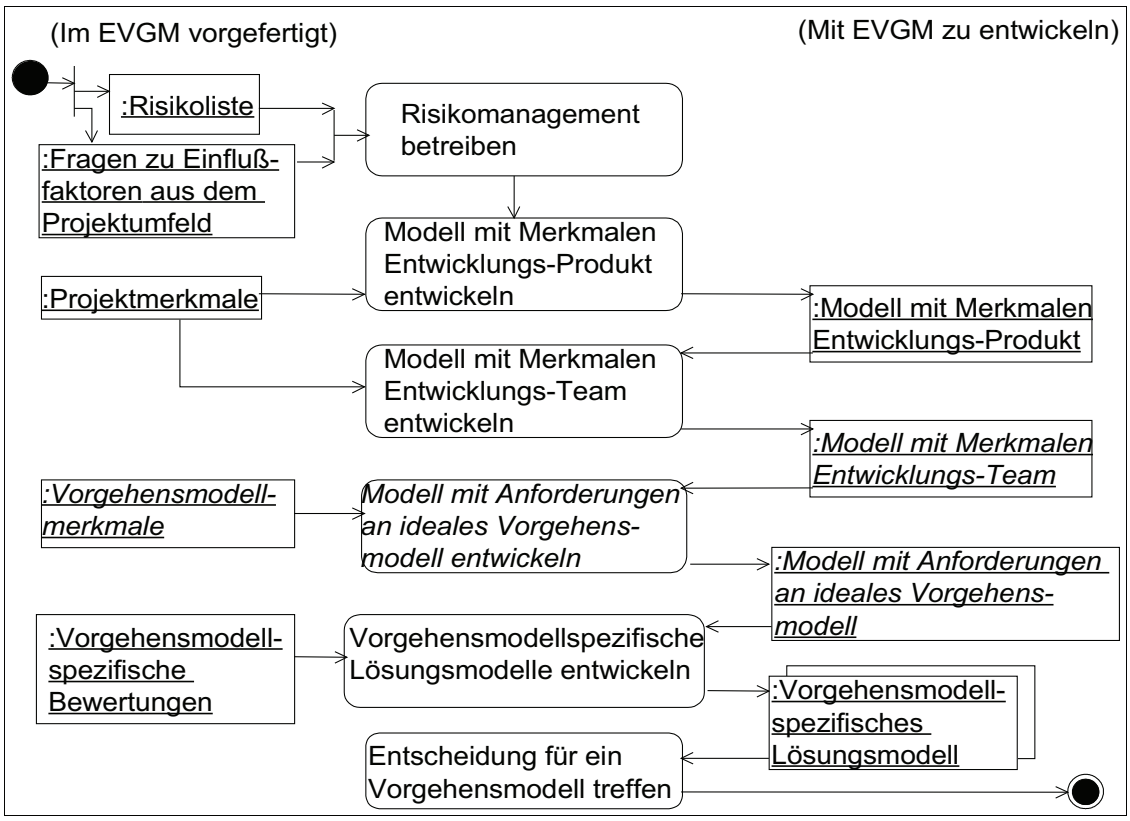

Abbildung 34 Aktivität Modell mit Merkmalen ideales Vorgehensmodell entwickeln im Zusammenhang

Inhalt

Vorgehensmodellmerkmale erheben. Das Projekt ist hinsichtlich seiner wesentlichen *Projektmerkmale* in projektgefährdendem Zustand aus den Risikobereichen Entwicklungs-Produkt und Entwicklungs-Team durch das EVGM-Team analysiert und modelliert. Hierbei wurden wesentliche, als projektgefährdend und projektförderlich beurteilte *Projektmerkmale* berücksichtigt. Jetzt werden *Vorgehensmodellmerkmale* gesucht, welche *Projektmerkmale* in projektgefährdendem Zustand positiv beeinflussen.
Vorgehensmodellmerkmale auswählen. Das EVGM schlägt *Vorgehensmodell-*

merkmale vor, welche durch Beeinflussungsbeziehungen die kritisch bewerteten *Projektmerkmale* positiv beeinflussen. Diese können vom EVGM-Team ausgewählt werden. Als Leitlinie dienen hierbei die Beeinflussungsbeziehungen zwischen *Projektmerkmalen* und *Vorgehensmodellmerkmalen* (siehe Kapitel 3).

Vorgehensmodellmerkmale in Ursache-Wirkungs-Diagramm modellieren. Die *Vorgehensmodellmerkmale* werden in das *Modell mit Anforderungen an ideales Vorgehensmodell* eingefügt.

Beeinflussungsbeziehungen zwischen Merkmalen erheben. Das EVGM-Team kann aus seiner Expertise eigene Beeinflussungsbeziehungen einbringen.

Beeinflussungsbeziehungen zwischen Merkmalen auswählen. Beeinflussungsbeziehungen zwischen Vorgehensmodell- und Projektmerkmalen werden vom EVGM vorgeschlagen und können in Kapitel 3 ausgewählt werden.

Beeinflussungsbeziehungen in Ursache-Wirkungs-Diagramm modellieren. Beeinflussungsbeziehungen zwischen Vorgehensmodell- und Projektmerkmalen werden in das *Modell mit Anforderungen an ideales Vorgehensmodell* eingefügt.

Merkmale und Beeinflussungsbeziehungen in Ursache-Wirkungs-Diagramm bewerten. Die Merkmale und ihre Beeinflussungsbeziehungen werden hinsichtlich ihres Zustands im konkreten Projektes bewertet. Hierbei gehen die Werte von „--“ für einen sehr projektgefährdenden Zustand“ bis „++“ für einen sehr projektförderlichen Zustand. Die Beeinflussungsbeziehungen werden mit „1/≈≈“ bis „≈≈“ bewertet.

Auswirkungen modellieren. Das Einfügen der *Vorgehensmodellmerkmale* hat zum Ziel, den Zustand von projektgefährdenden *Projektmerkmalen* zu verbessern. Diese Zustandsveränderungen werden nun entlang der Beeinflussungsbeziehung und gemäß der Einschätzung des EVGM-Teams modelliert.

Ursache-Wirkungs-Diagramm reduzieren. Nach der Erweiterung des Ursache-Wirkungs-Diagramms durch die Aktivität *Modell mit Anforderungen an ideales Vorgehensmodell entwickeln* wird das ergänzte Modell wieder reduziert um es kompakt und beherrschbar zu halten. Hierbei werden Merkmale entfernt, die durch das EVGM-Team als nicht wesentlich beurteilt werden. Hinweise hierauf sind eine vergleichsweise geringe Vernetzung und ein unkritischer Zustand. Es werden keine Merkmale entfernt, die erst durch eine positive Beeinflussung durch ein Merkmal in projektförderlichem Zustand einen unkritischen Zustand erreicht haben.

Prüffragen

Folgende Fragen unterstützen die Modellierung des *Modells mit Anforderungen an ideales Vorgehensmodell*:

- Was sind die wesentlichen projektförderlichen Merkmale von Vorgehensmodellen im gegebenen Projekt?
- Wie beeinflussen diese *Vorgehensmodellmerkmale* die *Projektmerkmale*?
- Welchen Zustand erhalten die *Projektmerkmale* durch die Beeinflussungen der *Vorgehensmodellmerkmale*?
- Was können die *Vorgehensmodellmerkmale* zum Erreichen des Soll-Zustandes des Projektes beitragen?
- Wie kann der Soll-Zustand durch die *Vorgehensmodellmerkmale* erreicht und erhalten werden?
- Wie stehen die *Vorgehensmodellmerkmale* in Beeinflussungsbeziehung zu den *Projektmerkmalen* in projektgefährdendem Zustand?
- Wie sollen die *Vorgehensmodellmerkmale* in Beeinflussungsbeziehung stehen, wie können sie die *Projektmerkmale* in projektgefährdendem Zustand positiv beeinflus-

sen?
- Was wirkt der Verbesserung des Zustandes kritischer *Projektmerkmale* entgegen? Wie kann das durch *Vorgehensmodellmerkmale* kontrolliert oder positiv beeinflusst werden?
- Wie beeinflußbar sind die Projektmerkmale und ihre Zustände?
- Paradoxiemethode: Was müßte man tun, um das Softwareprojekt sicher zum Scheitern zu bringen? Wie können diese Maßnahmen invertiert werden?
- Welche Auswirkungen haben ihre Veränderung auf den Zustand anderer Merkmale?
- Wurde das *Modell mit Anforderungen an ideales Vorgehensmodell* ausreichend hinterfragt?
- Besteht ein ausgewogenes umfassendes Bild von den Anforderungen an ein ideal geeignetes Vorgehensmodell für das betrachtete Projekt?
- Welche über die Vorschläge im EVGM hinausgehenden Merkmale und Beeinflussungsbeziehungen gibt es, wie wesentlich sind diese?

4.7.5 Vorgehensmodellspezifische Lösungsmodelle entwickeln

Intention

Die Aktivität V*orgehensmodellspezifische Lösungsmodelle entwickeln* beschreibt, wie mindestens zwei Exemplare des Arbeitsproduktes 4.6.7 „Vorgehensmodellspezifisches Lösungsmodell“ aus den Arbeitsprodukten 4.6.6 „Modell mit Anforderungen an ideales Vorgehensmodell“ und 4.6.5 „Vorgehensmodellmerkmale“ entwickelt werden. Die Bewertungen der einzelnen Vorgehensmodelle finden sich in „Tabelle 9 Zusammenfassung der Vorgehensmodellbewertungen„ in Kapitel 3.3.13 „Vorgehensmodellspezifische Bewertungen“ auf Seite 133.

Die Aktivität V*orgehensmodellspezifische Lösungsmodelle entwickeln*, ihre Eingabe- und ihre Ausgabe-Arbeitsprodukte (kursiv hervorgehoben) zeigt die folgende Abbildung 35.

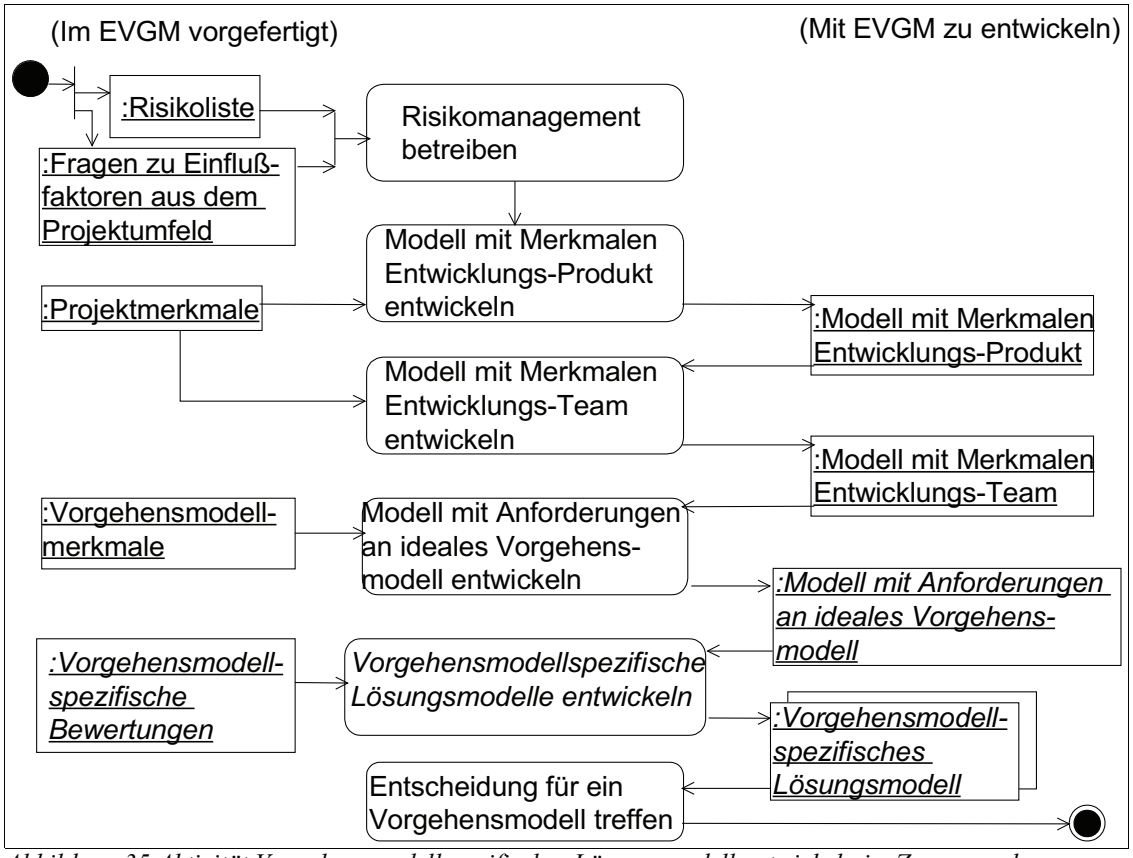

Abbildung 35 Aktivität Vorgehensmodellspezifisches Lösungsmodell entwickeln im Zusammenhang

Inhalt

Vorgehensmodelle auswählen. Das Projekt ist in seinem Soll-Zustand hinsichtlich seiner wesentlichen Merkmale in projektgefährdendem Zustand durch das EVGM-Team modelliert.

Nun werden für das Projekt geeignete Vorgehensmodelle ausgewählt. Dies geschieht anhand der in Kapitel 3 vorgeschlagenen vorgehensmodellspezifischen Bewertungen der *Vorgehensmodellmerkmale*. Am höchsten priorisiert werden hierbei die Vorgehensmodelle aus Kapitel 3, deren Bewertungen am besten den im *Modell mit Anforderungen an ideales Vorgehensmodell* modellierten Anforderungen an ein optimal geeignetes Vorgehensmodell entsprechen.

Vorgehensmodellmerkmale und vorgehensmodellspezifische Bewertungen einsetzen. Dies geschieht jeweils pro in Betracht gezogenem Vorgehensmodell in einem eigenen V*orgehensmodellspezifischen Lösungsmodell*. Es werden jeweils das *Modell mit Anforderungen an ideales Vorgehensmodell* um die für das Vorgehensmodell verbindlichen Merkmale und ihre Beeinflussungsbeziehungen ergänzt und die vorgehensmodellspezifischen Bewertungen des Vorgehensmodells eingesetzt. Im Gegensatz zur idealisierten Darstellung im *Modell mit Anforderungen an ideales Vorgehensmodell* müssen hier auch die *Vorgehensmodellmerkmale* berücksichtigt werden, deren Zustand als nicht förderlich für den Projekterfolg eingeschätzt werden, wenn sie innerhalb des betreffenden Vorgehensmodells verbindlich sind. Beispielsweise müßte für das V-Modell XT [VMX06] in jedem Fall das *Vorgehensmodellmerkmal EV-UnterstProjTheBer-ProjektManagement* modelliert werden, da dieses zum verbindlichen V-Modell-Kern gehört. Die *Vorgehensmodellmerkmale* der jeweiligen Vorgehensmodelle und ihre spezifischen Bewertungen werden dem Kapitel 3 entnommen.

Auswirkungen modellieren. Das Einfügen der neuer *Vorgehensmodellmerkmale* und ihrer spezifischen Bewertungen hat zum Ziel, die Auswirkungen der Einführung eines konkreten Vorgehensmodells auf das Projekt darzustellen. Dies wird nun entlang der Beeinflussungsbeziehungen und gemäß der Einschätzung des EVGM-Teams modelliert.

Modell reduzieren. Nach der Erweiterung des Ursache-Wirkungs-Diagramms wird das ergänzte Modell wieder reduziert um es kompakt und beherrschbar zu halten. Hierbei werden Merkmale entfernt, durch das EVGM-Team als nicht wesentlich beurteilt werden. Hinweise hierauf sind eine vergleichsweise geringe Vernetzung und ein unkritischer Zustand. Es werden keine Merkmale entfernt, die erst durch eine positive Beeinflussung durch ein Merkmal in projektförderlichem Zustand einen unkritischen Zustand erreicht haben oder die innerhalb eines Vorgehensmodells verbindlich sind.

Prüffragen

Folgende Fragen unterstützen die Modellierung der V*orgehensmodellspezifischen Lösungsmodelle*:

- Welchen Zustand erhalten jeweils die *Projektmerkmale* durch die Beeinflussungen der *Vorgehensmodellmerkmale* und deren vorgehensmodellspezifische Zustände?
- Welchen Gesamtzustand erhält das Diagramm jeweils?
- Was können jeweils die *Vorgehensmodellmerkmale* zum Erreichen des Soll-Zustandes des Projektes beitragen?
- Wie kann dieser Soll-Zustand durch die *Vorgehensmodellmerkmale* erreicht und erhalten werden?
- Was wirkt der Verbesserung des Zustandes entgegen?
- Wie beeinflussbar sind die Merkmale und ihr Zustand?
- Welche Auswirkungen haben die Veränderungen auf den Zustand anderer Merkmale?
- Wurden die jeweiligen V*orgehensmodellspezifischen Lösungsmodelle* jeweils ausreichend hinterfragt?
- Besteht ein jeweils ausgewogenes umfassendes Bild der durch die Anwendung der jeweiligen Vorgehensmodelle entstehenden Gesamtsituation?
- Welche über die Vorschläge im EVGM hinausgehenden Merkmale und Beeinflussungsbeziehungen gibt es im Projekt, wie wesentlich sind diese?

4.7.6 Entscheidung für Vorgehensmodell treffen

Intention

Die Aktivität *Entscheidung für Vorgehensmodell treffen* dient der Unterstützung der Entscheidungsfindung für ein Vorgehensmodell, sowie der Darstellung von Anpassungsvorschlägen für dieses. Sie hat zum Ziel, den Anwendern des EVGM eine fundierte, nachvollziehbare Grundlage für ihre schlußendliche Entscheidung zu schaffen. Sie benutzt mindestens zwei Exemplare des Arbeitsproduktes 4.6.7 „Vorgehensmodellspezifisches Lösungsmodell“.

Die Aktivität *Entscheidung für Vorgehensmodell treffen*, ihre Eingabe- und ihre Ausgabe-Arbeitsprodukte (kursiv hervorgehoben) zeigt die folgende Abbildung 36.

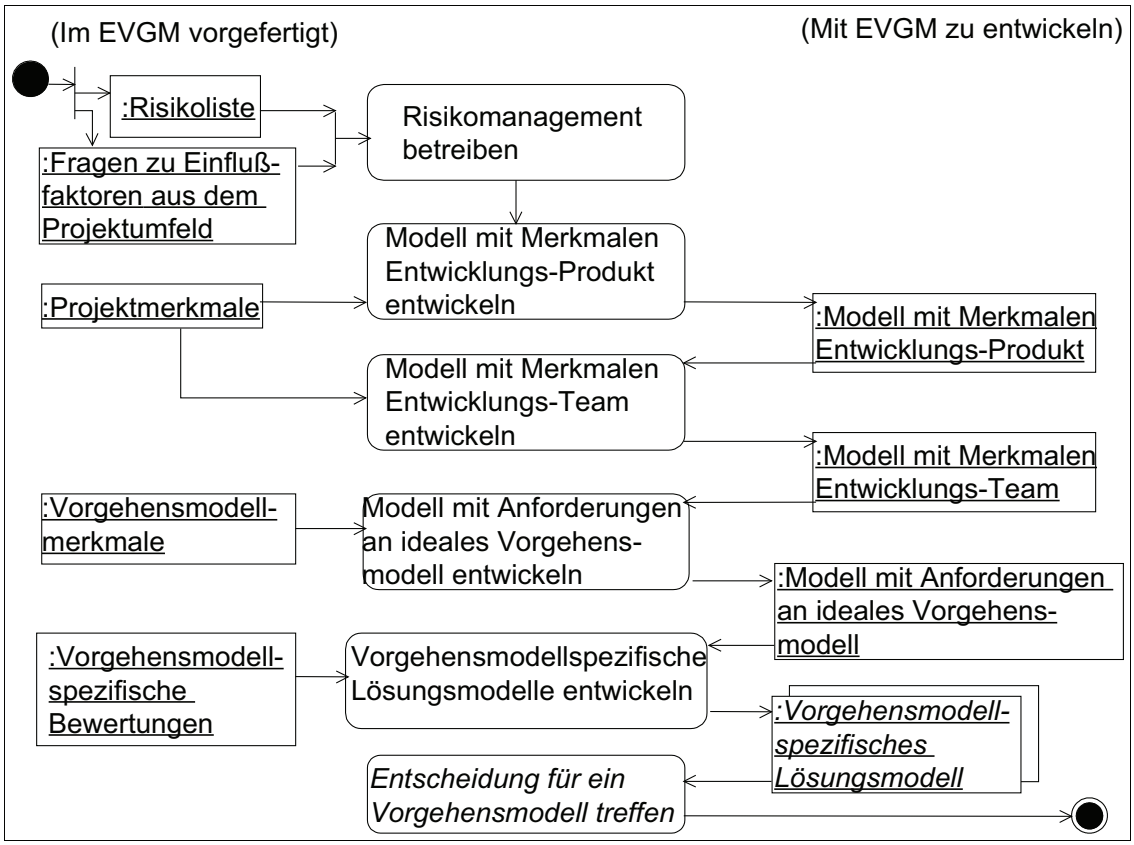

Abbildung 36 Aktivität Entscheidung für Vorgehensmodell treffen im Zusammenhang

Inhalt

Vorgehensmodellspezifische Lösungsmodelle vergleichen. Alle V*orgehensmodellspezifischen Lösungsmodelle* werden bewertet und verglichen. Hierbei sind die Einzelzustände der Merkmale entscheidend.

Ashby's Law anwenden. Ashby's „Law of requisite Variety“ gibt Hinweise für die Bewertung und den Vergleich (Überblick siehe Kapitel 4.2, Vertiefung siehe 5.5 „Prinzipien des EVGM“). Die Bewertung erfolgt hier anhand der Gesamtzustände der einzelnen V*orgehensmodellspezifischen Lösungsmodelle*.

Eigene Bewertung durchführen. Hier wird der subjektive Gesamteindruck des EVGM-Teams ausgewertet. Das Verfahren reduziert für das EVGM-Team die Kompliziertheit der Problemstellung. Es unterstützt ihre Folgerungen, ihr Kombinieren und ihre Entscheidung, kann ihnen diese aber nie abnehmen.

Vorgehensmodell auswählen. Das Vorgehensmodell, dessen V*orgehensmodellspezifisches Lösungsmodell* den durch das EVM-Team am besten beurteilten Gesamtzustand der Merkmale hat, wird ausgewählt. Bewertungen von *Vorgehensmodellmerkmalen*, die schlechter sind als im *Modell mit Anforderungen an ideales Vorgehensmodell*, dienen als Hinweis für Customizing-Maßnahmen, um das Vorgehensmodell projektspezifisch zu optimieren.

4.8 Exemplarischer Durchlauf durch das EVGM

In der folgenden Fallstudie (siehe einführendes Beispiel in Kapitel 1.5) wird anhand eines fiktiven Beispiels mit reduziertem Umfang veranschaulicht, wie das EVGM-Verfahren konkret funktioniert. Die Aktivitäten und Arbeitsprodukte des EVGM werden in der definierten Reihenfolge dargestellt.

Kurzbeschreibung des fiktiven Projektes. Es handelt sich um eine In-house Entwicklung eines Softwareproduktes, die komplett durch ein kompetentes, kleines Team durchgeführt werden soll. Als Stakeholder fungiert ein dezidiert verantwortlicher Produktmanager. Seine Verfügbarkeit für fachliche Rückfragen der Entwickler ist zugesagt. Das Projekt steht unter einem hohen Zeit- und Budgetdruck, da die Konkurrenz mit vergleichbaren Produkten ebenfalls eine Markteinführung anstrebt. Das Produkt hat einen hohen Innovationsgrad, wodurch dem verantwortlichen Produktmanager viele Anforderungen noch nicht klar sind. Die Anforderungen können sich daher auch fortwährend ändern.

4.8.1 Risikomanagement betreiben

Die Betrachtungen zum Risikomanagement hinsichtlich der Einführung eines Vorgehensmodells durch die Risikoliste werden in der Fallstudie nicht berücksichtigt, da sie nicht auf dem Kern des EVGM liegen.
Das Risikomanagement für das in der fiktiven Fallstudie anstehende Softwareentwicklungsprojekt durch die Berücksichtigung der Fragen zu Einflußfaktoren aus dem Projektumfeld aus Kapitel 2 wird im folgenden betrachtet.

Die Fragen zu Einflußfaktoren aus dem Projektumfeld hinsichtlich der *EPSWE-StaTrans-EigenschaftenEntwicklungswerkzeugeUndPlattformen*, der *EPP-Einf-DurchRäumlicheGeografischeVerteilung*, der *EPP-Einf-DurchWirtschaftlichJuristischeVerteilung*, der *EPP-Einf-DurchArtDesVertraglichenVerhältnisses*, der *EPP-Einf-DurchUnternehmenspolitischeRahmenbedingungen*, der *EPP-Einf-DurchOrganisatorischeRahmenbedingungen*, der *EPP-Einf-DurchRisikenDurchAbhängigkeiten* sowie der *EPM-Einf-DurchKompatibilitätZuWertenUndUnternehmenskultur* im Sinne der Definitionen in Kapitel 2 sind mit „Keine wesentlichen Einflüsse“ zu beantworten, da das Projekt eine reine In-house-Entwicklung eines Softwareproduktes sei.

Zusammengefaßt wird in dem Projekt mit vertrauten Werkzeugen und Plattformen gearbeitet, das Projektteam ist an einem Ort und von einer Firma, die politischen und organisatorischen Einflüsse stellen sich unkritisch dar und es gibt keine Kollisionen mit der Unternehmenskultur.

Die Frage zu Einflußfaktoren aus dem Projektumfeld hinsichtlich der *EPD-StaTrans-Projektziele* ist mit „wesentliche Einflüsse“ zu beantworten, da diese nicht gegeben sei. Dies beeinflusst das Merkmal *EPD-StaTrans-FachlicheAnforderungen*. Das führt zu ihrer Auswahl für die Modellierung, sowie auf ihre Bewertung in einem schlechten Zustand, da die Anforderungen von den Zielen abhängen (siehe Kapitel 2).
Die Frage zu Einflußfaktoren aus dem Projektumfeld hinsichtlich der *EPP-Einf-Durch-Entwicklungstiefe* ist mit „wesentliche Einflüsse“ zu beantworten, da der komplette Lebenszyklus des Projektes durch die eigen Organisation abgewickelt wird. Dies beeinflusst die Merkmale *EPD-BedDeck-Fähigk-Fachlich*, *EPSWE-BedDeck-Fähigk-SoftwareEngineering*, *EPSWE-BedDeck-Fähigk-SoftwareManagement* und *EPP-BedDeck-Fähigk-Projektmanagement*. Es versetzt sie in einen schlechten Zustand, da zunächst ein Bedarf zu verbuchen ist, der noch nicht gedeckt worden ist (siehe Kapitel 2).
Die Frage zu Einflußfaktoren aus dem Projektumfeld hinsichtlich der *EPP-Einf-Durch-RahmenbedingungenDesZielmarktes* ist mit „wesentliche Einflüsse“ zu beantworten, da

an dem Markt, an dem das Softwareprodukt plaziert werden soll, erheblicher Kosten- und Zeitdruck herrscht. Dies beeinflusst die Merkmale *EPP-BedDeck-Kap-KalendarischeZeit*, *EPP-BedDeck-Kap-Projektbudget* und *EPP-BedDeck-Kap-Personell*, da aufgrund der Marktsituation den diesbezüglichen Investitionen Grenzen gesetzt sind (siehe Kapitel 2). Es versetzt sie in einen schlechten Zustand, da unabhängig vom Bedarf eine mangelnde Verfügbarkeit festzustellen ist.

Zusammengefaßt sind die fachlichen Ziele und Anforderungen teilweise ungeklärt und verändern sich noch häufig. Das Projekt wird komplett von einem Entwicklungsteam bearbeitet, wodurch entsprechende Fähigkeiten im Software Engineering, Software Management und Projektmanagement erforderlich werden. Am Zielmarkt besteht signifikanter Kosten- und Zeitdruck.

4.8.2 Modell mit Merkmalen Entwicklungs-Produkt entwickeln

Mit Unterstützung der *Projektmerkmale* identifiziert das EVGM-Team die im fiktiven Projekt wichtigen Merkmale aus dem Risikobereich Entwicklungs-Produkt. Hilfreich sind hierfür auch die Fragen zu Einflußfaktoren aus dem Projektumfeld aus Kapitel 2, wie bei der Betrachtung der Aktivität *Risikomanagement betreiben* eben dargestellt wurde.
Aus der Betrachtung ergeben sich nach Einschätzung des EVGM-Teams die Einflussfaktoren

- Fachliche Anforderungen unklar
- Fachliche Anforderungen instabil
- Unveränderlicher Liefertermin
- Unveränderliches Budget

Diese werden im EVGM durch die *Projektmerkmale EPD-StaTrans-FachlicheAnforderungen*, *EPP-BedDeck-Kap-KalendarischeZeit* und *EPP-BedDeck-Kap-Projektbudget* repräsentiert (siehe Kapitel 2).
Diese Projektmerkmale werden nun durch das EVGM-Team zum *Modell mit Merkmalen Entwicklungs-Produkt* vernetzt (siehe Abbildung 37). Der Wert "--" bedeutet, daß sich die Projektmerkmale in projektgefährdendem Zustand befinden. Der Wert "≈" an den Beeinflussungsbeziehungen bedeutet, daß es sich um nebenläufige Beeinflussungen handelt. Im vorliegenden Fall wirkt sie zustandsverschlechternd (siehe auch Erklärung der Notation am Anfang Kapitel 4). Beide Beeinflussungsbeziehungen wurden vom EVGM-Team schwächer eingeschätzt als es das EVGM vorschlägt und daher von "≈≈" auf "≈" abgeschwächt. Die Bewertungen hängen vom konkreten Fall ab und unterliegen der Einschätzung des EVGM-Teams.

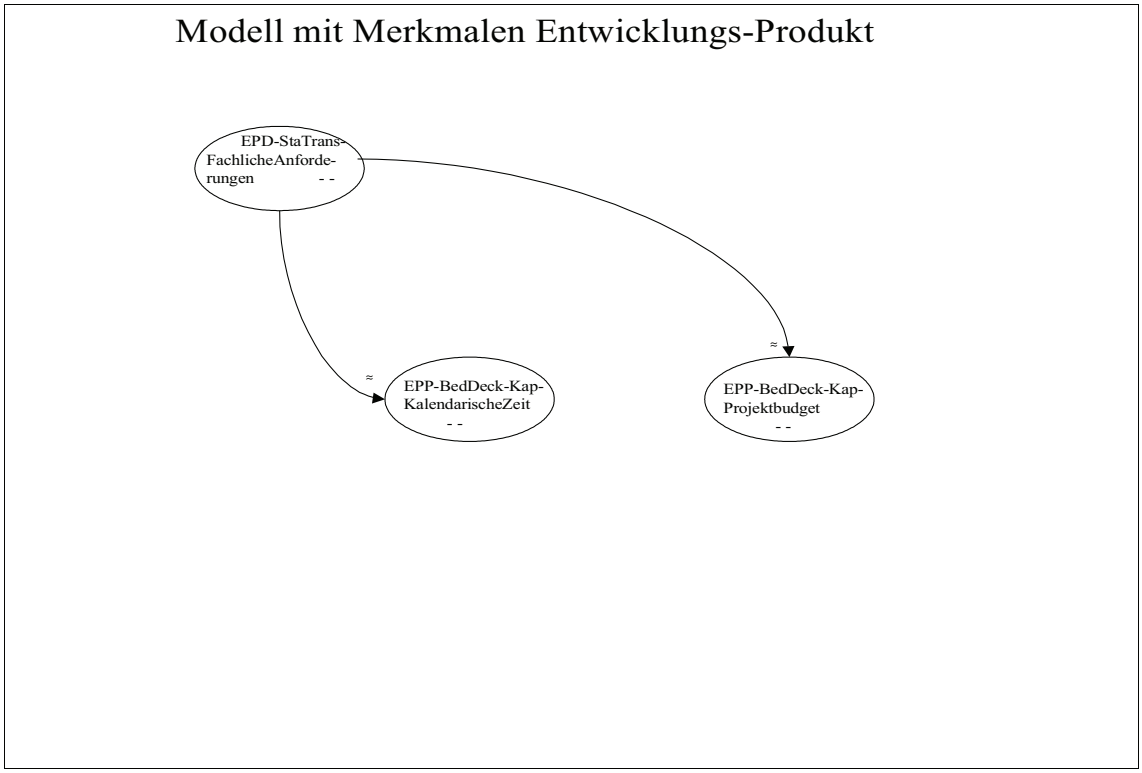

Abbildung 37 Modell mit Merkmalen Entwicklungs-Produkt

Das Modell in Abbildung 37 ist zu interpretieren wie folgt:
Der Zustand von *EPD-StaTrans-FachlicheAnforderungen* verschlechtert den Zustand von *EPP-BedDeck-Kap-KalendarischeZeit* und *EPP-BedDeck-Kap-Projektbudget.* Er erschwert die Aufwandsschätzung und Projektplanung und verursacht potentiell zusätzliche Aufwände zum Einarbeiten von veränderten und neuen Anforderungen und folglich Nacharbeiten an bereits implementiertem Code und Testfällen (siehe Kapitel 2).
Zusammengefaßt wird durch die unklaren und instabilen fachlichen Anforderungen der bereits bestehende Termin- und Budgetengpaß weiter verschärft.

4.8.3 Modell mit Merkmalen Entwicklungs-Team entwickeln

Im Arbeitsprodukt *Projektmerkmale* werden durch das EVGM-Team die im konkreten Projekt wichtigen Merkmale aus dem Risikobereich Entwicklungs-Team zur Entschärfung der *Projektmerkmale* in projektgefährdendem Zustand identifiziert:

- *EPD-BedDeck-Fähigk-Fachlich* (zum Beispiel Verfügbarkeit des Produktmanagers für fachliche Rückfragen der Entwickler, siehe Kapitel 2)
- *EPSWE-BedDeck-Fähigk-SoftwareEngineering* (durch Fertigkeiten und Erfahrung der Entwickler, siehe Kapitel 2)

Durch hinzufügen der personellen *Projektmerkmale* entsteht das *Modell mit Merkmalen Entwicklungs-Team* (siehe Abbildung 38). Der Wert "++" bedeutet, daß sich die neu hinzugefügten Merkmale in einem projektförderlichen Zustand befinden. Der Wert "≈" an den Beeinflussungsbeziehungen bedeutet, daß es sich um nebenläufige Beeinflussungen handelt, die in diesem Fall zustandsverbessernd wirken. Die Beeinflussungsbeziehung zwischen *EPD-BedDeck-Fähigk-Fachlich* und *EPD-StaTrans-FachlicheAnforderungen* wurden vom EVGM-Team schwächer eingeschätzt als es das EVGM vorschlägt und daher von "≈≈" auf "≈" abgeschwächt. Zur besseren Unterscheidung sind die perso-

nellen Merkmale gestrichelt dargestellt.

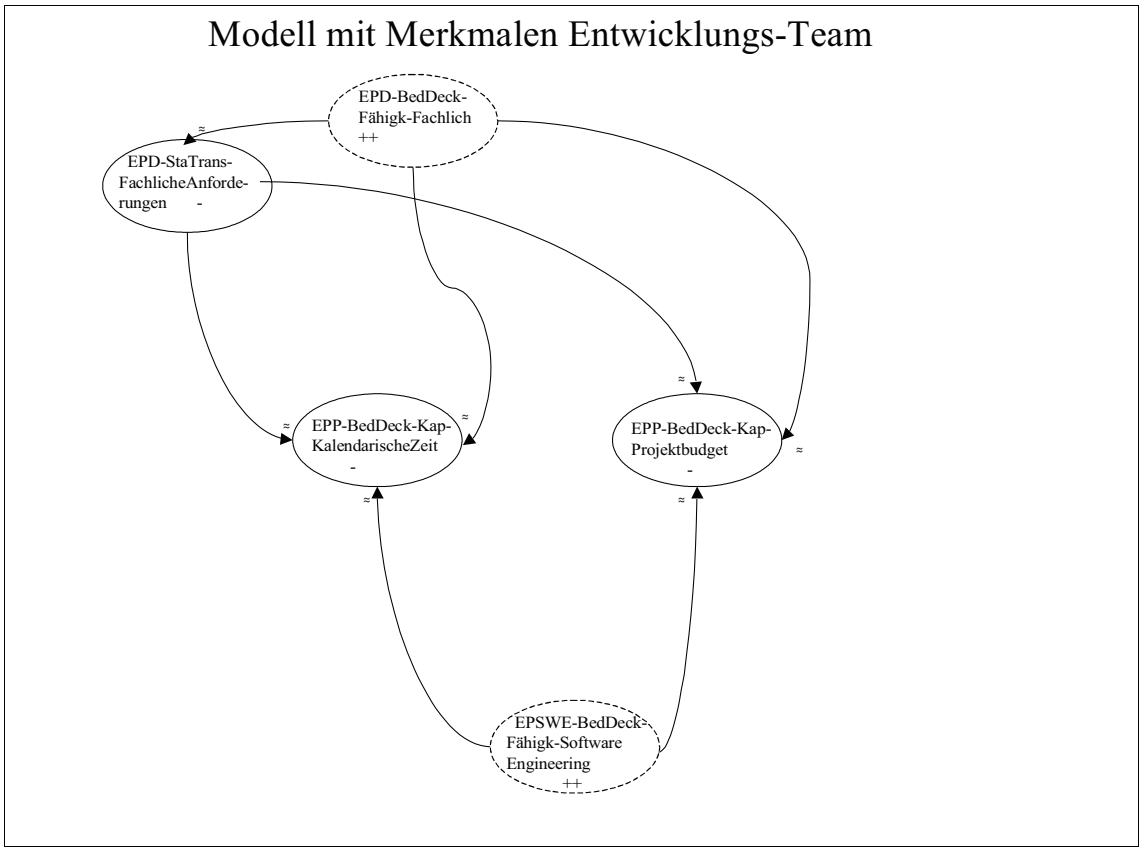

Abbildung 38 Modell mit Merkmalen Entwicklungs-Team

Das Modell in Abbildung 38 ist zu interpretieren wie folgt:
Der Zustand von *EPD-BedDeck-Fähigk-Fachlich* verbessert den Zustand von *EPD-Sta-Trans-FachlicheAnforderungen*, weil Anwender und Entwickler sich auf dem kurzen Weg über fachliche Fragen verständigen und auf dem Laufenden halten können (siehe Kapitel 2).
Er verbessert auch den Zustand von *EPP-BedDeck-Kap-KalendarischeZeit* und *EPP-BedDeck-Kap-Projektbudget*, weil durch verfügbare fachliche Fähigkeiten effektiver und effizienter gearbeitet und mit den Projektressourcen umgegangen werden kann (siehe Kapitel 2).
Der Zustand von *EPSWE-BedDeck-Fähigk-SoftwareEngineering* verbessert den Zustand von *EPP-BedDeck-Kap-KalendarischeZeit* und *EPP-BedDeck-Kap-Projektbudget*, weil es den Softwareentwicklern durch diese Fähigkeit schneller und mit weniger Aufwand möglich ist, das geplante Produkt zu entwickeln (siehe Kapitel 2).

Anders formuliert wird durch die verfügbaren Domänenkenntnisse die Instabilität und Intransparenz der fachlichen Anforderungen teilweise kompensiert. Dadurch werden auch deren negative Auswirkungen auf die Termin- und Budgetsituation abgemildert. Auch die verfügbaren Software Engineering Fähigkeiten tragen dazu bei, die Termin- und Budgetengpässe zu entschärfen.
Wenn Merkmale in projektgefährdendem Zustand der Beeinflussung von Merkmalen in projektförderlichem Zustand ausgesetzt werden, kann das EVGM Team entscheiden, ihnen bessere Werte einzutragen.

4.8.4 Modell mit Anforderungen an ideales Vorgehensmodell entwickeln

Nun werden durch das EVGM-Team im konkreten Projekt sinnvolle *Vorgehensmodellmerkmale* zur Entschärfung der *Projektmerkmale* in projektgefährdendem Zustand identifiziert. Dies erfolgt mit Hilfe der Beeinflussungsbeziehungen aus Tabelle 4 Überblick über Beeinflussung Projektmerkmale durch Vorgehensmodell-Merkmale aus Kapitel 3.2.6 „Zuordnung Vorgehensmodellmerkmale zu Projektmerkmalen“ auf Seite 112.

- *EV-Flexib-DurchIterativitätUndInkrementalität*
- *EV-UnterstProjTheBer-RequirementsEngineering*

Durch das Ergänzen der *Vorgehensmodellmerkmale* entsteht das *Modell mit Anforderungen an ideales Vorgehensmodell* (siehe Abbildung 39). Der Wert “++” bedeutet, daß sich die neuen Merkmale in projektförderlichem Zustand befinden. Der Wert “≈” an den Beeinflussungsbeziehungen bedeutet, daß es sich um nebenläufige Beeinflussungen handelt, die hier zustandsverbessernd wirken. Zur besseren Unterscheidung sind die *Vorgehensmodellmerkmale* mit Strich-Punkt-Linien dargestellt**.**

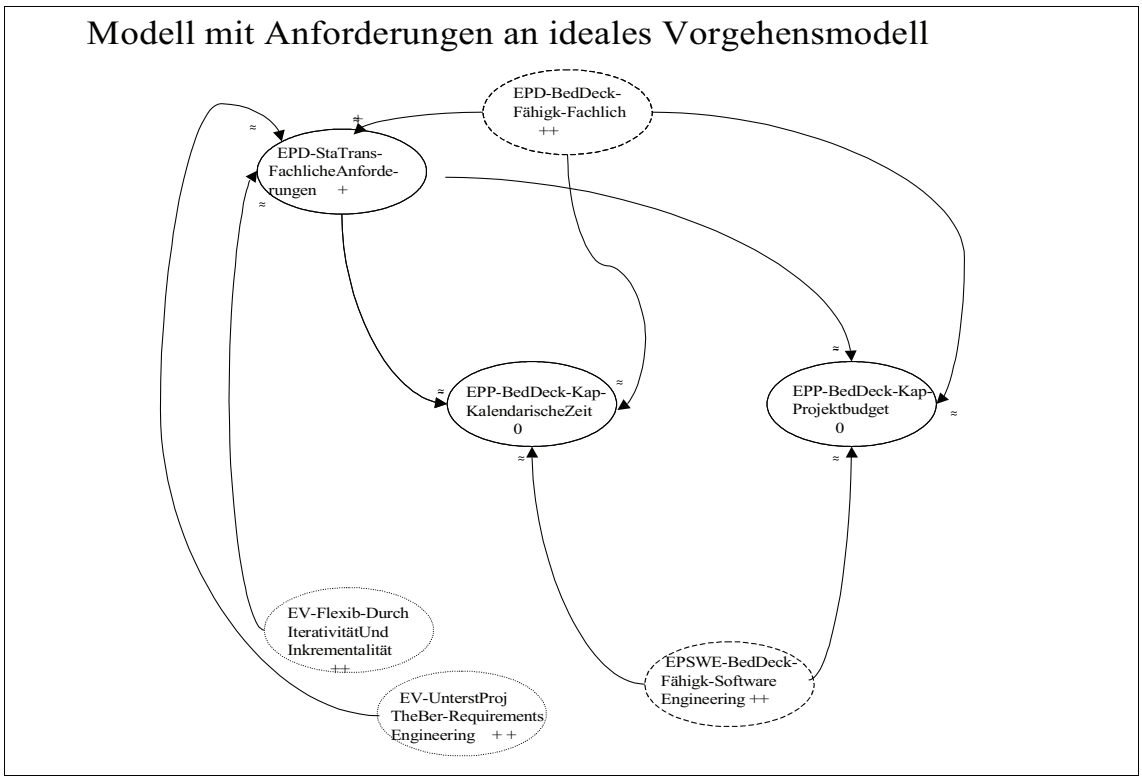

Abbildung 39 Modell mit Anforderungen an ideales Vorgehensmodell

Das Modell in Abbildung 39 ist zu interpretieren wie folgt:
Der Zustand von *EV-Flexib-DurchIterativitätUndInkrementalität* verbessert den Zustand von *EPD-StaTrans-FachlicheAnforderungen,* weil er ein schnelles und häufiges Feedback zur Nachsteuerung der Entwicklung unterstützt. Mehrmaliges Durchlaufen der Entwicklungsphasen erleichtert flexible Anpassungen der Projekt-Arbeitsprodukte an geänderte oder neue Erkenntnisse (siehe Kapitel 2 und 3).
Der Zustand von *EV-UnterstProjTheBer-RequirementsEngineering* verbessert den Zustand von *EPD-StaTrans-FachlicheAnforderungen,* weil eine methodische Anforderungsanalyse das umfassende und fundierte Erheben der Anforderungen unterstützt. Dies erhöht die Transparenz der Anforderungen und auch ihre Stabilität, weil der fachliche Erkenntnisprozeß beschleunigt wird. (siehe Kapitel 2 und 3).

Die Verbesserung des Zustandes von *EPD-StaTrans-FachlicheAnforderungen* kann über die modellierten Beeinflussungsbeziehungen an *EPP-BedDeck-Kap-KalendarischeZeit* und *EPP-BedDeck-Kap-Projektbudget* weiterpropagiert werden. Deren Zustände ändern sich jeweils von „-“ auf „0“ (siehe Abbildung 39).
Die Entscheidung, wie dies im *Modell mit Anforderungen an ideales Vorgehensmodell* zu bewerten ist, obliegt dem EVGM-Team.

Zusammengefaßt würde die Iterativität und Inkrementalität eines idealen Vorgehensmodells die Situation der unscharfen und instabilen fachlichen Anforderungen verbessern. Die von einem optimal geeigneten Vorgehensmodell gebotene Unterstützung für das Requirements Engineering hätte weitere positive Auswirkungen auf den Zustand der fachlichen Anforderungen. Diese Verbesserung hätte auch eine zusätzliche Entschärfung der Termin- und Budgetsituation zur Folge.

Der Zustand des *Lösungsmodells mit Merkmalen ideal geeignetes Vorgehensmodell* hat idealisierten Charakter und ist bewußt noch unabhängig von den Eigenschaften und Unterstützungsmöglichkeiten konkreter Vorgehensmodellen gehalten. Er ist für die folgenden Schritte die Meßlatte für die vom EVGM für das betrachtete Projekt vorgeschlagenen Vorgehensmodelle.

4.8.5 Vorgehensmodellspezifische Lösungsmodelle entwickeln

Nun werden durch das EVGM-Team im konkreten Projekt sinnvolle einsetzbare Vorgehensmodelle anhand ihrer spezifischen Bewertungen der *Vorgehensmodellmerkmale* aus Kapitel 3 in eine Vorauswahl genommen. Das EVGM schlägt für das vorliegende Projekt Scrum und Extreme Programming vor (mit Hilfe von „Tabelle 9 Zusammenfassung der Vorgehensmodellbewertungen“ auf Seite 134 in Kapitel 3.3.13 „Vorgehensmodellspezifische Bewertungen“).

- *EV-Flexib-DurchIterativitätUndInkrementalität* wird durch Scrum und Extreme Programming geboten (siehe Kapitel 3.) Beide haben für dieses Vorgehensmodellmerkmal die spezifische Bewertung „++“.
- *EV-UnterstProjTheBer-RequirementsEngineering* wird Extreme Programming geboten (siehe Kapitel 3). Es hat für dieses Vorgehensmodellmerkmal die spezifische Bewertung „+“, während Scrum „0“ hat.

V-Modell XT und Rational Unified Process werden aufgrund ihres Umfangs als nicht flexibel genug für das vorliegende Projekt bewertet („-“ beziehungsweise „+“). Ihre Unterstützung für das Requirements und Software Engineering ist zwar höher (siehe Kapitel 3), was aber im vorliegenden Projekt durch das Vorhandensein der entsprechenden fachlichen und Software Engineering Fähigkeiten auf anderem Weg und ohne negative Auswirkungen auf die Flexibilität ausgeglichen werden kann.
Das EVGM schreibt vor, alle Vorgehensmodellmerkmale in das *Vorgehensmodellspezifische Lösungsmodell* zu modellieren. Dies würde aber das vorliegende Beispiel mit reduziertem Umfang, welches das Prinzip der Anwendung des EVGM veranschaulicht, überfrachten.

4.8.5.1 Vorgehensmodellspezifisches Lösungsmodell für Scrum

Die vorgehensmodellspezifischen Bewertungen von Scrum ersetzen nun die Bewertungen des ideal geeigneten Vorgehensmodells. Dadurch entsteht ein *vorgehensmodellspezifisches Lösungsmodell* für Scrum (siehe Abbildung 40).

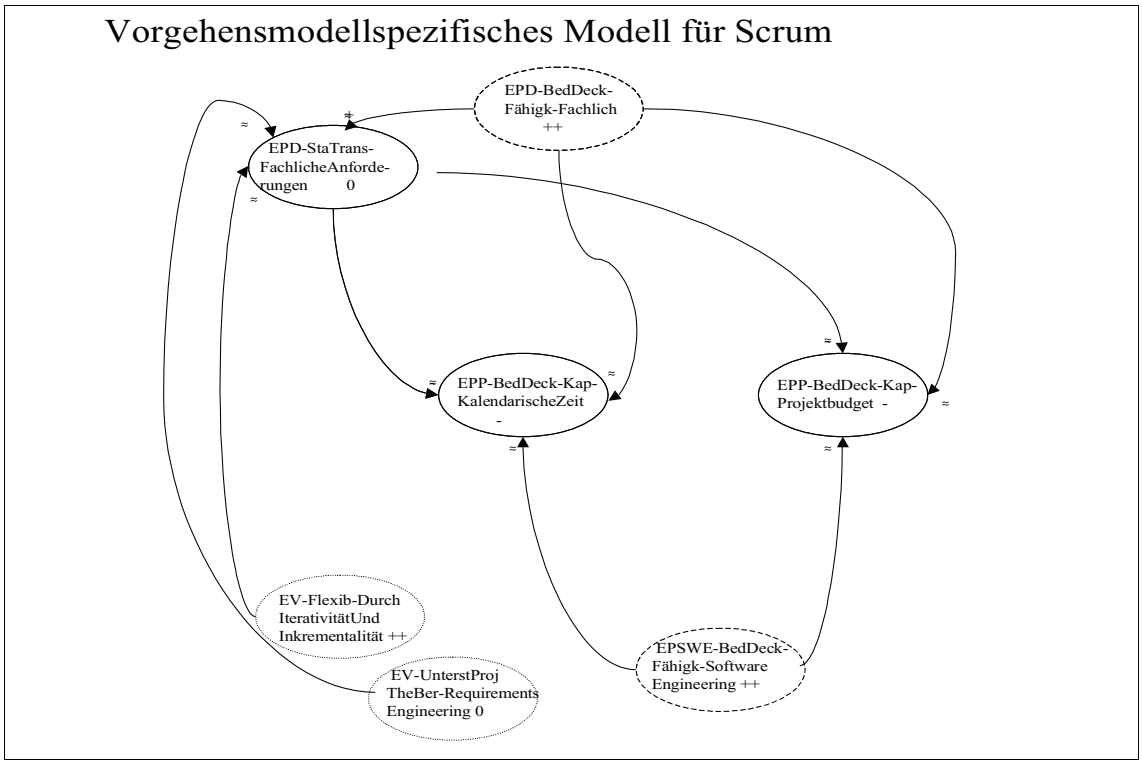

Abbildung 40 Vorgehensmodellspezifisches Lösungsmodell für Scrum

Das Modell in Abbildung 40 ist zu interpretieren wie folgt:
Srcum bietet *EV-Flexib-DurchIterativitätUndInkrementalität*, genügt also hier den Anforderungen aus dem *Modell mit Anforderungen an ideales Vorgehensmodell.* Allerdings bietet es gar keine *EV-UnterstProjTheBer-RequirementsEngineering* (siehe Bewertung im Modell oben). Das EVGM-Team bewertet dies mit Zustandsverschlechterungen von *EPD-StaTrans-FachlicheAnforderungen*, die auf *EPP-BedDeck-Kap-KalendarischeZeit* und *EPP-BedDeck-Kap-Projektbudget* weiterpropagiert werden. Deren Zustände verschlechtern sich jeweils von „0“ auf „-“. Der Gesamtzustand des Modells verschlechtert sich also.

Zusammengefaßt ist Scrum zwar so flexibel, wie es für ein ideal geeignetes Vorgehensmodell gefordert ist, bietet aber nicht die erforderliche Unterstützung im Requirements Engineering.

Dies ist ein Hinweis, daß Scrum kein ideal geeignetes Vorgehensmodell für das untersuchte Projekt ist.

4.8.5.2 Vorgehensmodellspezifisches Lösungsmodell für XP

Nun ersetzen die vorgehensmodellspezifischen Bewertungen von Extreme Programming die Bewertungen des ideal geeigneten Vorgehensmodells. Dadurch entsteht ein *vorgehensmodellspezifisches Lösungsmodell* für Extreme Programming (siehe Abbildung 41).

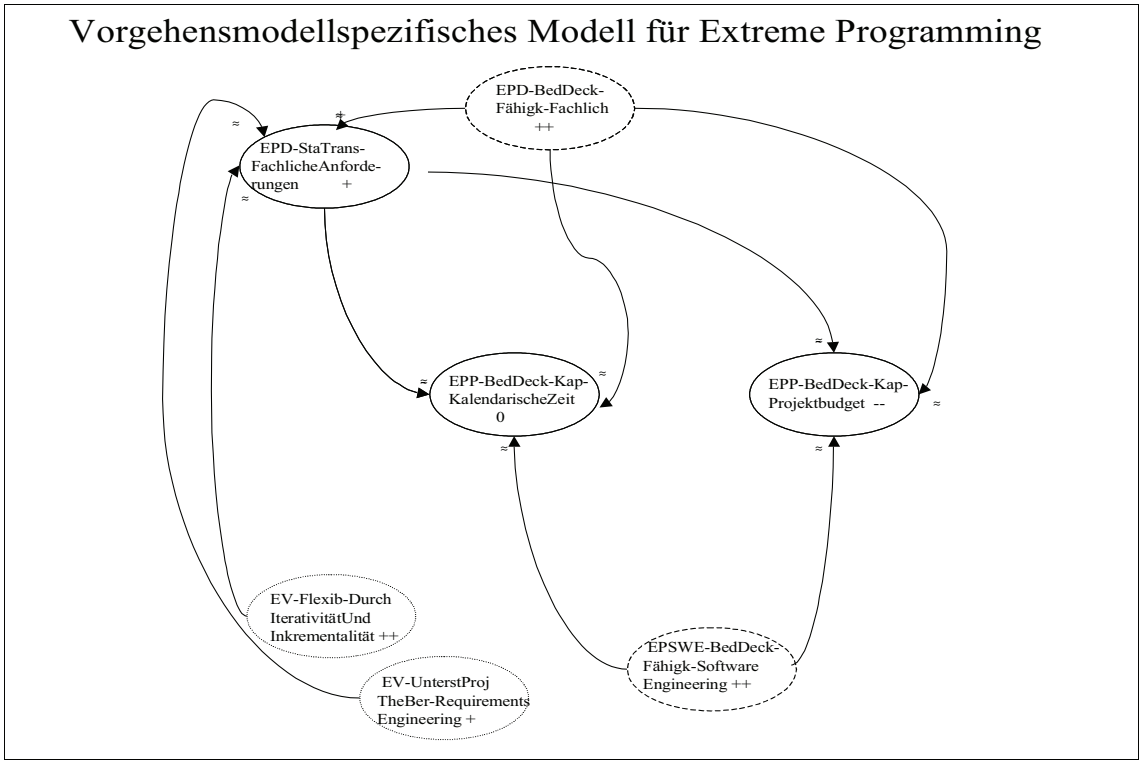

Abbildung 41 Vorgehensmodellspezifisches Lösungsmodell für Extreme Programming

Das Modell in Abbildung 41 ist zu interpretieren wie folgt:
Extreme Programming bietet *EV-Flexib-DurchIterativitätUndInkrementalität*, genügt also hier den Anforderungen aus dem *Modell mit Anforderungen an ideales Vorgehensmodell.* Es bietet weniger *EV-UnterstProjTheBer-RequirementsEngineering* (siehe Bewertung im Modell oben) als im *Modell mit Anforderungen an ideales Vorgehensmodell* gefordert wurde, allerdings mehr als Scrum (siehe oben).

Zusammengefaßt ist Extreme Programming flexibel genug und unterstützt das Requirements Engineering zwar nicht im geforderten Ausmaß, aber besser als Scrum.

Das EVGM-Team entschliesst sich daher, keine Zustände von Projektmerkmale schlechter zu bewerten.

4.8.6 Entscheidung für Vorgehensmodell treffen

Die jeweiligen Gesamtzustände der v*orgehensmodellspezifischen Lösungsmodelle* von Scrum und Extreme Programming können nun verglichen werden. Der Vergleich der beiden *vorgehensmodellspezifischen Lösungsmodelle* unterstützt das EVGM-Team bei der schlußendlichen Entscheidung für den Einsatz von einem der Vorgehensmodelle.
Scrum erreicht nicht alle Bewertungen der Vorgehensmodellmerkmale eines ideal geeignetes Vorgehensmodells, kann also nicht alle Anforderungen abdecken. Durch den Einsatz von Scrum kann also der idealisierte Gesamtzustand des *Lösungsmodells mit Merkmalen ideal geeignetes Vorgehensmodell* nicht gehalten werden.
Extreme Programming erreicht auch nicht alle Bewertungen des ideal geeignete Vorgehensmodell. Es schneidet aber aufgrund der besser eingeschätzten *EV-UnterstProjTheBer-RequirementsEngineering* etwas besser ab als Scrum. Durch den Einsatz von Extreme Programming kann also der idealisierte Gesamtzustand des *Lösungsmodells mit*

Merkmalen ideal geeignetes Vorgehensmodell fast erreicht werden.

Diskussion der Entscheidung für ein Vorgehensmodell. Es ist zu betonen, daß bei der Bewertung und dem Vergleich der V*orgehensmodellspezifischen Lösungsmodelle* die Zustände der Merkmale und Ashby's Law (siehe Kapitel 4.2 und 5.5 „Prinzipien des EVGM“ immer eine Entscheidungsunterstützung für die Benutzer des EVGM darstellen. Deren Ermessensspielraum ist schlussendlich maßgeblich für die getroffene Entscheidung bezüglich eines Vorgehensmodells. Das auch beim ausgewählten Vorgehensmodell Extreme Programming festgestellte Defizit in Bezug auf die *Unterstützung der Disziplin Requirements Engineering* stellt einen Vorschlag zu Prozeßverbesserung dar und läßt sich mit hierfür geeigneten Methoden wie beispielsweise Use Case Modellierung [Jec+04] umsetzen. Diese sind aber nicht Gegenstand der vorliegenden Arbeit.

Zusammenfassung. In diesem Kapitel wurde das EVGM mit seinen Prinzipien, seinen Rollen, Arbeitsprodukten, Aktivitäten und seiner Notation beschrieben. Seine Anwendung wurde anhand eines exemplarischen Durchlaufs für ein fiktives Projekt veranschaulicht.

Im nächsten Kapitel wird zur vorliegenden Arbeit kritisch Stellung bezogen. Zukunftsperspektiven und Möglichkeiten zur Weiterentwicklung des EVGM werden aufgezeigt.

5 Hintergrund und vertiefende Fundierung des EVGM

Dieses Kapitel vertieft die Hintergründe der vorliegenden Arbeit. Herleitungen und grundlegende Argumente in Bezug auf den für das EVGM gewählten Ansatz (5.1) werden dargelegt. Abgrenzungen zu methodisch verwandten Arbeiten verdeutlicht Kapitel 5.2. Die Auswahl der in der vorliegenden Arbeit berücksichtigten Vorgehensmodelle wird plausibel gemacht (5.3) und (5.4). Am Schluß werden die Prinzipien formuliert, denen das EVGM folgt (5.5) und die in der vorliegenden Arbeit verwendeten wissenschaftlichen Methoden (5.6) beschrieben.

Inhalt des Kapitels

5.1 Argumente für den im EVGM gewählten Ansatz

Bevor man die Eignungseinstufung von Vorgehensmodellen mit ihren Eigenschaften propagiert, muß der Nutzen plausibel gemacht werden, der durch die Verwendung von Vorgehensmodellen erzielt wird. Hierfür ist es erforderlich, zunächst häufig auftretende Probleme in Softwareentwicklungsprojekten darzustellen, die durch den Einsatz von Vorgehensmodellen gelindert werden können.

5.1.1 Vorgehensmodellrelevante Probleme in Softwareprojekten

Die heutzutage übliche Komplexität von Projekten ist intuitiv nicht mehr beherrschbar wie bereits dargestellt wurde. Allerdings ist häufig zu beobachten, daß Prozesse in Projekten nicht definiert oder nicht bekannt sind, beziehungsweise projektweise neu formuliert werden. Hierdurch wird jeweils viel hilfreiches Prozeßwissen nicht genutzt oder nicht wiederverwendet. Vorgehensmodelle explizieren Projektprozesse und standardisieren Projektprozesse, in ihnen ist externes Prozeßwissen aufbereitet und internes kann gespeichert werden, wie im folgenden verdeutlicht wird.

5.1.2 Eigenschaften, Nutzen und Grenzen von Vorgehensmodellen

Vorgehensmodelle sind Referenzprozesse für die Softwareentwicklung. Sie sind damit eine Vorlage für tatsächlich in der Organisation gelebte Softwareentwicklungsprozesse und unterstützen die Organisation und Durchführung unterschiedlicher Softwareentwicklungsprojekte. Vorgehensmodelle beschreiben Softwareentwicklung anhand von Modellelementen wie Rollen, Aktivitäten, Arbeitsprodukten, Methoden, Prinzipien und eines diese Elemente verbindenden Prozesses (siehe auch Glossar im Anhang).
Vorgehensmodelle haben gemeinsame Eigenschaften. Sie haben das Ziel, Projektprozesse explizit und transparent auszudrücken und zu standardisieren. Vorgehensmodelle bieten dem Projektteam aufbereitetes externes Software Engineering- und Management-

Wissen.
Die genannten Eigenschaften von Vorgehensmodellen bewirken Nutzen für ihre Anwender, bei einem geeigneten Einsatz in der Summe die Projekterfolgschancen erhöhen. Der Nutzen von Vorgehensmodellen hat aber auch Grenzen, welche ebenfalls dargestellt werden.

5.1.2.1 Vorgehensmodelle explizieren Projektprozesse

Mit Vorgehensmodellen werden Vorgehensweisen im Projekt explizit formuliert. Beispielsweise beschreiben Vorgehensmodelle ausdrücklich, welche Rollen welche Verantwortlichkeiten für welche Arbeitsprodukte haben, wie Arbeitsprodukte beschaffen sein müssen und durch welche Arbeitsschritte sie entwickelt werden.
Dies unterstützt eine Erhöhung der Transparenz des Projektes, die Beherrschung seiner Komplexität und die Verbesserung der Kommunikationsbasis.

Vorgehensmodelle erhöhen die Projekt-Transparenz. Manager von Softwareentwicklungsprojekten haben die Aufgabe, die Projektprozesse zu steuern und müssen hierfür deren Fortschritt kontrollieren. Die dafür erforderliche Transparenz können Vorgehensmodelle unterstützen, weil sie beispielsweise die Projekt-Aktivitäten und -Arbeitsprodukte vorgeben und somit eine Basis für einen Soll-Ist-Vergleich schaffen.
Durch die Erhöhung der Projekttransparenz unterstützen Vorgehensmodelle die Beherrschung der Projektkomplexität.

Vorgehensmodelle helfen, die Projektkomplexität zu beherrschen. Vorgehensmodelle dienen der Bewältigung der Projektkomplexität [Ber04]. Ein komplexes Produkt wie Software kann nur qualitativ hochwertig entwickelt werden, wenn seine Komplexität hinreichend durch ein Vorgehensmodell kompensiert wird [Chr92]. [Ver00] benennt „Fehlende Prozeßmodelle“ als Risikofaktor für Softwareentwicklungsprojekte. [Tha98] bezeichnet den „Prozeß als Schlüssel zum Erfolg“. Bei [Hal98] zitiert nach [Wal04] wird ein nicht hinreichend definierter „Entwicklungsprozess“ als Projektrisiko angesehen. Dies wird durch [Wal04] bestätigt, der einen nicht hinreichend definierten „Entwicklungsprozeß“ als Standardrisiko bezeichnet. Bei [ISA00] zitiert nach [Wal04] wird der Einsatz einer „Systementwicklungsmethode“ als kritischer Erfolgsfaktor des Projektmanagements gesehen.
Die Modellelemente von Entwicklungsprozessen werden durch die explizite Darstellung in Vorgehensmodellen greifbarerer als bei einer impliziten Definition, die beispielsweise ausschließlich in den Hinterköpfen der Mitarbeiter vorliegt. Sie vermindern damit aber auch Interpretationsspielräume innerhalb der Vorgehensweise im Projekt und somit die Auswahl der Lösungsmöglichkeiten und -wege, was die Kommunikationsbasis verbessert.

Vorgehensmodelle verbessern die Kommunikationsbasis im Projekt. Durch die explizite Darstellung von Projektprozessen schaffen Vorgehensmodelle auch eine gemeinsame Kommunikationsbasis im Projektteam. Vorgehensmodelle reduzieren die Begriffsvielfalt bezüglich der Projektprozesse. Sie bilden die Grundlage für eine gemeinsame Sprache und ein gemeinsames Verständnis im Team, wie auch [Ber04] bestätigt. Über die Festlegung von Arbeitsprodukten, Aktivitäten und Rollen wird eine gemeinsame definierte Begriffswelt als Kommunikationsbasis im Team geschaffen. Diese hilft, kostspielige Mißverständnisse im Team zu vermeiden.

5.1.2.2 Vorgehensmodelle standardisieren Projektprozesse

Vorgehensmodelle vereinheitlichen standardisierbare Prozeßbestandteile und definieren Entscheidungsspielräume. Ein Vorgehensmodell unterstützt innerhalb eines Projektes einen geordneten, vereinheitlichten, nachvollziehbaren Entwicklungsprozeß. Sie beschreiben und regeln Vorgehensweisen im Projekt und entlasten somit die Projektmitarbeiter. Vorgehensmodelle definieren und beschränken damit auch die Entscheidungsspielräume der Projektbeteiligten. Sie unterstützen die Projektarbeiter bei Entscheidungen.
Nicht alle Projektmitarbeiter sehen diese unterstützende Eigenschaft von Vorgehensmodellen. Es herrscht bei Mitarbeitern in Softwareprojekten mitunter die Ansicht, daß Vorgehensmodelle die eigentliche Entwicklungsarbeit behindern und nur unnützem Zusatzaufwand, Meßbarkeit und Kontrolle ihrer persönlichen Leistung, sowie Bevormundung mit sich bringen. Sie haben daher Vorgehensmodellen gegenüber Vorbehalte. Auch [Bad02] bestätigt, daß das Bewußtsein für den Nutzen von Entwicklungsprozessen und deren Verbesserung bei Entwicklern gering ist.
Die Standardisierung von Vorgehensweisen durch Vorgehensmodelle weist weitere Nutzen auf. Nicht nur durch die Explizierung, sondern auch durch die Standardisierung von Projektprozessen (s.o.) unterstützen Vorgehensmodelle eine höhere Effizienz, weil sich Mitarbeiter nicht bei jedem Projektwechsel auf neue Vorgehensweisen umstellen müssen. Sie schaffen eine Leitlinie für die Projektabläufe, vermindern aber auch die Auswahl der Lösungsmöglichkeiten und -wege im Projekt. Ein gemeinsames Verständnis der Vorgehensweisen und Abläufe im Projekt reduziert den Aufwand für die Suche nach einem geeigneten Lösungsweg.
Neben der Explizierung verbessert auch die Standardisierung durch Vorgehensmodelle die gemeinsame Kommunikationsbasis im Projektteam. Durch die Vorgabe von Projektprozessen reduziert sich auch der Aufwand für die Abstimmung derselben.

Geeigneter Einsatz von Vorgehensmodellen hilft Reifegrade zu erfüllen. Vorgehensmodelle sind keine Reifegradmodelle für Softwareentwicklungsprozesse wie SPICE [SQI06], [Tha98], Automotive SPICE [ASP07], [Mue+07] oder CMMI [SEI06a], [Kne03]. Reifegradmodelle definieren was ein Prozeß leisten muß, Vorgehensmodelle beschreiben wie er es leistet (siehe auch Kapitel 1.4 „Vergleich mit inhaltlich verwandten Arbeiten“). Der Einsatz von Vorgehensmodellen kann daher helfen, die Anforderungen von Reifegradmodellen zu erfüllen.

5.1.2.3 Vorgehensmodelle speichern externes und internes Prozeßwissen

Vorgehensmodelle bieten strukturiert aufbereitetes Expertenwissen. In bestehenden Vorgehensmodellen ist die Erfahrung und das Expertenwissen ihrer Autoren in Bezug auf Softwareentwicklungsprojekte systematisch und benutzbar verdichtet und aufbereitet [Chr92]. Die Entwicklung eines von Grund auf eigenen Prozesses setzt das Vorhandensein entsprechenden Wissens in der Organisation voraus. Wenn in einer Organisation nicht genügend Wissen über Softwareentwicklungsprozesse und Softwareentwicklungs-Managementprozesse vorhanden ist, behindert dies den organisationsspezifischen Entwurf geeigneter Softwareentwicklungsprozesse. Dann ist die Vorgehensweise, ein vorgefertigtes Vorgehensmodell für eine gegebene Problemstellung zu benutzen, mitunter besser zu handhaben.

Vorgehensmodelle repräsentieren den spezifischen Erfahrungshintergrund ihrer Autoren. Vorgehensmodelle adressieren eine begrenzte Menge von Projektmerkmalen, da sie jeweils den individuellen Erfahrungshintergrund ihrer Autoren und den daraus resultierenden Bedarf an Unterstützung durch Prozesse widerspiegeln. Die Nutzung des Gestaltungsspielraums beim Entwurf von Vorgehensmodellen richtet sich an den von den Autoren wahrgenommenen Problemschwerpunkten bei deren Tätigkeit in sehr unterschiedlichen Arten von Softwareentwicklungsprojekten aus. Deshalb stellt sich die Landschaft der bestehenden Vorgehensmodelle als äußerst variantenreich dar. Vorgehensmodelle sind immer nur für eine Teilmenge von Softwareentwicklungsprojekten geeignet [Oes+99].

Vorgehensmodelle ergänzen technisches Wissen durch Prozeß-Aspekte. Vorgehensmodelle werden in der Informatik-Ausbildung nicht immer vermittelt. Wenn sie in der Ausbildung nicht umfassend dargestellt und ihr Nutzen plausibel motiviert werden, haben die Studenten als spätere Projektmitarbeiter keine entsprechenden Vorkenntnisse und oft keine Erfahrung in Bezug auf Vorgehensmodelle. Projektmitarbeiter in Softwareentwicklungsprojekten haben nicht immer das Bewußtsein und den Blick für den Nutzen von Prozeß- und Organisationsfestlegungen. Methodik und Prozesse stehen meist im Hintergrund hinter technischen Belangen. Vorgehensmodelle ergänzen das technische Wissen von Projektmitarbeitern, beispielsweise in Bezug auf Werkzeuge oder Programmiersprachen, um Prozeß-Aspekte. Dafür erfordern sie von ihren Anwendern, ihren Blickwinkel von technischen, fachlichen und kaufmännischen Facetten um Prozeß- und Organisationsaspekte zu erweitern. Hierfür ist bei den Projektmitarbeitern Umdenken, sowie Verständnis für den Nutzen von Vorgehensmodellen erforderlich. Standardisierte Entwicklungsverfahren sind in den älteren Ingenieurdisziplinen in Projekt und Ausbildung vergleichsweise etabliert [Chr92]. Dies stellt einen Gegensatz zur vergleichsweise seltenen Verwendung von Vorgehensmodellen im Software Engineering dar.
Durch das Anreichern des im Projekt verfügbaren Software Engineering Wissen können Vorgehensmodelle eine Erweiterung des methodischen Lösungsspektrums im Projekt bewirken.

Vorgehensmodelle dienen als Speicher für organisationsspezifisches Prozeßwissen. Durch ihre organisationsspezifische Anpassung, Ausgestaltung, Einbettung und Interpretation für die betreffende Organisation werden Vorgehensmodelle um projektspezifisches Organisations- und Prozeßwissen angereichert. Dieses Wissen wird in der Vorgehensmodellbeschreibung schriftlich ergänzt. Diese stellt somit einen Wissensspeicher und die Definition eines Standards dar. Das Wissen ist personenunabhängig für die Organisation verfügbar.

5.1.2.4 Geeignete Vorgehensmodelle erhöhen Chancen für Projekterfolg

Durch die bessere Transparenz, Standardisierung und Wissensbasis erhöhen Vorgehensmodelle durch ihren geeigneten Einsatz die Chancen für einen Projekterfolg und rechtfertigen so die bei ihrer Einführung getätigten Anfangsinvestitionen.
Hierzu muß verdeutlicht werden, daß es ohne die übergeordnete Sicht eines explizit formulierten Entwicklungsprozesses deutlich schwieriger ist, über die Komplexität eines Softwareentwicklungsprojektes den Überblick zu behalten. Dadurch werden Ressourcen wie Zeit und Geld ineffizient eingesetzt und so der Projekterfolg gefährdet. Aktuell wer-

den bei weitem nicht in allen Softwareentwicklungsprojekten in Deutschland Vorgehensmodelle eingesetzt beziehungsweise befolgt (vergleiche [EVA00]). Bezüglich des Einsatzgrades von methodisch ausgewählten Vorgehensmodellen in der Praxis führte der Autor der vorliegenden Arbeit Experteninterviews durch (siehe 1.2 „Heutige Situation beim Einsatz von Vorgehensmodellen"), welche diese Sachlage bestätigten. Auch die geringe Verfügbarkeit von unabhängigen Erfahrungsberichten in der Literatur über den Einsatz von Vorgehensmodellen (siehe auch Abschnitt 5.3) kann nicht als Indiz für die häufige und bedarfsgerechte Anwendung von Vorgehensmodellen in der industriellen Praxis gewertet werden.
Vorgehensmodelle sind für die Entwicklung guter Software nicht hinreichend, weil sie immer durch Projektmitarbeiter ausgeführt werden müssen. Sie sind nicht einmal zwingend notwendig da nicht ausgeschlossen werden kann, daß auch ohne die Verwendung von Vorgehensmodellen in Projekten qualitativ gute Software entwickelt wird.
Ein Vorgehensmodell ist kein Erfolgsgarant für ein Softwareentwicklungsprojekt. Ein geeignetes Vorgehensmodell kann aber die Erfolgswahrscheinlichkeit eines Softwareprojektes erhöhen, in dem es die Projektmitarbeiter bedarfsgerecht unterstützt, die projektspezifischen Schwierigkeiten zu meistern.

Geeignete Vorgehensmodelle rechtfertigen Anfangsinvestitionen. Prozeßverbesserungen kosten Aufwand [Hum90]. Prozeßverbesserungen durch die Einführung von Vorgehensmodellen erfordert initiale Zusatzinvestitionen und -aufwände durch den Aufbau der erforderlichen Infrastruktur und das Einlernen der Projektmitarbeiter. [Tha98] belegt den Amortisationseffekt von Prozeßverbesserungen anhand einiger Beispiele. Auch [Wal07] und [Hör06] berichten davon, daß sich die Investitionen für Prozeßverbesserungen durch bessere Planbarkeit, Komplexitätsbeherrschung, Produktqualität und Kundenzufriedenheit in Softwareprojekten bei gleichzeitig geringeren Kosten und Zeitbedarf mehrfach amortisieren können. Die Einführung eines geeigneten Vorgehensmodells ist eine Möglichkeit, die Softwareentwicklungsprozesse einer Organisation zu verbessern. Bei der Prozeßverbesserung durch den Einsatz eines geeigneten Vorgehensmodells können sich also die Investitionen im Projektverlauf durch ihren oben verdeutlichten Nutzen amortisieren und sich so auf längere Sicht rechtfertigen.

Nachdem der Nutzen von Vorgehensmodellen für Softwareentwicklungsprojekte transparent gemacht wurde, werden nun die Vorteile eines Eignungseinstufungsverfahrens für Vorgehensmodelle verdeutlicht.

5.1.3 Warum eine Eignungseinstufung für Vorgehensmodelle?

Warum ist es sinnvoll, Vorgehensmodelle hinsichtlich ihrer projektspezifischen Eignung für ein Projekt einzustufen?
Softwareprojekte sind meist komplex. Eine Vielzahl von fachlichen, technischen, kaufmännischen, organisatorischen oder terminlichen Einflußfaktoren stehen in Wechselwirkung miteinander. Softwareprojekte sind hinsichtlich ihrer charakterisierenden Eigenschaften, Problemschwerpunkte und Rahmenbedingungen äußerst variantenreich und diversifiziert (siehe auch [Bal96]).
Auch Vorgehensmodelle sind aufgrund der jeweiligen Anzahl ihrer Aktivitäten, Rollen und Arbeitsprodukte oft komplex. Die Ausrichtung ihrer Modellelemente und Ausgestaltung von deren Anordnung hängt stark von den bei ihrem Entwurf im Vordergrund gestandenen Problemschwerpunkten in Projekten ab (siehe auch [Boe04]).

Da sowohl Softwareprojekte als auch Vorgehensmodelle komplexe Eigenschaften haben, ist die Eignungseinstufung von Vorgehensmodellen für Softwareprojekte ebenfalls ein komplexes Vorhaben, welches mit Hilfe eines methodischen Verfahrens besser beherrschbar wird.
Der Schlüssel für die Nutzenpotentiale von Vorgehensmodellen ist deren Eignung für eine konkrete Problemstellung. Es gibt kein universell optimales und gleichzeitig konkretes Vorgehensmodell. Dafür sind die Projekte hinsichtlich von Problemstellungen und Problemschwerpunkten zu stark diversifiziert, wie auch [Boe04] bestätigt. Ein universell geeignetes Vorgehensmodell müßte sehr abstrakt bleiben und könnte somit keine hinreichend konkrete Unterstützung im Projekt bieten [Oes+99]. Ein Vorgehensmodell muß auf die Anforderungen durch das Entwicklungs-Produkt und das Entwicklungs-Umfeld passen, sowie den Stärken und Schwächen des Entwicklungs-Teams gerecht werden, wie auch [Boe04] betont.
Die Vorteile eines Eignungseinstufungsverfahrens für Vorgehensmodelle wurden auf allgemeiner Basis verdeutlicht. Nun wird diskutiert, welche speziellen Eigenschaften des EVGM welchen spezifischen Nutzen bewirken.

5.1.4 Eigenschaften, Nutzen und Grenzen des EVGM

Warum hat das EVGM genau die vorliegenden Eigenschaften? Das EVGM ist wie ein Vorgehensmodell aufgebaut und hat die gleichen Grundeigenschaften. Das EVGM expliziert und standardisiert wesentliche Vorgehensweisen bei der Eignungseinstufung von Vorgehensmodellen, bereitet projektunabhängiges, externes Fachwissen auf und dient als Speicher für projektspezifisches, internes Wissen. Die Eigenschaften des EVGM, sowie sein Nutzen und seine Grenzen beziehen sich auf das Verfahren der Eignungseinstufung, sowie auf dessen Inhalt in Form von Fachwissen über Software Engineering, Software- und Projektmanagement, sowie Menschenführung.

5.1.4.1 Das EVGM expliziert den Eignungseinstufungs-Prozeß

Explizierung von Verfahren und Inhalten des EVGM-Prozesses

Eigenschaft. Das EVGM beschreibt konkret Rollen, Aktivitäten und Arbeitsprodukte, sowie die enthaltenen Inhalte bezüglich Software Engineering, Software Management, Projektmanagement und Menschenführung.
Nutzen. Dadurch ist es unmittelbar anwendbar, Begriffe und Verfahren sind konkretisiert, eine Kommunikationsbasis in Form einer schriftlichen Terminologie ist geschaffen. Der Prozeß ist überprüfbar.
Grenzen. Diese vorgefertigten Inhalte können allerdings eine nicht gegebene Abgeschlossenheit und Vollständigkeit suggerieren. Sie unterstützen das eigenständige Denken der Benutzer nicht.

Modellbasierter Ansatz mit vorgefertigten Merkmalen und Beeinflussungsbeziehungen

Eigenschaft. Die Explizierung der Inhalte des EVGM erfolgt modellbasiert. Im EVGM werden vorgefertigte Merkmale und deren Beeinflussungsbeziehungen in Form von Modellelementen bereitgestellt. Das Verfahren selbst ist durch die Beschreibungen von Aktivitäten und Arbeitsprodukten definiert.
Nutzen. Die Modellelemente bewirken eine Präzisierung des Verfahrensablaufes, sowie von dessen Inhalten. Eine Kommunikationsbasis für die Benutzer wird geschaffen.

Durch seine Modellbasierung hilft das EVGM, die Komplexität des Eignungseinstufungsprozesses besser zu überblicken. Modelle reduzieren die Komplexität der Realität zielführend auf ein beherrschbares Maß, weil sie nur als wesentlich erachtete Aspekte enthalten (siehe auch [Chr92] beziehungsweise Glossar im Anhang). Durch ihre Kompaktheit und Definiertheit sind Modelle besser auf Konsistenz überprüfbar als natürlichsprachlicher Text.
Die modellbasiert dargestellten Inhalte des EVGM besitzen Vorschlagscharakter. Das Verfahren des EVGM wird durch die explizite Darstellung in Modellen besser handhabbar. Die Modellbasierung im EVGM fördert eine präzise Ausdrucksweise bezüglich von Merkmalen und Beeinflussungsbeziehungen und hilft dadurch, die Problemkomplexität besser zu beherrschen. Sie bietet auch eine präzisere Kommunikationsbasis als natürlichsprachliche Definitionen für die Benutzer des EVGM.
Grenzen. Nur durch die vordefinierten Modellelemente abgedeckte Merkmale und Beeinflussungsbeziehungen können im Rahmen des EVGM ausgedrückt und dargestellt werden, ohne Erweiterungen oder Veränderungen des Sprachschatzes vorzunehmen. Dies begrenzt die Ausdrucksmächtigkeit des Verfahrens.

5.1.4.2 Das EVGM standardisiert den Eignungseinstufungs-Prozeß

Standardisierung von Aktivitäts- und Arbeitsproduktfolgen

Eigenschaft. Das Verfahren des EVGM ist durch die Beschreibungen von Aktivitäten, Arbeitsprodukten und deren Abfolge standardisiert.
Nutzen. Dadurch ist die Komplexität der Eignungseinstufung, sowie des Verfahrens hierfür beherrschbar und nachvollziehbar, sein Ergebnis überprüfbar. Begriffe und Verfahren sind vereinheitlicht, eine Kommunikationsbasis in Form einer kompakten Terminologie wird zur Verfügung gestellt.
Grenzen. Die Standardisierung kann eine nicht gegebene Vollständigkeit und Abgeschlossenheit suggerieren, sie unterstützt nicht das selbständige Denken seiner Benutzer.

Standardisierte Schritt für Schritt Anweisungen

Eigenschaft. Das EVGM unterstützt seine Anwender, die komplexe Problemstellung der Wahl eines Vorgehensmodells methodisch anzugehen. Innerhalb der einzelnen Aktivitäten im EVGM wird ein direkt anwendbares Schritt für Schritt Vorgehen angeboten. Es bereitet eine Veränderung und Verbesserung der bestehenden Softwareentwicklungs-Prozesse systematisch vor.
Nutzen. Dadurch ist das Verfahren direkter anwendbar, besser beherrschbar, nachvollziehbar und somit sein Ergebnis leichter überprüfbar. Die Auswahl eines Vorgehensmodells ist auch im nachhinein nachvollziehbar. Dies ist möglich, da die Arbeitsprodukte der Aktivitäten schriftlich dokumentiert und modelliert werden müssen. Auch iterative Nachbesserungen der Eignungseinstufung werden so erleichtert. Dadurch können in Feedbackzyklen weitere zielgerichtete Optimierungen der Softwareentwicklungsprozesse durchgeführt werden, wie es zum Beispiel auf CMMI-Level 5 [Kne03] gefordert wird.
Durch seine standardisierte Begriffssystematik hilft das EVGM seinen Anwendern, das Projekt systematisch, umfassend und komplett zu untersuchen und sich hierbei bestehende Zusammenhänge zu vergegenwärtigen. Ein gemeinsames Bewußtsein und eine Kommunikationsbasis des EVGM-Teams für Prozeß-Aspekte ist somit geschaffen. Dadurch ist die Komplexität der Aufgabenstellung, Vorgehensmodelle hinsichtlich ihrer Eignung für ein Projekt einzuschätzen, auf ein für die Zielgruppe beherrschbares Maß

reduziert.
Grenzen. Die Standardisierung kann eine nicht gegebene Komplettheit und Abgeschlossenheit suggerieren, das eigenständige Denken der Benutzer kann sie nicht unterstützen.

5.1.4.3 Das EVGM speichert externes und internes Prozeßwissen

Im EVGM ist externes Software Engineering- und Managementwissen für den Eignungseinstufungsprozeß aufbereitet. Zusätzlich dient es als Informationsspeicher für organisationsspezifisches Prozeß- und Projektwissen, mit dem es angereichert werden kann.

Anreichern von Fachwissen durch Modellelemente und Benutzerbeteiligung
Eigenschaft. Durch die Merkmale und Beeinflussungsbeziehungen im EVGM wird der Eignungseinstufungsprozeß durch externes projekt- und organisationsunabhängiges Fachwissen unterstützt. Internes und projektspezifisches Benutzerwissen und Expertise wird abgefragt, kanalisiert und systematisch in das Verfahren eingebracht. Das EVGM unterstützt somit das Organisieren, Kombinieren und Kanalisieren von spezifischem Expertenwissen über das Projekt für dessen Analyse. Das EVGM bietet verdichtetes, projektunabhängiges Expertenwissen über Software Engineering, Vorgehensmodelle und Software Prozeßverbesserung und ergänzt so vorhandenes Expertenwissen über das Projekt.
Nutzen. Der Eignungseinstufungsprozeß gewinnt so an verfügbarer Fachkompetenz, sein Ergebnis an Fundiertheit und Akzeptanz. Die Benutzer führen das Verfahren im Wissen ihrer Verantwortlichkeit für das Ergebnis aus. Durch seine obligatorische Benutzereinbindung bietet und fördert das EVGM Hilfe zur kurz- und langfristigen Selbsthilfe für seine Anwender.
Grenzen. Der Nutzen wird durch die Kenntnisse der Benutzer über das fragliche Projekt begrenzt. Vorgefertigte Inhalte unterstützen nicht das eigenständige Denken der Benutzer.

Multiperspektivische Betrachtung durch Anforderungen an die Stakeholder
Eigenschaft. Das EVGM stellt die Anforderung, Entwicklungsprozeß-Stakeholder unterschiedlicher Fachdisziplinen, Rollen, Ausbildungs- und Erfahrungshintergründe am Eignungseinstufungsprozeß zu beteiligen.
Nutzen. Dadurch hilft das EVGM den Anwendern, ihr Softwareprozeßproblem aus verschiedenen Perspektiven zu untersuchen, um es somit umfassender und gründlicher durchdringen zu können. Hierbei wird das spezifische Wissen unterschiedlicher Fachdisziplinen in den Eignungseinstufungsprozeß eingebracht. Es unterstützt sie, sich bestehende, interdisziplinäre Zusammenhänge im Projekt besser bewußt zu machen und diese berücksichtigen zu können. Das EVGM verfolgt einen Ansatz, der das Projekt und Unterstützungsoptionen durch Vorgehensmodelle umfassend darstellt, um alle wesentlichen Merkmale zu berücksichtigen.
Grenzen. Der Nutzen begrenzt sich an den Fähigkeiten und dem Wissen der Stakeholder über ihr Projektumfeld.

Qualitativer Ansatz durch Bewertung der Modellelemente
Eigenschaft. Bei der Beurteilung der Eignung von Vorgehensmodellen verfolgt das EVGM einen qualitativen Ansatz. Im EVGM werden die Modellelemente hinsichtlich ihres Zustandes durch seine Benutzer qualitativ bewertet. Hierbei werden Werte einer

Ordinalskala benutzt, die von „--“ bis „++“ geht (siehe Kapitel 4).
Nutzen. Dadurch wird eine problemangemessene Ausdrucksmächtigkeit und Genauigkeit der Beschreibung erreicht, weil Wissen über Zustände wesentlicher Projektmerkmale, sowie über Zustandsverbesserungen oder -verschlechterungen von Modellelementen ausgedrückt werden können, aber keine unrealistische Genauigkeit suggeriert wird.
Grenzen. Eine Quantifizierung innerhalb der Bewertungen ist nicht sinnvoll. Dies rührt daher, daß für die Beurteilung von Projektmerkmalen individuelle und somit subjektive Einschätzungen der EVGM-Teammitglieder über das Projekt herangezogen werden. Daher ist für Modelle mit rechnerischer Genauigkeit keine mathematische Grundlage vorhanden. Somit ist es nicht möglich, auf den Zuständen zu rechnen.

Vernetzte Betrachtung durch Berücksichtigung der Wechselwirkungen
Eigenschaft. Das EVGM beschreibt die Beeinflussungsbeziehungen zwischen den Merkmalen von Projekten und von Vorgehensmodellen in Softwareentwicklungsprojekten. Somit wird das Wissen um die Wirkweise von Vorgehensmodellen in das Verfahren eingebracht. Die wichtigen Merkmale in Softwareentwicklungsprojekten werden im EVGM im Zusammenhang betrachtet.
Auch etablierte Ansätze wie COCOMO [Boe81] und COCOMO II [Boe00] fassen Softwareentwicklungsprojekte als System konkurrierender und zu balancierender Ziele auf, extrahieren die wesentlichen Größen und setzen sie zueinander in Beziehung. Im Gegensatz zum EVGM folgen sie einem quantitativen Ansatz mit dem Ziel der Kostenschätzung.
Nutzen. Durch die vernetzte Betrachtung können Zusammenhänge ausgedrückt und Seiteneffekte der Zustandsveränderungen von Merkmalen dargestellt werden. Die dynamischen Facetten der Problemstellung werden sichtbar.
Ein alternativer Ansatz, der die identifizierten Merkmale isoliert betrachtet, würde die Beeinflussungsbeziehungen zwischen den Merkmalen vernachlässigen, die das Verhalten eines Systems maßgeblich mitbestimmen. Er würde also wichtige Eigenschaften im Projekt ignorieren, wodurch ein verfälschtes und irreführendes Bild entstünde. Eine Lösung, welche die existierenden Wechselwirkungen nicht berücksichtig, kann das Problem nicht optimal lösen.
Grenzen. Die Anzahl der Modellelemente wächst durch das Einbeziehen der Beeinflussungsbeziehungen. Dies verkompliziert die Darstellung und verringert ihre Übersichtlichkeit.

5.1.4.4 Das EVGM erhöht die Chancen für Projekterfolg

Durch die Unterstützung der bedarfsgerechten Auswahl eines Vorgehensmodells trägt das EVGM dazu bei, die Chancen für den Projekterfolg zu erhöhen.
Projektspezifischer Ansatz durch Benutzerbeteiligung und Anpaßbarkeit
Eigenschaft. Das EVGM berücksichtigt und adressiert die spezifischen Eigenheiten eines Projektes oder einer Organisation. Dies erfolgt durch die Beteiligung der späteren Anwender des auszuwählenden Vorgehensmodells, sowie durch die Möglichkeit, Inhalte des Verfahrens spezifisch anzupassen.
Nutzen. Durch den projektspezifischen Ansatz wird die Problemanalyse und der Lösungsentwurf individuell optimierbar. Dies ist erforderlich, da sich komplexe Probleme Pauschallösungen entziehen [Ves99]. Durch die projektspezifische, zielgerichtete Auswahl eines Vorgehensmodells mit Hilfe des EVGM wird die Effizienz, Effektivität und Qualität der Softwareentwicklung spezifisch für das gegebene Projekt erhöht. Hierbei

sind subjektive Meinungen der Projektmitarbeiter wichtig. Es wird berücksichtigt, daß Softwareentwicklungsprojekte und -Organisationen stark diversifiziert sind.
Der Vergleich der Vorgehensmodelle erfolgt dabei auf Basis des direkten, unmittelbaren Nutzens bei der Adressierung projektspezifischer Probleme. Dies rührt daher, daß die verschiedenen Vorgehensmodelle mitunter verschiedene Modellelementtypen verwenden. Beispielsweise formuliert XP Values, Principles, Practices und Techniques. Das V-Modell XT setzt sich hingegen aus Rollen, Aktivitäten und Produkten zusammen. Diese Vorgehensmodelle sind nicht direkt auf der Ebene der Modellelemente vergleichbar. Ein Vergleichsraster, sowie auch eine Abgrenzung von Vorgehensmodellen von Nicht-Vorgehensmodellen ist grundsätzlich anhand einer Vorgehensmodelldefinition beziehungsweise eines Metamodells sind möglich. Vergleiche dieser Art sind in der Literatur zu finden (siehe Abschnitt 1.4 „Vergleich mit inhaltlich verwandten Arbeiten“). Für die Eignungseinstufung von Vorgehensmodellen ist diese Art des Vergleichs und der Abgrenzung aufgrund ihres mangelnden Nutzens für die Problemstellung nur begrenzt sinnvoll. Deshalb wird im EVGM der Vergleich von Vorgehensmodellen auf der Basis des direkten projektspezifischen Nutzens für das jeweilige Softwareentwicklungsprojekt beziehungsweise die Organisation durchgeführt. Dieser zeigt sich im adressieren der Projektmerkmale mit erfolgsgefährdendem Zustand.
Grenzen. Der Nutzen wird durch das Wissen der Benutzer über das Projekt begrenzt. Dies stellt speziell in einer Projektvorphase, in der das EVGM gedacht ist, eingesetzt zu werden, ein nicht zu unterschätzendes Risiko dar.

Fokussierung durch Beschränkung der Merkmale und Modellreduktion

Eigenschaft. Klassische, aus der Literatur extrahierte Projekterfolgsfaktoren der Softwareentwicklung werden explizit durch die Projektmerkmale repräsentiert. Durch die Beschränkung der vorgeschlagenen Projektmerkmale auf empirisch belegbare Risikoschwerpunkte fokussiert sich das EVGM. Auch wird während der Modellentwicklung beständig angeregt, das Modell um die am unwesentlichsten eingeschätzten Modellelemente zu reduzieren.
Nutzen. Der Blick der Benutzer des EVGM wird auf die häufig wesentlichen Projektmerkmale gelenkt, die erfolgsentscheidend für ein Projekt sein können. Dadurch wird eine zielgerichtete Untersuchung des eigenen Projektes unterstützt. Das Verfahren bleibt dadurch übersichtlich und beherrschbar, externes und internes Wissen unterliegen einer Priorisierung. Durch seine explizite Orientierung an klassischen und empirisch belegten Risikoschwerpunkten hilft das EVGM seinen Anwendern, sich bei der Auswahl eines Vorgehensmodells auf die aus der Erfahrung dringlichsten und wichtigsten Probleme im Projekt zu fokussieren. Diese Fokussierung hilft auch die Größe des Modells beherrschbar zu halten. Das EVGM unterstützt seine Benutzer, sich die wesentlichen Fragestellungen bewußt zu machen, sie zu untersuchen, signifikante Projektmerkmale zu finden und diese zueinander in Beziehung zu setzen.
Grenzen. Der Nutzen der Priorisierung externen und internen Wissens wird begrenzt durch den Stand des Fachwissens, sowie die subjektiven Einschätzungen der Benutzer. Die vorgefertigten Modellelemente unterstützen nicht das selbständige Denken der Benutzer.

Erweiterbarkeit des EVGM durch modularen Ansatz

Eigenschaft. Das EVGM ist modular aufgebaut. Es trennt die Modellierungssprache von der Modellierungsmethode. Innerhalb der Modellierungsmethode sind Eigenschaf-

ten spezifischer Vorgehensmodelle von allgemeinen Vorgehensmodellmerkmalen separiert.

Nutzen. Durch seinen modularen Ansatz ist das EVGM hinsichtlich seiner Modellelemente und betrachteten Vorgehensmodelle veränder- und erweiterbar. Somit ist es projektspezifisch anpass- und optimierbar. Zu einem späteren Zeitpunkt neu entwickelte Vorgehensmodelle können in das EVGM integriert, Veränderungen an bestehenden Vorgehensmodellen im EVGM aktualisiert werden.

Grenzen. Das EVGM kann nicht unmittelbar auf eine andere Problemstellung als die Eignungseinstufung von Vorgehensmodellen transferiert werden.

Unkontrollierte Erweiterungen können zu ungeeigneten Modellelementen und Inkonsistenz führen. Der Erweiterungsprozess muß durch einen Verantwortlichen moderiert und kontrolliert sein.

Zusammenfassung. Das EVGM kombiniert systematisch externes, kontextunabhängiges Wissen über Software Engineering und Management mit internem spezifischem Wissen über das jeweilige Projekt beziehungsweise die Organisation. Es vereinigt somit allgemein für Softwareprojekte relevante Fakten mit kontextspezifischen Einflußfaktoren. Der Ablauf des EVGM ist vorgegeben. Sein Ergebnis hat heuristischen Charakter, da Verfahren teilweise auf das projektspezifische Wissen und die Ermessensentscheidung seiner Benutzer baut. Das EVGM ist ein anpaßbarer Ansatz. Es erfordert die aktive und kompetente Beteiligung seiner Benutzer.

5.2 Methodisch verwandte Arbeiten

Die Modellierungsmethode im EVGM orientiert sich an der Methode von Gomez und Probst [Gom99], welche auf dem System Dynamics Ansatz von Jay Forrester [For80] aufbaut. Beide Ansätze werden nachfolgend kurz skizziert. Weitere Ansätze, mit System Dynamics Probleme zu modellieren, die der Themenstellung der vorliegenden Arbeit ähnlich sind, werden kurz vorgestellt.

5.2.1 System Dynamics von Forrester

In der vorliegenden Arbeit wurde die Grundidee der Ursache-Wirkungs-Diagramme von Jay W. Forrester benutzt. Er stellt in [For80] eine Notation zur Beschreibung von komplexen Systemen vor. Forrester wendet sie im Rahmen von „Industrial Dynamics“ auf industrielle Produktion, in „Urban Dynamics“ auf geografische Regionen und in „World Dynamics“ auf globale Wirkungsbeziehungen an. Für die Modellierung von Ursachen und Wirkungen wurde sie verschiedentlich angepasst und vereinfacht, [Sen94] ist ein bekanntes Beispiel hierfür aus dem Organisations- und Managementbereich. Auch Gomez und Probst haben den Ansatz von Forrester für ihre Zwecke adaptiert.

5.2.2 Die Methode von Gomez und Probst

In [Gom99] wird eine problemunabhängig einsetzbare Methode für ganzheitliches Problemlösen beschrieben. Sie basiert auf dem System Dynamics Ansatz von Forrester [For80]. Da sie als Anregung für die Methode des EVGM verwendet wurde, wird sie im Folgenden kurz dargestellt.

Verschiedene Sichten ermitteln. Es werden zunächst unterschiedliche Sichtweisen verschiedener Interessengruppen auf die Problemstellung ermittelt. Dies geschieht, um das zu lösende Problem besser zu durchdringen.

Interessen ermitteln. Dann wird versucht, das vorliegende Problem einzugrenzen.

Hierbei werden die Interessen dieser Interessengruppen ermittelt und aus diesen zentrale Einflußfaktoren abgeleitet.

Zentralen Regelkreis aufstellen. Dann wird versucht, mit diesen Einflußfaktoren einen zentralen Regelkreis in einem Ursache-Wirkungs-Diagramm zu modellieren. Hierauf werden die weiteren Einflußfaktoren zu diesem Regelkreis hinzu modelliert.

Nicht beeinflußbare Bereiche ermitteln. Dann werden die nicht beeinflußbaren Einflußfaktoren ermittelt.

Lösungen ansetzen. Hierauf wird versucht, Lösungsoptionen anzusetzen, indem weitere Einflußfaktoren zum Ursache-Wirkungsdiagramm hinzu modelliert und bewertet werden.

Abgrenzung der Methode von Gomez und Probst zum EVGM

Inhaltliche Unterstützung. Gomez und Probst beziehen sich nicht auf eine dezidierte Problemstellung wie das EVGM und können daher ausschließlich methodische, aber keine inhaltliche Unterstützung bieten. Sie können keine Merkmale und deren Beeinflussungsbeziehungen vorschlagen, da diese von der spezifischen Problemstellung abhängen. Deren Ermittlung bleibt dem Benutzer überlassen.

Da das EVGM spezifisch für die Eignungseinstufung von Vorgehensmodellen entwickelt ist, können hier im Gegensatz zu Gomez und Probst inhaltliche Unterstützungen in Form von vorgeschlagenen Merkmalen und Beeinflussungsbeziehungen geboten werden.

Handeln auf Unternehmensebene. Gomez und Probst behandeln über die Modellierungsebene auch das konkrete Agieren im Unternehmen, in dem die Methode von Gomez und Probst angewendet wird.

Das Veränderungsmanagement wird im EVGM abgegrenzt, um die Fokussierung auf die zentrale Problemstellung der Eignungseinstufung von Vorgehensmodellen nicht zu verwischen.

Zustandsrepräsentation. Gomez und Probst versehen ihre Einflußfaktoren nicht mit Zuständen.

Die Merkmale im EVGM sind zustandsbehaftet, damit sie bewertet und verglichen werden können. Zustandsveränderungen können ausgedrückt werden.

Vergleich von Lösungsoptionen im Modell. Gomez und Probst vergleichen erarbeitete Lösungsoptionen nicht direkt im Ursache-Wirkungs-Diagramm. Sie beschreiben keinen Maßstab, mit dem ein nach Interventionen entstehendes Zielsystem anhand seines Ursache-Wirkungs-Diagramms zu bewerten ist.

Im Gegensatz dazu werden im EVGM die durch den Einsatz verschiedener Vorgehensmodelle gegebenen Lösungsoptionen anhand von deren unterschiedlichen Ursache-Wirkungs-Diagramme verglichen. Diese werden hinsichtlich ihres Gesamtzustandes mit Ashby's „Law of requisite Variety“ als kontextunabhängige Meßlatte bewertet.

Regelkreis-Ansatz. Gomez und Probst fordern als Modellierungsbasis immer einen zentralen Regelkreis. Dieser Modellierungsansatz stellte sich bei der Entwicklung des EVGM als zu schematisch und daher nicht zielführend heraus. Der regelkreisbetonte Modellierungsansatz ist bei Prozessen mit vielen Durchlaufzyklen sinnvoll, was beim EVGM nicht gegeben ist, da nur eine Ursache-Wirkungs-Sequenz betrachtet wird. Er wurde daher nicht übernommen.

Größenbeschränkung. Gomez und Probst beschränken nicht die Größe der Ursache-Wirkungs-Diagramme.

Das EVGM forciert die Beschränkung auf die wesentlichen Merkmale der zu modellie-

renden Problemstellung, um die Diagramme übersichtlich und handhabbar zu behalten.

Nicht beeinflussbare Bereiche. Gomez und Probst kennzeichnen nicht lenkbare Einflußfaktoren in den Ursache-Wirkungs-Diagrammen explizit in ihrer Notation.
Im EVGM wird die Beeinflussbarkeit von Merkmalen durch die Einschätzung der Benutzer ermittelt und über die von ihnen vergebenen Zustandsbewertungen implizit ausgedrückt. Dies wird als hinreichend angesehen, da die Existenz signifikanter nicht beeinflussbarer Bereiche ein Vorhaben zu Bewertung und Einführung eines Vorgehensmodells prinzipiell in Frage stellt.

5.2.3 Weitere Ansätze mit System Dynamics

System Dynamics wurde für Problemstellungen aus dem thematischen Umfeld von Softwareentwicklungsprozessen und Entscheidungsunterstützung bereits mehrfach eingesetzt, wie nachfolgende Beispiele belegen.
[Ped04] benutzt „causal maps“, um Wissen über Zusammenhänge in Softwareentwicklungsprojekten zu dokumentieren. Dies ist ein ähnlicher Modellierungsansatz wie im EVGM, dessen Einflußfaktoren und deren Beziehungen aber nicht systematisiert und hergeleitet sind.
[Roe+00] benutzen System Dynamics wie auch das EVGM mit Hinblick auf Entscheidungsunterstützung. Sie betrachten allerdings das Thema Outsourcing.
[Hen00] simuliert mit System Dynamics Abläufe von Softwareentwicklungsprojekten, leitet aber seine Einflußfaktoren und Beziehungen nicht systematisch her.
[Mad00] modelliert Abläufe von Softwareentwicklungsprozessen. Auch seine Einflußfaktoren und Beziehungen sind nicht systematisch hergeleitet.
[Bar+02] erstellt mit System Dynamics Metamodelle von Softwareentwicklungsprozessen.
[Lee04] modelliert Einflußfaktoren in Softwareentwicklungsprojekten mit System Dynamics. Auch seine Einflußfaktoren und Beziehungen sind nicht systematisiert und hergeleitet.

5.2.4 Abgrenzung von Ansätzen mit anderer Methodik

- **Nutzwertanalyse** [Lit95] leistet für die Problemstellung nicht dasselbe wie das EVGM, weil sie von gewichteten, aber voneinander unabhängigen Kriterien ausgeht. Diese Unabhängigkeit ist im EVGM-Umfeld aufgrund der dynamischen Wechselwirkungen zwischen den Merkmalen nicht gegeben.
- **Differentialgleichungssysteme** benötigen als Voraussetzung eine Grundlage, auf der Rechenoperationen für reelle Zahlen angewendet werden können. Diese Genauigkeit ist bei der Eignungseinstufung von Vorgehensmodellen jedoch nicht gegeben. Beispielsweise entzieht die Notwendigkeit subjektiver menschlicher Entscheidungen und Bewertungen genaueren Berechnungen die Grundlage.
- **Bayesianische Netze** [Rus03] basieren auf quantifizierten Wahrscheinlichkeiten für die Pfade durch ihr Netz. Für diese Wahrscheinlichkeiten existiert bei der Eignungseinstufung von Vorgehensmodellen keine Grundlage.
- **Neuronale Netze** [Rey08] benötigen empirische Trainingsdaten. Diese stehen für die Problemstellung der Eignungseinstufung von Vorgehensmodellen nicht zur Verfügung.

5.3 Die Auswahl der Vorgehensmodelle

Es ist in einer Arbeit wie der vorliegenden sinnvoll, sich auf eine begrenzte Auswahl von Vorgehensmodellen zu fokussieren, um diese fundiert analysieren zu können. Hierfür wurden die folgenden, in der Literatur verfügbaren Vorgehensmodelle und vorgehensmodellartigen Ansätze untersucht. Entsprechend der weiter unten dargestellten Kriterien wurde eine Auswahl aus ihnen vorgenommen.
Adaptive Software Development (ASD) [Hig00], [Hig02], Agile Modelling [Amb02], Catalysis [DSo01], Cleanroom [Pro+99], [Dye92], Crystal clear [Coc05], Dynamic System Development Method (DSDM) [Sta98], [Sta03] auch "Dynamic Solution Delivery Model", EVO [Gil88], Extreme Programming (XP) [Bec00], Feature-based Programming [Ric03], Feature-Driven Development (FDD) [Hig02], [Pal02], Microsoft Solutions Framework [Han04], [Pow+04], Objektorientierte Software-Konstruktion [Hor93], Lean Development (LD) [Pop03], [Hig02], [Boe04], [Lar04], Object Engineering Process (OEP) [OEP06], Object Modelling Technique (OMT) [Rum+93], Object Oriented Design (OOD) [Boo94], Object Oriented Process, Environment and Notation (OPEN) [Gra97], [Hen+98], Personal Software Process (PSP) [Hum97], Rational Unified Process (RUP) [Kru99], Real Time Object Oriented Method (ROOM) [Sel+94], Scrum [Sch02], Seamless Process for Efficient and Economic Development (SPEED) [BMS06], Select Perspective [All98], Team Software Process (TSP) [Hum00], [Hum02], [Hum04], V-Modell XT [VMX06] wurden untersucht.

Kriterien für die Berücksichtigung von Vorgehensmodellen im EVGM. Eine fundierte Arbeit muß fokussiert sein, um ausgewählte Aspekte gründlich behandeln zu können. Deshalb wurden vom Autor der vorliegenden Arbeit aus der Menge der am Markt verfügbaren Vorgehensmodelle eine Auswahl getroffen. Für die Berücksichtigung der Vorgehensmodelle wurden mehrere Auswahlkriterien gewählt:

- **Unabhängige empirische Erkenntnisse** über den Projekteinsatz eines Vorgehensmodells müssen in Form von hinreichend vielen und geeigneten Erfahrungsberichten verfügbar sein. Hierbei wird Erfahrungsberichten aus der Industrie der Vorzug vor universitären Reporten gegeben. Industrielle Projekte liefern realistischere und somit für die vorliegende Arbeit relevantere Aussagen, weil ihre Zielsetzung Softwareentwicklung und nicht Erkenntnisgewinn ist. Die Laborsituation, in der universitäre Erfahrungsberichte entstehen, ist nur bedingt auf die Praxis übertragbar. Industrielle Erfahrungsberichte können im Gegensatz zu universitären auch als Indiz für die Verbreitung von Vorgehensmodellen in der kommerziellen Praxis gewertet werden. Durch sie können die Standpunkte der Originalautoren und des Autors der vorliegenden Arbeit untermauert werden. Die Aussagen der vorliegenden Arbeit können auf eine breitere Basis gestellt und durch die Meinung unabhängiger Dritter objektiviert werden.
  Existierende Erfahrungsberichte aus der Industrie sind auch ein Indiz für die tatsächliche Nutzung von Vorgehensmodellen in der Praxis. Sie zeigt eine zielgruppengerechte Relevanz, Repräsentativität und Benutzbarkeit eines Vorgehensmodells. Für die Eignungseinstufung von Vorgehensmodellen stehen praxisrelevante und in der Praxis verwendete Vorgehensmodelle im Vordergrund.
  Projektbetreiber müssen sich die Vorgehensmodellbeschreibung zugänglich machen können. Daher werden keine firmenspezifischen Vorgehensmodelle berücksichtigt.
- Ein Vorgehensmodell muß aktuell gepflegt sein, um zu gewährleisten, daß es sich **auf dem derzeitigen Stand der Softwareentwicklung** befindet. Dies wird vom Au-

tor der vorliegenden Arbeit als Voraussetzung für die Berücksichtigung von Vorgehensmodellen im EVGM betrachtet.

- Ein Vorgehensmodell muß **explizit Teams adressieren** und dies über Rollen ausdrücken. Ohne diesen Aspekt wird es vom Autor der vorliegenden Arbeit als nicht geeignet für in der Praxis realistische und somit und für diese Arbeit relevante Softwareprojekte angesehen, weil diese nicht von Einzelpersonen durchgeführt werden.
- Die Vorgehensmodellelemente wie Aktivitäten, Rollen, Arbeitsprodukte müssen in der Vorgehensmodellbeschreibung **in Zusammenhang** gesetzt sein. Eine lose Sammlung von diesen Elementen wird vom Autor der vorliegenden Arbeit als für die Zielgruppe des EVGM nicht geeignet angesehen, weil sie nicht unmittelbar anwendbar ist. Es ist keine Anforderung an die Zielgruppe des EVGM, aus einer Sammlung von Prozeßbestandteilen ohne weiteres ein Vorgehensmodell spezifisch für ein Projekt entwickeln zu können.
  Ein Vorgehensmodell im EVGM muß mehr von der Softwareentwicklung beschreiben als Modellierungstechnik und deren Notation. Management-Aspekte der Softwareentwicklung müssen beschrieben sein.

Bewertung der im Rahmen der Arbeit untersuchten Vorgehensmodelle. Die Bewertung der oben genannten Vorgehensmodelle nach den eben beschriebenen Kriterien wird in folgender Tabelle 14 dargestellt:

Abgrenzungskriterium *Vorgehensmodell*	*Genügend unabhängige Erfahrungsberichte verfügbar*	*Genügend aktuell und gepflegt*	*Genügende Berücksichtigung Teamaspekt*	*Genügende Beschreibung Zusammenhang Modellelemente*
Adaptive Software Development (ASD)	N	J	N	N
Agile Modelling	N	N	N	N
Catalysis	N	J	N	N
Cleanroom	J	J	N	J
Crystal clear	N	J	J	N
Dynamic Systems Development Method (DSDM)	N	J	N	N
EVO	N	N	N	N
Extreme Programming (XP)	J	J	J	J
Feature-based Programming	N	J	N	N
Feature-Driven Development (FDD)	N	N	J	J
Microsoft Solutions Framework (MSF)	N	J	J	J
Objektorientierte Software-Konstruktion	N	N	N	J
Lean Development (LD)	N	N	N	N
Object Engineering Process (OEP)	N	J	J	J

Abgrenzungskriterium *Vorgehensmodell*	*Genügend unabhängige Erfahrungsberichte verfügbar*	*Genügend aktuell und gepflegt*	*Genügende Berücksichtigung Teamaspekt*	*Genügende Beschreibung Zusammenhang Modellelemente*
Object Modelling Technique (OMT)	J	N	N	N
Object Oriented Design (OOD)	N	N	J	N
Object Oriented Process, Environment and Notation (OPEN)	N	N	N	N
Personal Software Process (PSP)	J	N	N	J
Rational Unified Process (RUP)	J	J	J	J
Real Time Object Oriented Method	N	N	N	J
Scrum	J	J	J	J
Seamless Process for Efficient and Economic Development (SPEED)	N	J	J	J
Select Perspective	N	N	J	J
Team Software Process (TSP)	N	J	J	J
V-Modell XT	J	J	J	J

Tabelle 14 Bewertung der in der vorliegenden Arbeit untersuchte Vorgehensmodelle

Bei der Auswertung der Bewertungskriterien fällt auf, daß „Genügend unabhängige Erfahrungsberichte verfügbar“ am stärksten selektierend wirkt. Das Fehlen von unabhängigen Erfahrungsberichten spricht nicht dafür, daß Vorgehensmodelle in der Praxis der Softwareentwicklungsprojekte systematisch und auf breiter Basis eingesetzt werden. Der Praxiseinsatz von Scrum und Extreme Programming wird auch von [Abr+03] bestätigt, für das V-Modell XT liegt er auf der Hand, da dieses für Entwicklungen von IT-Systemen des Bundes verpflichtend anzuwenden ist. Auch der Praxiseinsatz des Rational Unified Process wird in den Experteninterviews belegt (siehe auch Kapitel 1.2).

Die anderen Vorgehensmodelle aus der Tabelle 14 verletzten gemäß der Untersuchung des Autors der vorliegenden Arbeit ein oder mehrere Bewertungskriterien und werden im EVGM nicht berücksichtigt. Sie stellen ein Potential für benutzergetriebene Erweiterungen des EVGM dar. Die Bewertung der Kriterien in der obigen Tabelle 14 stellen keine über den Geltungsbereich dieser Arbeit hinausgehende Wertung für die betrachteten Vorgehensmodelle dar.
In der vorliegenden Arbeit werden folgende Vorgehensmodelle berücksichtigt, die den obigen Kriterien insgesamt am besten entsprachen:

- **Extreme Programming (XP)**
- **Rational Unified Process (RUP)**
- **Scrum**
- **V-Modell XT**

5.4 Überdeckung des Vorgehensmodellspektrums

Für die ausgewählten Vorgehensmodelle wird im folgenden veranschaulicht, wie sie einerseits das Spektrum von Vorgehensmodellen überdecken und sich anderseits gegeneinander abgrenzen. Somit wird gezeigt, daß sie eine geeignete Auswahl darstellen. Hierzu wird untersucht, was die vorwiegenden Treiber von Vorgehensmodellen sind, sowie welche Themenorientierung an ihnen zu beobachten ist.

Treiber. Es wird in der vorliegenden Arbeit zwischen vorwiegend prozeßgetriebenen und vorwiegend menschengetriebenen Ansätzen unterschieden.
Bei **prozeßgetriebenen Ansätzen** ist ein definierter Entwicklungsprozeß, etwa in Form eines Vorgehensmodells maßgeblich für den Verlauf der Entwicklung und begrenzt menschliche Entscheidungsspielräume. Sie orientieren sich eher am Stand des Wissens bei der Darstellung von Prozessen wie bei [Chr92] und sind strukturiert dargestellt, was durch ihren Beschreibungsumfang erforderlich wird (siehe [VMX06] beziehungsweise[RUP06]).
Bei **menschengetriebenen Ansätzen** ist der Verlauf der Entwicklung maßgeblich durch menschliche Entscheidungen getrieben [Sch02], die Festlegungen durch Entwicklungsprozesse sind dabei eher im Hintergrund. Sie sind weniger konform zu Standards wie [Chr92] und dessen Strukturen, obwohl ihr Beschreibungsumfang mitunter nicht geringer ist wie beispielsweise bei Extreme Programming [Bec00], das insgesamt auf vielen hundert Buchseiten beschrieben wird. Da ihre Modellelemente mitunter sehr generisch dargestellt sind, ist deren Einordnung in eine Systematik oftmals stärker vom Ermessen des Betrachters abhängig als bei prozeßgetriebenen Ansätzen.

Hieraus lassen sich folgende Überlegungen grundlegenderer Natur weiterführen (siehe auch Glossar im Anhang):
Kompliziertheit entsteht, wenn viele Einflüsse und Abhängigkeiten zu berücksichtigen sind. Ein Beispiel sind viele, voneinander abhängige fachliche und technische Anforderungen.
Unsicherheit entsteht, wenn Instabilität und Intransparenz in Bezug auf Einflüsse und Abhängigkeiten bestehen. Ein Beispiel hierfür ist, wenn Anforderungen in großem Umfang unbekannt oder vage sind, beziehungsweise sich oft ändern.
Komplexität entsteht, wenn Kompliziertheit und Unsicherheit bestehen. Im Beispiel von oben wären sehr viele Anforderungen und Abhängigkeiten nicht bekannt beziehungsweise würden sich ständig ändern.
Prozeßgetriebene Ansätze adressieren Kompliziertheit mit umfassenden Aktivitäts- und Arbeitsproduktmodellen, sind aber weniger geeignet, Unsicherheit zu bewältigen. Sie erfordern weniger Detailinformationen bezüglich des Projektes im Gedächtnis der Projektmitarbeiter.
Menschengetriebene Ansätze adressieren Unsicherheit mit flexiblen Rollen, Entscheidungsspielräumen und Prinzipien, sind aber weniger geeignet, Kompliziertheit zu bewältigen. Sie erfordern mehr Detailinformationen bezüglich des Projektes im Gedächtnis der Projektmitarbeiter.
Da die gleichzeitige Bewältigung von Kompliziertheit und Unsicherheit widersprüchliche Ziele sind, folgt hieraus, daß auch Komplexität nur sehr begrenzt beherrschbar ist. Ein Projekt mit sehr vielen Anforderungen und Abhängigkeiten, welche sich fortwährend ändern, kann folglich durch kein existierendes Vorgehensmodell zum Erfolg geführt werden.

Themenorientierung. Es wird in der vorliegenden Arbeit bei Vorgehensmodellen zwischen engineering-orientierten und management-orientierten Ansätzen unterschieden.
Engineering-orientierte Ansätze betrachten vorwiegend die fachlich-technischen Aspekte eines Softwareentwicklungsprojektes (siehe auch Glossar).
Management-orientierte Ansätze sehen wirtschaftliche und organisatorische Facetten im Vordergrund (siehe auch Glossar).
Die im EVGM berücksichtigten Vorgehensmodelle enthalten alle diese Punkte, allerdings mit unterschiedlichen Schwerpunktbildungen, wie die folgende Tabelle 15 verdeutlicht.

Treiber ***Themenorientierung***	***prozeßgetrieben***	***menschengetrieben***
engineering-orientiert	Rational Unified Process	Extreme Programming
management-orientiert	V-Modell XT	Scrum

Tabelle 15 Überdeckung des Spektrums durch die berücksichtigten Vorgehensmodelle

Die Tabelle 15 veranschaulicht Schwerpunkte und Tendenzen innerhalb der genannten Vorgehensmodelle. Diese stehen aber nicht repräsentativ für Vorgehensmodellklassen. Eine Klassenbildung für Vorgehensmodelle ist aufgrund von deren starken Diversifizierungen in vielen Facetten nur begrenzt möglich. Mit Hinblick auf die Problemstellung und die Zielgruppe der vorliegenden Arbeit wäre ein Klassenbildung auch nicht zielführend. Die Tabelle 15 verdeutlicht die Überdeckung des Vorgehensmodellspektrums durch die im Rahmen der vorliegenden Arbeit getroffenen Vorgehensmodellauswahl. Durch diese Herangehensweise bei der Auswahl stehen im EVGM Vorgehensmodelle zur Verfügung, die einerseits repräsentativ das Spektrum überdecken und sich andererseits genügend gegeneinander abgrenzen.

5.5 Prinzipien des EVGM

Das EVGM stützt sich auf die folgenden Prinzipien, die seine Anwender bei ihren Überlegungen und Entscheidungen zu Rate ziehen können.

Ashby's Law of requisite Variety: Vollständige Betrachtung

Definition: Das Gesetz von Ashby [Ash63] liegt dem Ansatz im EVGM zu Grunde. Es besagt, daß in einer komplexen Problemstellung mit voneinander abhängigen Einflußfaktoren alle identifizierten problematischen Größen hinreichend durch Lösungsansätze adressiert werden müssen.
Motivation: Wenn nicht alle erfolgsgefährdenden Einflußfaktoren eines Problems berücksichtigt werden, verbessert sich der Zustand mancher, während sich der Zustand anderer verschlechtert, wenn zwischen diesen entsprechende Beeinflussungsbeziehungen bestehen. Vernachlässigte erfolgsbestimmende Einflussfaktoren können in jedem Fall den Gesamtzustand des Systems in einem erfolgsgefährdenden Zustand halten und ein Projekt zum Scheitern bringen. Durch eine umfassende Berücksichtigung aller Projektmerkmale in projektgefährdendem Zustand entsteht ein balanciertes Wirkungsgefüge, in dem alle Merkmale, deren Zustand als gefährdend für den Projekterfolg bewertet werden, durch Einflüsse von Merkmalen in projektförderlichem Zustand kompensiert werden. Zusätzlich wird vermieden, Probleme durch kurzsichtige Maßnahmen nur zu verschieben. Zu bedenken ist hierbei weiterhin, daß ein Lösungsansatz für manche Merk-

male zustandsverbessernd, für andere aber zustandsverschlechternd wirken kann. Um dies im Überblick zu behalten, ist eine vernetzte Modellierung erforderlich, wie sie mit den Ursache-Wirkungsdiagrammen im EVGM angeboten wird.

Beispiel: Wenn in einem Projekt entsprechend des magischen Projektmanagement Dreiecks [Bur95] Termin- und Qualitätsanforderungen, sowie Budgetrestriktionen bestehen, kann ein Ansatz, der einen dieser drei Aspekte komplett vernachlässigt, nicht zum Erfolg führen.

Einflüsse durch Fähigkeiten von Menschen und durch Vorgehensmodelle ergänzen sich: Geeigneter Potentialeinsatz

Definition: Im EVGM werden personelle und durch Vorgehensmodelle gegebene Einflußfaktoren als zueinander komplementär aufgefaßt, sich bei der Adressierung von Problemen ergänzen können. Dies ist bei der Darstellung von menschengetriebenen und prozeßgetriebenen Ansätzen von Vorgehensmodellen (siehe Kapitel 5.4) motiviert.

Motivation: Sie ergänzen sich bei der umfassenden Lösung für eine komplexe Problemstellung wenn sie gezielt und aufeinander abgestimmt eingesetzt werden. Es gibt Probleme, die durch menschliche Fähigkeiten besser gelöst werden können und solche, für die Festlegungen eines Vorgehensmodells besser geeignet sind. Durch die Verwendung von menschlichen und Vorgehensmodellmerkmalen können Merkmale in projektgefährdendem Zustand effektiv positiv beeinflusst werden.

Beispiel: Personen bringen Kreativität, Kombinationsvermögen und Intuition ein, während Vorgehensmodelle Wissen um die Projektabwicklung und die Struktur von Arbeitsprodukten speichern.

Einflüsse aus dem Projektumfeld berücksichtigen: Übergreifende Betrachtung

Definition: Das EVGM stützt sich auf Erfolgsfaktoren des Software Process Improvement aus der Literatur, die im Rahmen der Aktivität „Risikomanagement betreiben“, einer vorgegebenen Risikoliste und durch die Anforderungen an die Rollen und deren Besetzung adressiert werden.

Motivation: In der Risikoliste werden die klassischen Risiken bei der Einführung eines ausgewählten Vorgehensmodells betrachtet und Vorschläge für Gegenmaßnahmen angeboten. Es wird unterstützt, frühzeitig ein Bewußtsein für die bekannten Risikofaktoren bei der Softwareprozeßverbesserung und Prozeßveränderung zu wecken und Gegenmaßnahmen zu formulieren.

Zielgerichtete Anleitung durch fundierte Basis: Fokussierte Betrachtung

Definition: Klassische, aus der Literatur extrahierte Projekterfolgsfaktoren der Softwareentwicklung werden explizit durch die Projektmerkmale adressiert.

Motivation: Der Blick der Benutzer des EVGM wird auf die empirisch belegbar häufig erfolgsbestimmenden Merkmale von Softwareprojekten gelenkt. Dadurch wird eine zielgerichtete Analyse der erfolgsbestimmenden Merkmale des zu untersuchenden Projektes unterstützt.

5.6 Verwendung wissenschaftlicher Methoden

Der folgenden Abschnitt zeigt, welche wissenschaftlichen Methoden für die vorliegenden Arbeit eingesetzt wurden und welche Erfahrungen damit gemacht wurden. Darauffolgend wird dargestellt, wie die Arbeit aus Sicht der Wissenschaftstheorie fundiert ist.

5.6.1 Erfahrungen mit wissenschaftlichen Methoden

Die **Literaturrecherche** wurde in der vorliegenden Arbeit vorwiegend dazu verwendet, die Risikoschwerpunkte in Softwareprojekten heraus zu kristallisieren. Sie war auch die

Grundlage für die Aussagen über Vorgehensmodelle, Erfolgsfaktoren für die Prozeßverbesserung, hinsichtlich der im EVGM verwendeten Modellierungstechnik und der Anregungen aus der Systemtheorie. Literaturstudien sind wichtig für die breite Abstützung von Hypothesen in der vorliegenden Arbeit. Literaturbelege stellen aber aus der Sicht des Autors nie Erkenntnisse aus erster Hand dar und sind immer durch die subjektiven Sichtweisen und Erfahrungshintergründe der einzelnen Original-Autoren beeinflußt.

Ein umfassende **empirische Praxiserprobung** des EVGM war für die vorliegende Arbeit nicht möglich. Aufgrund der signifikanten Diversifizierung von Softwareprojekten wäre sie nicht ausreichend aussagekräftig und vergleichbar. Es könnten nie zwei Vorgehensmodell-Alternativen an Projekten unter identischen Bedingungen verwendet und darauf basierend Projektverlauf und -Ergebnis verglichen werden. Der direkte Zugang zu repräsentativen und vergleichbaren Erfahrungsdaten aus der Industrie war nicht ausreichend gegeben. Aufgrund der nicht vermeidbaren Subjektivität von Erfahrungsberichten wäre dieser Ansatz mit den selben Schwierigkeiten behaftet wie die Herangehensweise über Literaturbelege.
Die Durchführung von Fallstudien, basierend auf realen Projekten aus dem Umfeld der Softwareentwicklung für Mobiltelefone belegten die praktische Relevanz und die Problemabdeckung der Fragen zu Einflußfaktoren aus dem Projektumfeld, Merkmale und Beeinflussungsbeziehungen, die das EVGM vorschlägt [Olf06]. Sie erfolgte durch einen Softwarearchitekten mit mehrjähriger Projekterfahrung. Auch die Eignung der Darstellung des EVGM für seine Zielgruppe wurde durch die Fallstudien untermauert. Sie belegte auch die Notwendigkeit der Eigenschaft des EVGM, an projektspezifische Eigenheiten anpassbar zu sein. Die Anforderungen an die Fähigkeiten der Benutzer des EVGM (siehe Kapitel 4.5 „Rollen im EVGM“) konnten durch die Fallstudien ebenfalls bestätigt werden.

Die vorliegende Arbeit stützt sich bei der Plausibilisierung von Merkmalen und deren Beeinflussungsbeziehungen auf ein **rationalistisches Modell** [Sch06], untermauert und validiert durch die Erfahrungen aus der mehrjährigen industriellen Tätigkeit in der Softwareentwicklung und der Softwareentwicklungsprozeß-Beratung des Autors. Dies impliziert einen Anteil an subjektiver Einschätzung in der vorliegenden Arbeit.

5.6.2 Wissenschaftstheoretische Fundierung

Dieser Abschnitt stellt die in der vorliegenden Arbeit verwendeten wissenschaftlichen Vorgehensweisen explizit dar und macht sie auf diese Weise besser nachvollziehbar und überprüfbar.
Innerhalb der vorliegenden Arbeit werden Aussagen bezüglich von Softwareentwicklungsprojekten, sowie der geeigneten Verwendung von Vorgehensmodellen vorgelegt, sowie Schlüsse gezogen. Um diese und die Art ihres Zustandekommens zu überprüfen, wurden Ansätze aus der Wissenschaftstheorie herangezogen. Hierbei wurden insbesondere der Rationalismus in Zusammenhang mit der Deduktion, sowie der Empirismus in Zusammenhang mit der Induktion angewendet.

5.6.2.1 Eigenschaften, Vorteile und Nachteile des Empirismus

Eigenschaften. Der Empirismus basiert auf wahrnehmbaren Phänomenen und verwendet Beobachtungsaussagen. Aus diesen werden induktiv allgemeinere Aussagen abgeleitet [Sei96], [Sch06].

Beispiel: In mehreren Projekten haben instabile Anforderungen Verschiebungen des Liefertermins bewirkt.
Folgerung: In Projekten bewirken instabile Anforderungen allgemein Terminverschiebungen. Innerhalb des Empirismus gezogene induktive Schlüsse sind endgültig widerlegbar (falsifizierbar). Wenn in Fortführung des Beispiels von oben in einem Projekt instabile Anforderungen keine Terminverschiebung bewirkt haben, gilt der induktive Schluß als falsifiziert. Induktive Schlüsse sind nicht endgültig hinsichtlich ihrer Korrektheit (Validität) überprüfbar (verifizierbar) [Sei96]. Auch bei unendlich vielen Projekten, in denen instabile Anforderungen Terminverschiebungen bewirkten, kann nicht ausgeschlossen werden, daß ein Projekt möglich ist, in dem diese Auswirkung nicht zutrifft. Der Sachverhalt, daß eine Universalaussage nicht endgültig belegbar, aber endgültig widerlegbar ist, wird auch als einseitige Falsifizierbarkeit bezeichnet [Pop94], [Cha01].
Vorteile. Der Empirismus kann durch induktive Schlüsse neue Sätze erzeugen.
Nachteile. Der Empirismus kann durch endlich viele Beobachtungsaussagen nicht auf verifizierte oder validierte Universalaussagen führen (s.o.) [Cha99]. Beobachtungsaussagen und ihre Interpretation sind nicht objektiv und daher fehlerbehaftet [Cha90], [Ch01], [Kuh89].

5.6.2.2 Eigenschaften, Vorteile und Nachteile des Rationalismus

Eigenschaften. Der Rationalismus basiert auf logischen Aussagen, innerhalb derer mit Hilfe der Deduktion von allgemeineren auf speziellere Aussagen geschlossen wird [Sch06], [Sei96]. Diese logischen Aussagen bilden ein konsistentes gedankliches Modell aus abstrakten Begriffen [Spe80].
Beispiel: Requirements Management wird in der vorliegenden Arbeit als Teil des Software Managements definiert. Im Vorgehensmodell Y wird Requirements Management beschrieben. Folgerung: Das Requirements Management im Vorgehensmodell Y ist Teil des Software Managements.
Vorteile. Innerhalb eines rationalistischen Modells kann Konsistenz verifiziert werden.
Nachteile. Eine logische Aussage eines rationalistisches Modells kann nicht endgültig durch Beobachtungsaussagen hinsichtlich seiner Validität widerlegt werden, weil man die Definition seiner Abstraktionen immer entsprechend anpassen kann [Kuh89].

5.6.2.3 Verwendung wissenschaftstheoretischer Ansätze in Kapitel 2

Im Kapitel 2 „Charakterisierung von Projekten“ werden empirische Studien bezüglich Projekterfolgsfaktoren und Projektrisikofaktoren verwendet. Diese Studien repräsentieren endlich viele Beobachtungsaussagen der Teilnehmer der Studien. Dies kommt einer deskriptiven, statistischen Betrachtungsweise gleich, weil weder Begründungen für die Fragen, noch für die Beantwortung der Fragen angegeben werden.
Aufgrund ihrer geschlossenen Fragen konnte jede der zitierten Studien nur die Hypothesen ihrer Initiatoren in Bezug auf wichtige Risikofaktoren der Softwareentwicklung widerspiegeln (vergleiche [Sch06]) und diese gegebenenfalls bestätigen [Leh01]. Die Hypothesen als solche werden somit innerhalb der Studien nicht hinterfragt.
Auch die Antworten der Befragten geben deren subjektive Wahrnehmungen im Sinne des Konstruktivismus [Sch06] wieder, da jeder Befragte aufgrund seines individuellen Erfahrungshintergrundes eine eigene Vorstellung von beispielsweise „instabilen fachlichen Anforderungen“ hat. Deshalb bewirkt ein Mitteln über mehrere Studien eine besse-

re Ausgewogenheit der im Rahmen der vorliegenden Arbeit abgeleiteten Aussagen.
Die Vielzahl empirischer Erkenntnisse über Erfolgs- und Risikofaktoren in Softwareprojekten wird in der vorliegenden Arbeit zu Projektmerkmalen verdichtet. Dieser Schluß von den endlich vielen Beobachtungsaussagen der Studien auf die Projektmerkmale ist induktiv. Er erfolgt im Sinne einer Begriffsanalyse [Ebe87] für Softwareprojekte. Weiterhin werden im Sinne einer Relationsanalyse [Ebe87] Beeinflussungsbeziehungen zwischen den Projektmerkmalen identifiziert. Merkmale und Beeinflussungsbeziehungen bilden zusammen ein logisches Modell. Dieses Modell ist deskriptiv, es macht verallgemeinernde Aussagen über die Realität.
Aus dem induktiven Schluß von Risiko- und -erfolgsfaktoren auf Projektmerkmale läßt sich aber strenggenommen keine Universalaussage ableiten, dies scheitert am sogenannten unendlichen Regress [Pop94].
Deshalb wird dem induktiven Schluß für jedes Projektmerkmal eine rationalistische Begründung beigefügt. Die Projektmerkmale werden durch die Veranschaulichung ihrer Bedeutung für Softwareprojekte und für die Eignungseinstufung von Vorgehensmodellen plausibel gemacht.
Die Beschreibung der Beeinflussungsbeziehungen zwischen den Projektmerkmalen unterstreicht ebenfalls deren Bedeutsamkeit für Softwareprojekte und die Eignungseinstufung von Vorgehensmodellen. Der Zusammenhang zwischen Projektmerkmalen und deren wechselseitige Beeinflussungen werden veranschaulicht.
Diese Argumentation kann nur auf der Basis von Plausibilitätserklärungen erfolgen. Die somit abgeleiteten Projektmerkmale und ihre Beeinflussungsbeziehungen können daher keinen Anspruch auf Allgemeingültigkeit für jedes Softwareprojekt erheben, sie stellen lediglich Tendenzaussagen dar und haben deshalb für die Anwender des EVGM Vorschlagscharakter.
Durch diese rationalen Begründungen der Bedeutung von Projektmerkmalen wird die empiristisch-induktive Argumentation zu ihrer Herleitung rationalistisch untermauert. Diese zweiseitige Absicherung erhöht die Fundierung der in Kapitel 2 getätigten Aussagen.

5.6.2.4 Verwendung wissenschaftstheoretischer Ansätze in Kapitel 3

Im Kapitel 3 „Charakterisierung von Vorgehensmodellen“ werden die Merkmale für die Repräsentation von Vorgehensmodellen beziehungsweise ihrer Bestandteile hergeleitet. Die Vorgehensmodellmerkmale stellen Abstraktionen dar, die Merkmale von Vorgehensmodellen unabhängig von konkreten Vorgehensmodellen repräsentieren. Sie werden über Beeinflussungsbeziehungen zu den Projektmerkmalen plausibel gemacht. Die Vorgehensmodellmerkmale werden durch die Beeinflussungsbeziehungen zu einem rationalistischen Modell verbunden.
Die Vorgehensmodellmerkmale werden empirisch gestützt durch die vorgehensmodellspezifischen Bewertungen, weil diese jeweils deren konkrete Exemplarisierungen in existierenden Vorgehensmodellen darstellen. Die Berücksichtigung von vorgehensmodellübergreifender und vorgehensmodellspezifischer Sicht auf die Vorgehensmodell-Merkmale erhöht ihre Fundiertheit.

5.6.2.5 Verwendung wissenschaftstheoretischer Ansätze in Kapitel 4

Das EVGM-Verfahren in Kapitel 4 „Das Modellierungsverfahren im EVGM“ besteht mit seinen Modellen und Aktivitäten aus abstrakten Begriffen, die miteinander verbunden sind. Es stellt somit ein rationalistisches Modell dar. Da es die Verfahrensweise

während des Eignungseinstufungsprozesses festlegt, besitzt es normativen und prädiktiven Charakter. Das Modell des Projektes, das mit Hilfe des EVGM aufgebaut wird, besteht aus den Abstraktionen, welche die Merkmale und ihre Beeinflussungsbeziehungen darstellen. Es hat daher rationalistischen Charakter. Somit ist es auf Widerspruchsfreiheit überprüfbar.
Seine Validität bezüglich der Realität im spezifischen Projekt wird durch die Möglichkeit verbessert, projektspezifische qualitative Bewertungen dieser Modellelemente, sowie Anpassungen der Modellelemente als solche vornehmen zu können.

6 Zusammenfassung, kritische Betrachtung, Ausblick

„The road goes ever on and on
down from the door where it began
Now far ahead the road has gone
and I must follow if I can
I pursue it with eager feet
until it meets a larger way
where many paths and errands meet
And wither then? I cannot say.“
J.R.R. Tolkien

Dieses Kapitel fasst die Inhalte der vorliegenden Arbeit zusammen und betrachtet kritisch ihre Ergebnisse. Perspektiven zur Verwendung und Weiterentwicklung der Arbeit werden dargestellt.
Die Zusammenfassung wiederholt die Kernaussagen der vorliegenden Arbeit, um diese einprägsamer zu machen. Eine kritische Betrachtung verdeutlicht Nutzen, Grenzen und Beitrag der Arbeit. Sie ist ein Ausgangspunkt für die Perspektiven zur Weiterentwicklung und Verwendung der Arbeit. Diese Perspektiven zeigen Möglichkeiten auf, die Ergebnisse der Arbeit auch über ihre grundsätzliche Intention hinaus zu benutzen und ihre Fortschreibung zielführend weiter zu treiben.

Inhalt des Kapitels

6.1 Zusammenfassung des EVGM

Im EVGM wurden Merkmale zur Beschreibung von Projekten und Vorgehensmodellen hergeleitet, definiert und motiviert. Ihre wechselseitigen Abhängigkeiten wurden durch Beeinflussungsbeziehungen ausgedrückt.
Mit dem EVGM steht Betreibern von Softwareentwicklungsprojekten eine Unterstützung bei der Analyse von Projekten und der Auswahl eines projektspezifisch geeigneten Vorgehensmodells zur Verfügung. Durch ein Schritt-für-Schritt-Verfahren ist es möglich, die komplexe Aufgabenstellung der Evaluation von Vorgehensmodellen für Projekte auch ohne weitreichenden spezifischen Kenntnishintergrund in Bezug auf Softwareentwicklungsprozesse, existierende Vorgehensmodelle und Prozeßverbesserung durchzuführen. Das EVGM schlägt konkrete Vorgehensmodelle und keine Vorgehensmodellkategorien vor. Dies erleichtert den Benutzern die direkte Verwendung der Ergebnisse des EVGM.

6.2 Kritische Betrachtung des EVGM

6.2.1 Nutzen des EVGM für die Industrie

Über die geeignete Auswahl und Anpassung eines Vorgehensmodells werden für die Industrie Softwareprojekte methodisch unterstützt, ihre drängenden Risikopotentiale analysiert, sowie zielgerichtet und priorisiert adressiert. Dadurch ist schlußendlich eine höhere Wirtschaftlichkeit in Softwareprojekten möglich.

Zielgerichtete Unterstützung von Projekten. Industrielle Softwareentwicklungsprojekte können mit dem EVGM bedarfs- und zielgerichteter unterstützt werden. Dies erhöht die Effizienz in Projekten.

Höhere Erfolgswahrscheinlichkeit für Projekte. Mit der Anwendung geeigneter Vorgehensmodelle erhöht sich die Wahrscheinlichkeit eines Projekterfolges. Dies wird durch das gezielte adressieren von Risikopotentialen möglich.

Systematische Erfahrungswerte von Projekten. Die Erfahrungswerte über einen zielgerichteten Einsatz von Vorgehensmodellen können systematisch gesammelt und ausgewertet werden. Somit können diesbezügliche Optimierungen durchgeführt werden.

Verbessertes Bewußtsein für Nutzen von Prozessen. Durch den zielgerichteten Einsatz von Vorgehensmodellen können bestehende Vorurteile in der Praxis über sie entkräftet werden, sie seien zu kompliziert, zu aufwendig anzuwenden, zu abstrakt, irrelevant, zu theoretisch, unfundiert und somit in der Praxis nicht anwendbar. Bestehende Vorbehalte gegen die Benutzung von Vorgehensmodellen können somit reduziert werden. Ein konstruktiveres Bewußtsein der Entwickler für Prozeß-Aspekte kann somit erreicht werden.

6.2.2 Nutzen des EVGM für die Forschung

Für die Forschung können durch das EVGM systematisiertere Erfahrungswerte aus der Industrie verfügbar werden. Eine bessere theoretische Durchdringung der Thematik ermöglicht einen effektiveren Dialog. Dadurch ist ein verbesserter Erkenntnisgewinn möglich.

Systematisierter Erfahrungsaustausch mit der Praxis möglich. Wenn durch den Einsatz des EVGM Vorgehensmodelle zielführender eingesetzt werden, entsteht die Möglichkeit einer qualitativen und quantitativen Steigerung der Rückkopplung aus der Praxis für die Forschung bezüglich der Themen Software Engineering, Softwareprozeßverbesserung und Vorgehensmodelle.

Bessere Durchdringung der Thematik. Das herausarbeiten und darstellen von wesentlichen Projekt- und Vorgehensmodellmerkmalen und deren Wirkungszusammenhängen erhöht die Durchschaubarkeit der Problemstellung von Softwareprojekten. Hierauf können weitere diesbezügliche Forschungsaktivitäten aufgesetzt werden. Die vereinheitlichte Begriffswelt ermöglicht einen effektiveren Dialog und Erfahrungsaustausch zwischen Forschung und Praxis.

6.2.3 Überprüfung des EVGM gegen wichtige Qualitätskriterien

Vollständigkeit. Das EVGM stellt nur eine Schnittstelle zu den Themengebieten der Softwareprozeßverbesserung und des Veränderungsmanagements zur Verfügung. Es muß mit einem Verfahren mit diesen Themenstellungen kombiniert werden, um das ausgewählte Vorgehensmodell auch in die Organisation einführen zu können.

Auch die im EVGM erarbeiteten Merkmale von Projekten und Vorgehensmodellen er-

heben nicht den Anspruch der Vollständigkeit, da nur die empirisch belegbar am häufigsten auftretenden berücksichtigt wurden. Spezifisch relevante Merkmale können im Bedarfsfall ergänzt werden.

Überprüfbarkeit. Diese stellt nach [Pop94] ein Abgrenzungskriterium für Wissenschaftlichkeit dar. Das EVGM ist bedingt überprüfbar. Durch die sehr konkreten Merkmale und Beeinflussungsbeziehungen können die Aussagen gedanklich nachvollzogen werden. Dadurch kann man entscheiden, ob man den dargestellten Sachverhalten und Schlußfolgerungen zustimmt.
Es ist nicht möglich, den Wirkungsgrad des EVGM schlußendlich und empirisch zu überprüfen. Kein Projekt kann zweimal unter identischen Bedingungen mit verschiedenen Vorgehensmodellen durchgeführt werden. Empirische Erfahrungsdaten über Projekte, eingesetzte Vorgehensmodelle und damit verbundene Erfolge oder Mißerfolge sind nicht im hinreichenden Maß zugänglich, um belastbare Schlüsse ableiten zu können.

Durchführbarkeit. Die Anwendung des EVGM setzt gute Kenntnisse über das betreffende Projekt voraus. Allerdings sind nicht alle Projektmerkmale des EVGM am Anfang eines Softwareentwicklungsprojektes zwingend hinreichend bewertbar. Dies kann die Durchführung des EVGM erschweren.

Objektivität. Durch den erforderlichen hohen Definitionsanteil der Arbeit beinhaltet sie viel individuelle Einschätzung und Festlegung des Autors. Dies mindert insgesamt die Objektivität der Arbeit notgedrungen.

Benutzbarkeit. Subjektive, projektspezifische Entscheidungen der Benutzer des EVGM bezüglich der Auswahl, Bewertung und Vernetzung von Merkmalen sind unverzichtbar für die Durchführung des EVGM. Dies erfordert angemessene Kenntnisse der Benutzer über Softwareentwicklungsprojekte, die in der Praxis nicht immer selbstverständlich sind. Der Benutzerkreis und die Benutzbarkeit des EVGM sind somit begrenzt.

Nachvollziehbarkeit. Trotz einer Schritt-für-Schritt-Vorgehensweise kann kein vollständiger Determinismus bei der Durchführung des EVGM angeboten werden, da dieser mehr Fallunterscheidungen enthalten müßte, als bei der Durchführung handhabbar wäre. Der subjektive Ermessensspielraum der EVGM-Benutzer ist unverzichtbar. Dies setzt der Nachvollziehbarkeit der Ergebnisse des EVGM Grenzen.

Repräsentativität. Das EVGM berücksichtigt vier konkrete Vorgehensmodelle. Diese überdecken das Spektrum von Vorgehensmodellen homogen. Obwohl das EVGM leicht um weitere Vorgehensmodelle erweitert werden kann, ist es begrenzt repräsentativ in Bezug auf die existierenden Vorgehensmodelle.

6.3 Wissenschaftlicher Beitrag der Arbeit

In diesem Abschnitt wird für den wissenschaftlichen Beitrag in dieser Arbeit jeweils unter „Definition“ beschrieben, was ihren Inhalt ausmacht, worin ihr Transfer und ihre Neuheit besteht und wie sie erbracht wurde. Unter „Motivation“ wird jeweils verdeutlicht, warum sie nützlich ist und ihre Anwendbarkeit veranschaulicht.

6.3.1 Domänenerschließung

Konkrete Merkmale und Beeinflussungsbeziehungen der Problemstellung extrahiert

Definition. In der vorliegenden Arbeit wurden zentrale und konkrete Merkmale in Softwareentwicklungsprojekten und deren Beeinflussungsbeziehungen analysiert, extrahiert, abgegrenzt und beschrieben. Speziell die Merkmale von Projekten und Vorge-

hensmodellen sind sehr konkret ausgearbeitet. Dies erfolgte durch Extraktion aus der Literatur und durch eigene Argumentation.

Motivation. Dies ist notwendig, um die Merkmale und deren Beeinflussungsbeziehungen überprüfbar und Schlüsse in Bezug auf die Problemstellung nachvollziehbar zu machen, was nach [Pop94] ein wesentliches Abgrenzungskriterium für Wissenschaftlichkeit darstellt. Klare Aussagen über die Zusammenhänge in der Problemdomäne erhöhen die Überprüfbarkeit der Darstellung.

Begriffssystematik über der Domäne entwickelt

Definition. Eine Begriffssystematik über der betrachteten Problemstellung und ihren Merkmalen wurde entwickelt. Eine kompakte und kohärente Darstellungsmöglichkeit wurde geschaffen.

Motivation. Dadurch wird eine präzise Kommunikationsbasis geschaffen. Die Problemstellung kann strukturiert und übersichtlich dargestellt werden. Klare und konkrete Aussagen können über sie getroffen werden, wodurch sie besser verständlich gemacht werden kann. Dadurch können die Aussagen über die Problemstellung besser nachvollzogen werden.

Herausarbeiten von Vorgehensmodell-Merkmalen

Definition. Eine Analyse existierender Vorgehensmodelle und ihrer Wechselwirkungen mit Projekten auf exemplarischer und prinzipieller Ebene wurde durchgeführt. Dies erfolgte anhand der Darstellung von Vorgehensmodell-Merkmalen und deren jeweiligen vorgehensmodellspezifischen Bewertungen.

Motivation. Dadurch können grundsätzliche Wirkweisen von Vorgehensmodell-Merkmalen verdeutlicht und anhand von konkreten Vorgehensmodellen exemplarisch veranschaulicht werden.

Extrahieren von Einsatzkriterien für Vorgehensmodelle

Definition. Kriterien für die Eignung von Vorgehensmodellen wurden analysiert und spezifiziert. Dies erfolgt durch die Zuordnung von Merkmalen, die Vorgehensmodelle beschreiben, zu Merkmalen, die Projekte beschreiben.

Motivation. Dadurch entsteht eine systematische Basis für eine Eignungseinstufung.

Definieren von Schnittstellen zum SPI und Veränderungsmanagement

Definition. Das Verfahren grenzt sich gegen das operative Risikomanagement, Software Prozeß Verbesserung und Veränderungsmanagement (siehe Glossar) ab.

Motivation Dadurch bietet es Schnittstellen, an denen weiterführende Arbeiten ansetzen können.

Umfassendes Definieren von Begriffen

Definition. Die Arbeit hat einen hohen Anteil an Begriffsdefinitionen.

Motivation. Für viele der in der vorliegenden Arbeit verwendeten Begriffe waren in der Fachliteratur keine geeigneten und standardisierten Definitionen etabliert.

6.3.2 Theorietransfer

Transferieren systemtheoretischer Ansätze in die Domäne

Definition. Anregungen aus der allgemeinen Systemtheorie [Wil99], [Wil00], [Wil01], [Kri98], speziell Ashby's „Law of Requisite Variety“ [Ash63] wurden erschlossen, in den Themenkontext der Softwareentwicklungsprojekte und der Eignungseinstufung von Vorgehensmodellen übertragen und an diesen angepasst.

Motivation. Die Kernaussage von Ashby's Gesetz ist, alle erkennbaren Problemaspekte mit dezidierten Lösungsansätzen zu adressieren. In anderen Worten ausgedrückt heißt das, die bestehende Problemkomplexität durch eine angemessene Lösungskomple-

xität aufzuwiegen.
Falls beispielsweise technische, kaufmännische und fachliche Probleme in einem Projekt bestehen, genügt es für einen Projekterfolg nicht, nur einen Teil von ihnen zu beachten, da die hierbei ignorierten Probleme genügen können, das Projekt scheitern zu lassen. Wenn ein Lösungsansatz nur die technischen und fachlichen Probleme angeht, kann das Projekt beispielsweise trotzdem an Ressourcenknappheit scheitern.
Die Anwendung dieses Prinzips auf ein konkretes Projekt ist nicht trivial, da jede am EVGM beteiligte Rolle nur ihren individuellen Blickwinkel und für diesen spezifischen Probleme sehen kann. Das Erkennen der Problemkomplexität sowie der erforderlichen Lösungsansätze erfolgt innerhalb des EVGM durch die jeweiligen spezifischen Blickwinkel der einzelnen beteiligten Rollen, die Varietät ist erst durch die Beteiligung vieler Rollen erkennbar.
Beispielsweise würde ein Manager eher die kaufmännischen, ein Analytiker die fachlichen und ein Entwickler die technischen Probleme sehen. Die Probleme der jeweils anderen Kategorien wären für jede dieser Rollen weniger präsent.
Auch die Beeinflussungen zwischen den Einflußfaktoren repräsentiert Problemkomplexität, die es mit Lösungskomplexität aufzuwiegen gilt.
Beispielsweise wirken sich fachliche und technische Probleme auf den Bedarf an Ressourcen aus und verschärfen so einen bestehenden Engpaß.
Durch die Anwendung von Ashby's Law wird eine ausgewogene, vernetzte und nachhaltige Problemlösung unterstützt. Durch eine Theorie als unabhängige Messlatte gewinnt die Darstellung der Problemdomäne an Fundiertheit.

6.3.3 Notations- und Methodentransfer

Transferieren und projektspezifisches weiterentwickeln von Modellierungsmethode und -notation

Definition. Eine bereits erprobte Modellierungsmethodik aus der Systemtheorie (System Dynamics) [For80], [Kir98] beziehungsweise [Gom99] wurde entsprechend der Problemstellung abgewandelt. Hierbei wurde eine Schritt für Schritt Modellierungsmethode für die Eignungseinstufung von Vorgehensmodellen entwickelt, welche Ashby's Law [Ash63] als Maßstab benutzt.

Motivation. Dadurch steht eine ausdrucksstarke und benutzbare Sprache zur Beschreibung von Problem und Lösung zur Verfügung. Dies erhöht die Verständlichkeit und Benutzbarkeit des EVGM. Eine beherrschbare Vorgehensweise, die vom Problem zur Lösung führt, ist verfügbar.

6.4 Weiterentwicklungspotentiale und Perspektiven des EVGM

Explizite Kombinierung mit Verfahren zur Software Prozeßverbesserung. Eine Kombination mit Verfahren zur Verbesserung von Softwareprozessen und zum Veränderungsmanagement [Dop02], [Dop02a], [Moh98] wird in der vorliegenden Arbeit angedeutet. Um Nutzbarkeit des EVGM im industriellen Umfeld weiter zu erhöhen, ist eine weitere Definition von Schnittstellen zielführend. Anbindung an konkrete Verfahren wie zum Beispiel IDEAL [McF96] können den Nutzen und damit die Akzeptanz des EVGM im industriellen Umfeld verstärken. Auch wer kein Vorgehensmodell einführen möchte, erhält durch das EVGM Hinweise zur bedarfsgerechten projektspezifischen Verbesserung seines bisher benutzten Vorgehensmodells.

Erweiterung um Vorgehensmodelle. Um das Leistungsportfolio des EVGM auszubauen, können weitere Vorgehensmodelle ergänzt werden. Beim Entwurf des EVGM

wurde bewußt eine modulare Struktur gewählt, die eine diesbezügliche Erweiterung erleichtert.

Weitreichende Validierung in der Praxis. Systematische Praxistests in realistischen Industrieprojekten können empirische Belege für den Wirkungsgrad des EVGM, sowie die Fundiertheit seiner Aussagen liefern oder gegebenenfalls diesbezügliche Optimierungspotentiale offenlegen.

7 Quellenverzeichnis

[Abr+03] Pekka Abrahamsson, Juhani Warsta, Mikko T. Siponen, Jussi Ronkainen: New Directions on Agile Methods: A Comparative Analysis, in Proceedings of the 25th International Conference on Software Engineering (ICSE 2003), Portland, Oregon, USA, Rick Kazman, Len Bass, Jan Bosch (Eds.), IEEE Computer Society 2003

[Agi06], http://www.agilemanifesto.org/ am 09.03.2006

[Ale91] Linda C. Alexander, Alan M. Davis: Criteria for Selecting Software Process Models, Proceedings of the 15th Annual International Computer Software and Applications Conference, Tokyo, IEEE Computer Society Press 1991

[Amb02] Scott W. Ambler: Agile Modelling – Effective Practices for eXtreme Programming and the Unified Process, Wiley 2002

[All98] Paul Allen, Stuart Frost: Component-Based Development for Enterprise Systems – Applying The SELECT Perspective, Cambridge University Press 1998

[Ash63] W. Ross Ashby: An Introduction To Cybernetics, Chapman & Hall 1963

[ASP07] Automotive SPICE Process Assessment Modell (PAM) Version 2.3 vom 05.05.2007, www.automotivespice.com, am 10.08.2007

[Bad02] Nathan Baddoo, Tracy Hall: Practitioner Roles in Software Process Improvement: An Analysis using Grid Technique, in Software Process Improvement And Practice, Volume 7, Issue 1, Wiley 2002

[Bal99] Heide Balzert: Lehrbuch der Objektmodellierung, Spektrum Akademischer Verlag 1999

[Bal96] Helmut Balzert: Lehrbuch der Softwaretechnik I – Software-Entwicklung, Spektrum Akademischer Verlag 1996

[Bal98] Helmut Balzert: Lehrbuch der Softwaretechnik II – Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung, Spektrum Akademischer Verlag 1998

[Bar+02] Márcio De Oliveira Barros, Cláudia Maria Lima Werner, Guilherme Horta Travassos: A System Dynamics Metamodel for Software Process Modelling, in Software Process Improvement and Practice, Volume 7 Issue 3-4, Wiley 2002

[Bec00] Kent Beck: Extreme Programming – Das Manifest, Addison-Wesley 2000

[Bec01] Kent Beck, Martin Fowler: Planning Extreme Programming, Addison-Wesley 2001

[Bec04] Kent Beck, Cynthia Andres: Extreme Programming Explained, Addison-Wesley 2004

[Ben06] Gerd Beneken: Mehrere Diskussionen über Vertragsverhältnisse als Einflußfaktoren auf Softwareentwicklungsprojekte, 2006

[Ben06a] http://www.software-kompetenz.de/servlet/is/30282/HSE06_Lieferanten-management.pdf?command=downloadContent&filename=HSE06_Lieferantenmanage-ment.pdf, am 08.03.2007

[Ber04] Stefan Bergström, Lotta Råberg: Adopting the Rational Unified Process – Success with the RUP, Addison-Wesley 2004, www.adoptingrup.com

[Bez05] Konstantin Beznosov, Philippe Kruchten: Software development practices: Towards agile security assurance, in Proceedings of the 2004 workshop on New security paradigms (NSPW '04), ACM Press 2005

[Bir05] Miklós Biró, Péter Fehér: Forces Affecting Offshore Development, in Proceedings of the 12th European Conference on Software Process Improvement (EuroSPI 2005), Ita Richardson, Pekka Abrahamsson, Richard Messnarz (eds), Budapest, Hungary, LNCS 3792, Springer 2005

[Bis06] Hubert Biskup, Ralf Kneuper (Hrsg.): Nutzen und Nutzung von Vorgehensmodellen, 13. Workshop der Fachgruppe WI-VM der Gesellschaft für Informatik, Shaker 2006

[BMS06] Berner & Mattner Systeme: Seamless Process for Efficient and Economic Development, vertrauliche Kopie des Prozeß-Handbuches durch Siefried Hörfarter am 16.02.2006 per e-mail

[Boe90] Barry Boehm, Frank Belz: Experiences with the Spiral Model as a Process Model Generator, in Proceedings of the 5th International Software Process Workshop, IEEE Computer Society Press 1990.

[Boe91] Barry W. Boehm: Software Risk Management: Principles and Practices, IEEE Software, Vol.8, Issue 1 1991

[Boe98] Software Risk Management, CS577, University of California, Center of Software Engineering, 1998

[Boe+00] Barry W.Boehm, Chris Abts, A.Winsor Brown, Sunita Chulanis, Bradfort K.Clark, Ellis Horowitz, Ray Madachy, Donald Reifer, Bert Steece: Software Cost Estimation with COCOMO II, Prentice Hall 2000

[Boe04] Barry Boehm, Richard Turner: Balancing Agility and Discipline – A Guide for the Perplexed, Addison-Wesley 2004

[Boo94] Grady Booch: Objektorientierte Analyse und Design, Addison-Wesley 1994

[Bra05] Keith Braithwaite, Tim Joyce: XP Expanded: Distributed Extreme Programming, Extreme Programming and Agile Processes in Software Engineering, Hubert Baumeister; Michele Marchesi, Mike Holcombe, (Eds.) LNCS 3556 Springer 2005

[Bro95] Frederick P. Brooks Jr.: The Mythical Man-Month, Addison-Wesley Longman, 1995

[Bur95] Manfred Burghardt: Einführung in Projektmanagement, Publicis MCD 1995

[Bus+06] Ralf Buschermöhle, Heike Eekhoff, Bernhard Josko: SUCCESS 2006: SUCCess and failurE of hard- and Software Projects, Eine Studie, die vom Bundesministeri-

um für Bildung und Forschung (BMBF) unter dem Förderkennzeichen FKZ 01 IS C65 gefördert wurde

[Cao+04] Lan Cao, Kannan Mohan, Peng Xu, Balasubramaniam Ramesh: How Extreme does Extreme Programming Have to be? Adapting XP Practices to Large-Scale Projects, in Proceedings of the 37th Hawaii International Conference on System Sciences, IEEE 2004

[Cha99] Alan F. Chalmers: Grenzen der Wissenschaft, Springer 1999

[Cha01] Alan F. Chalmers: Wege der Wissenschaft, Springer 2001

[Cha02] Keith C.C. Chan, Lawrence M.L. Chung: Integrating Process and Project Management for Mulit-Site Software Development, Annals of Software Engineering (14), 2002, S.115-143

[Chr+07] Mary Beth Chrissis, Mike Konrad, Sandy Shrum: CMMI Guidelines for Process Integration and Product Improvement, Second Edition, Addison-Wesley 2007

[Coc05] Alistair Cockburn: Crystal Clear – A Human-Powered Methodology for Small Teams, Addison-Wesley 2005

[Col06] Gerry Coleman, Martin MacAnallen: Managing Challenges of Legacy Systems Using Extreme Programming, Software Process Improvement and Practice, Volume 11 Issue 3, Wiley 2006

[Con04] Kieran Conboy, Brian Fitzgerald: Toward a Conceptual Framework of Agile Methods, in Proceedings of the 4th Conference on Extreme Programming and Agile Methods (XP/Agile Universe 2004), C. Zannier, H. Erdogmus, L. Lindstrom (eds.), Calgary, Canada, LNCS 3134, Springer 2004

[Chr92] Gerhard Chroust: Modelle der Software-Entwicklung, Oldenbourg 1992

[Dal+05] Darren Dalcher, Oddur Benediktsson, Helgi Thorbergsson: Development Life Cycle Management: A Multiproject Experiment, in Proceedings of the 12th IEEE International Conference and Wokshop on the Engineering of Computer-Based Sytems (ECBS'05) IEEE Computer Society Press 2005

[DeM97] Tom DeMarco: Warum ist Software so teuer?, Hanser 1997

[DeM99] Tom DeMarco, Timothy Lister: Wien wartet auf Dich!, Hanser 1999

[DeM01] Tom DeMarco: Spielräume, Hanser 2001

[DeM03] Tom DeMarco, Timothy Lister: Bärentango – Mit Risikomanagement Projekte zum Erfolg führen, Hanser 2003

[Doe89] Dietrich Dörner: Die Logik des Mißlingens – Strategisches Denken in komplexen Situationen, Rowohlt 1989

[Dop02] Klaus Doppler, Christoph Lauterburg: Change Management – Den Unternehemenswandel gestalten, Campus 2002

[Dop+02] Klaus Doppler, Hellmuth Fuhrmann, Birgit Lebbe-Waschke, Bert Voigt: Unternehmenswandel gegen Widerstände, Campus 2002

[Dro+04] Jerry Drobka, David Noftz, Rekha Raghu: Piloting XP on Four Mission-Critical Projects, IEEE Software 2004

[DSo01] Desmond D'Souza, Alan Cameron Wills: Objects, Components and Frameworks with UML – The Catalysis Approach: Addison-Wesley 2001

[DUD91] Dudenredaktion: Der Kleine Duden – Fremdwörterbuch, Bibliografisches Institut und F.A. Brockhaus AG 1991

[Dye92] Michael Dyer: The Cleanroom Approach to Quality Software Development, Wiley 1992

[Ebe87] Kurt Eberhard: Einführung in die Erkenntnis- und Wissenschaftstheorie – Geschichte und Praxis der konkurrierenden Erkenntniswege, Kohlhammer Urban 1987

[Eck04] Jutta Eckstein: Agile Softwareentwicklung im Großen – Ein Eintauchen in die Untiefen Erfolgreicher Projekte, dpunkt.verlag 2004

[Ema+99] Khaled el Emam, Pierfrancesco Fusaro, Bob Smith: Success Factors and Barriers for Software Process Improvement in „Better Software Practice for Business Benefit: Principles and Experience“, Richard Messnarz (Editor), Colin Tully (Co-Editor) IEEE Computer Society Press 1999

[Ess03] Andreas Essigkrug, Thomas Mey: Rational Unified Process kompakt, Spektrum akademischer Verlag 2003

[Ess07] Andreas Essigkrug, Thomas Mey: Rational Unified Process kompakt, 2.Aufl. Spektrum Akademischer Verlag 2007

[EVA00] Evasoft Abschlußbericht: „Analyse und Evaluation der Softwareentwicklung in Deutschland“, Eine Studie für das Bundesministerium für und Forschung, Projektgemeinschaft: GfK Marktforschung GmbH Fraunhofer-Institut für Experimentelles Software Engineering IESE Fraunhofer-Institut Systemtechnik und Innovationsforschung ISI, im Dezember 2000

[Fic08] Telefoninterview mit Matthias Fickler, Business Unit Leiter für Process Improvement bei der Actano GmbH am 08.04.2008

[Fil05] Christian Filßer: Vergleichsmethoden für Vorgehensmodelle, Dipl. Arb. Technische Universität Dresden 1995

[For80] Jay Wright Forrester: Industrial dynamics, MIT Press 1980

[Fow99] Martin Fowler: Refactoring: Improving the Design of existing Code, Addison-Wesley 1999

[Gau02] Markus Gaulke: Risikomanagement in IT-Projekten, Oldenbourg 2002

[Gau04] Markus Gaulke: Risikomanagement in IT-Projekten, Oldenbourg 2004

[Gil88] Tom Gilb: Principles of Software Engineering Management, Addison-Wesley 1988

[Gol95] Dennis R. Goldenson, James D.Herbsleb: After the Appraisal: A Systematic Survey of of Process Improvement, its Benefits, and Factors that influence Success,

Software Engineering Institute, CMU/SEI-95-TR-009-1995

[Gom99] Peter Gomez, Gilbert Probst: Die Praxis des ganzheitlichen Problemlösens, Haupt 1999

[Gra97] Ian Graham: The OPEN Process Specification, Addison-Wesley 1997

[Gre01] James Grenning: Launching Extreme Programming at a Process-Intensive Company, IEEE Software Volume 18, Issue 6 2001

[Gru01] Bruno Grupp: Der professionelle IT-Projektleiter, MITP-Verlag 2001

[Hal98] Elaine Hall: Managing Risks, Addison-Wesley 1998

[Han+05] Geir Kjetil Hanssen, Hans Westerheim, Finn Olav Bjørnson: Using Rational Unified Process in an SME – A Case Study, in Proceedings of the 12th European Conference on Software Process Improvement, Budapest, Hungary, LNCS 3792, I. Richardson et al. (Eds.), Springer 2005

[Han04] Brian Hanks: Distributed Pair Programming – An Empirical Study, in Extreme Programming and Agile Methods – XP/ Agile Universe 2004, Proceedings of 4th Conference on Extreme Programming and Agile Methods, Calgary, Canada 2004, LNCS 3134, C. Zannier, H. Erdogmus, L. Lindstrom (eds.), Springer 2004

[Han04a] John Erik Hansen, Carsten Thomsen: Enterprise Development with Visual Studio .NET, UML, and MSF, Apress 2004

[Hen+98] Brian Henderson-Sellers, Anthony Simons, Houman Younessi: The OPEN toolbox of techniques, Addison-Wesley 1998

[Hen00] Peter Henderson, Yvonne Howard: Simulating a Process Strategy for Large Scale Software Development using Systems Dynamics, in Software Process Improvement and Practice, Volume 5, Issue 2-3, Wiley 2000

[Her08] Telefoninterview mit Christian Hertneck, SEI authorized Instructor for Introduction to CMMI, SEI authorized SCAMPI A/B/C Lead Appraiser (with high maturity certification) und Geschäftsführer der Anywhere.24 GmbH am 01.04.2008

[Hig00] James A. Highsmith: Adaptive Software Development – A Collaborative Approach to Managing Complex Systems, Dorset House Publishing 2000

[Hir02] Michael Hirsch: Making RUP Agile: in Proceedings of the 17th Annual ACM Conference on Object-oriented Programming, Systems, Languages and Applications OOPSLA 2002 Practitioners Reports, Seattle, Washington, USA, ACM Press 2002

[Hig02] Jim Highsmith: Agile Software Development Ecosystems, Addison-Wesley 2002

[Hör+06] Klaus Hörmann, Lars Dittmann, Bernd Hindel, Markus Müller: SPICE in der Praxis, dpunkt.verlag 2006

[Hof81] Geert Hofstede: Culture and Organisations, International Studies of Management & Organisation, Vol 10, No. 4, Sharpe 1981

[Hof+90] Geert Hofstede, Bram Neuijen, Denise Daval Ohayv, Geert Sanders: Measu-

ring Organizational Cultures: A Qualitative and Quantitative Study across Twenty Cases, Administrative Science Quarterly, 35 1990

[Hof05] Geert Hofstede, Gert Jan Hofstede: Cultures and Organisations – Software of the Mind, 2nd Edition McGraw Hill 2005

[Hol03] Harald Holz, Frank Maurer: Knowledge Management Support for Distributed Agile Software Processes, LSO 2002, LNCS 2640, Scott Henninger; Frank Maurer (Eds.), Springer 2003

[Hol+06] Helena Holmström, Brian Fitzgerald, Pär J. Ågerfalk, Eoin Ó. Conchúir: Agile Practices Reduce Distance In Global Software Development, in Information Systems Management 2006 Volume 23, Number 3, Taylor & Francis 2006

[Hoo05] Colin Hood, Rupert Wiebel: Optimieren von Requirements Management & Engineering, Springer 2005

[Hor93] Erika Horn, Wolfgang Schubert: Objektorientierte Software-Konstruktion, Hanser 1993

[Hul+02] M.E.C. Hull, P.S. Taylöor, J.R.P. Hanna, R.J.Millar: Software Development Processes-an assessment, in Information and Software Technology 44 (2002), Elsevier 2002

[Hum00] Watts S. Humphrey: Introduction to the Team Software Process, Addison-Wesley 2000

[Hum90] Watts S. Humphrey: Managing the Software Process, Addison-Wesley 1990

[Hum95] Watts S. Humphrey: A Discipline for Software Engineering: The complete PSP Book, Introduction to the Team Software Process, Addison-Wesley 1995

[Hum97] Watts S. Humphrey: Introduction to the Personal Software Process, Addison-Wesley 1997

[Hum97a] Watts S. Humphrey: Managing Technical People, Addison-Wesley 1997

[Hum02] Watts S. Humphrey: Winning with Software – An Executive Strategy, Addison-Weseley 2002

[ISA00] ISACA: CobiT. 3.Auflage IT Governance Institute, 2000

[ISO06] International Standard ISO/IEC 12207 Software Life Cycle Processes, http://www.abelia.com/docs/12207cpt.pdf am 21.03.2006

[Jac+99] Ivar Jacobson, Grady Booch, James Rumbaugh: The Unified Software Development Process, Addison-Wesley 1999

[Jac+04] Andrew Jackson, Shiu Lun Tsang, Alan Grey, Cormac Driver, Siobhán Clarke: Behind the rules: XP Experiences, in Proceedings of the Agile Development Conference, Salt Lake City, UT, USA, IEEE Computer Society 2004

[Jal00] Pankay Jalote: CMM in Practice – Processes for Executing Software Projects at Infosys, Addison-Wesley 2000

[Jec+04] Mario Jeckle, Chris Rupp, Jürgen Hahn, Barbara Zengler, Stefan Queins: UML2 glasklar, Hanser 2004

[Jef+01] Ron Jeffries, Ann Anderson, Chet Hendrickson: Extreme Programming installed, Addison-Wesley 2001

[Kar93] Gerald M. Karam, Ronald S. Kasselmann: A Cataloging Framework for Software Development Methods, in IEEE Computer, Vol 26 (2), IEEE Computer Society Press 1993

[Kar+00] Even-André Karlsson, Lars-Göran Andersson, Per Leion: Daily build and feature development in large distributed projects, in Proceedings of the 22nd International Conference on on Software Engineering, Limerick, Ireland, ACM Press2000

[Kei05] Patrick Keil: Principal Agent Theory and its Application to AnalyzeOutsourcing of Software Development, in Proceedings of the Int. Workshop on Economics-Driven Software Engineering Research (EDSER), St. Louis, USA, May 2005. IEEE Computer Society Press 2005

[Kel94] Hedwig Kellner: Die Kunst, DV-Projekte zum Erfolg zu führen, Hanser 1994

[Kim05] So-Young Kim, Ho-Jin Choi, An Evaluation of Process Performance for a Small-Team Project – A Case Study, in Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science, IEEE Computer Society 2005

[Kir+01] M. Kirchner et al., Distributed eXtreme Programming, Proceedings of Second International Conference on eXtreme Programming and Agile Processes in Software Development Cagliari, Italy, Addison-Wesley 2001

[Ket05] Petri Kettunen, Maarit Laanti: How to Steer an Embedded Software Project: Tactics for Selecting the Software Process Model, Information And Software Technology (47) 2005, S. 587-608

[Kne+98] Ralf Kneuper, Günther Müller-Luschnat, Andreas Oberweis: Vorgehensmodelle für die betriebliche Anwendungsentwicklung, Teubner Reihe Wirtschaftsinformatik, Teubner 1998

[Kne03] Ralf Kneuper: Verbesserung von Softwareprozessen mit dem Capability Maturity Model Integrated, dpunkt.verlag 2003

[Kne03] Ralf Kneuper: CMMI-Verbesserung von Softwareprozessen mit dem Capability maturity Model Integrated, dpunkt-verlag 2003

[Kne08] Telefoninterview mit Dr. Ralf Kneuper, CMMI Lead Appraiser und selbständiger Berater am 25.03.2008

[Koc08] Telefoninterview mit Lutz Koch, iNTACS Prinzipal Assessor (ISO/IEC 15504), Lead Appraiser und SEI certified CMMI Instructor bei der Firma Wibas IT Maturity Services am 07.04.2008

[Kos04] Juha Koskela, Pekka Abrahamsson: On-Site Customer in an XP Project: Empirical Results from a Case Study, in Proceedings of the 11th European Conference on Software Process Improvement (EuroSPI 2004), Trondheim, Norway, LNCS 3281, T. Dingsøyr (Ed.), Springer 2004

[Kot96] John P. Kotter: Leading Change, Harvard Business School Press 1996

[Kot02] Andreas Kotulla: Management von Softwareprojekten – Erfolgs- und Mißerfolgsfaktoren bei international verteilter Entwicklung, Deutscher Universitätsverlag 2002

[Kri98] David J.Krieger: Einführung in die allgemeine Systemtheorie, UTB für Wissenschaft, Fink Verlag 1998

[Kro03] Per Kroll, Phillippe Kruchten: The Rational Unified Process Made Easy, Addison-Wesley 2003

[Kru99] Phillippe Kruchten: Der Rational Unified Process, Addison-Wesley 1999

[Kuh89] Thomas S. Kuhn: Die Struktur wissenschaftlicher Revolutionen, Suhrkamp 1989

[Kus+04] Clifton Kussmaul, Roger Jack, Barry Sponsler: Outsourcing and Offshoring with Agility: A Case Study, in Extreme Programming and Agile Methods – XP/ Agile Universe 2004, Proceedings of 4th Conference on Extreme Programming and Agile Methods, Calgary, Canada 2004, LNCS 3134, C. Zannier, H. Erdogmus, L. Lindstrom (eds.), Springer 2004

[Lar04] Craig Larman: Agile & Iterative Development – A Mangers Guide, Addison-Wesley 2004

[Law05] Amy Law, Raylene Charron: Human and Social Factors of Software Engineering (HSSE): Effects of agile practices on social factors, in Proceedings of the 2005 workshop on Human and social factors of software engineering HSSE '05, ACM SIGSOFT Software Engineering Notes, Volume 30, Issue 4, ACM Press 2005

[Lay+04] Lucas Layman, Laurie Williams, Lynn Cunningham: Exploring Extreme Programming in Context: An Industrial Case Study, in Proceedings of the Agile Development Conference (ADC'04),Salt Lake City, Utah, USA, IEEE Computer Society 2004

[Leh01] Günter Lehmann: Das Interview – Erheben von Fakten und Meinungen im Unternehmen, expert-Verlag 2001

[Lee04] Bengee Lee, James Miller: Multi-project Software Engineering Analysis Using Systems Thinking, in Software Process Improvement and Practice, Volume 9 Issue 3, Wiley 2004

[Lip+02] Martin Lippert, Stefan Roock, Henning Wolf: Software entwickeln mit eXtreme Programming – Erfahrungen aus der Praxis, dpunkt Verlag 2002

[Lip+03] Martin Lippert, Petra Becker-Pechau, Holger Breitling, Jörn Koch, Andreas Kornstädt, Stefan Roock, Axel Schmolitzky Henning Wolf, Heinz Züllighoven: Developing Complex Problems Using XP with Extensions, IEEE Computer 2003

[Lip04] Martin Lippert: Towards a Proper Integration of Large Refactorings in Agile Software Development, in Extreme Programming and Agile Processes, Proceedings of the 5th International Conference XP 2004, Garmisch-Partenkirchen, Germany 2004, LNCS 3092, Jutta Eckstein; Hubert Baumeister (Eds.), Springer 2004

[Lit95] Hans-Dieter Litke: Projektmanagement – Methoden, Techniken Verhaltensweisen, 3. Aufl. Hanser 1995

[Lit96] Hans-Dieter Litke: DV-Projektmanagement – Zeit und Kosten richtig einschätzen, Hanser 1996

[Maa08] Telefoninterview mit Bonifaz Maag, Process Director und Geschäftsführer der Kugler Maag CIE am 08.04.2008

[Mad00] Ray Madachy, Denton Tarbet: Case Studies in Software Process Modeling with System Dynamics, in Software Process Improvement and Practice, Volume 5 Issue 2-3, Wiley 2000

[Man05] Chris Mann, Frank Maurer: A Case Study on the Impact of Scrum on Overtime and Customer Satisfaction, in Proceedings of the Agile Development Conference (ADC'05) IEEE Computer Society 2005

[Mar+04] Angela Martin, Robert Biddle, James Noble: When XP Met Outsourcing, Extreme Programming And Agile Processes in Software Engineering, Garmisch-Partenkirchen, Germany 2004, LNCS 3092, Jutta Eckstein; Hubert Baumeister (Eds.), Springer 2004

[McC96] Steve McConnel: Rapid Development – Taming Wild Software Schedules, Microsoft Press 1996

[McF96] Bob McFeeley: IDEAL SM: A User's Guide for Software Process Improvement, Handbook CMU/SEI-96-HB001 http://www.sei.cmu.edu/publications/documents/96.reports/96.hb.001.html am 20.06.2006

[Mer+01] Peter Mertens, Andrea Back, Jörg Becker, Wolfgang König, Herrmann Krallmann, Bodo Rieger, August-Wilhelm Scheer, Dietrich Seibt, Peter Stahlknecht, Horst Strunz, Rainer Thome, Hartmut Wedekind (Hrsg.): Lexikon der Wirtschaftsinformatik, Springer 2001

[Mey90] Bertrand Meyer: Objektorientierte Software-entwicklung, Hanser 1990

[Moh98] Niko Mohr, Jens Marcus Woehe: Widerstand erfolgreich managen- Professionelle Kommunikation in Veränderungsprojekten, Campus 1998

[Mou+98] JAB Moura, AFC Medeiros, MADE Barros: R-Cycle - A practical approach for managing processes in the real life cycle of software products: Proceedings of World Multi-Conference on Systemics Cybernetics, and Informatics (SCI 98)/4th International Conference on Information Systems, Analysis and Synthesis (ISAS 98), JUL 12-16, 1998 ORLANDO FLORIDA, p. 356-363, 1998

[Mül01] Matthias M. Müller, Walter F. Tichy: Case Study: Extreme Programming in a University Environment, in ICSE 2001, Proceedings of the 23rd International Conference on Software Engineering, Toronto, Canada, IEEE Computer Society 2001

[Mue+07] Markus Müller, Klaus Hörmann, Lars Dittmann, Jörg Zimmer: Automotive SPICE in der Praxis, dpunkt.verlag 2007

[Mül08] Telefoninterview mit Markus Müller, Principal Assessor Automotive SPICE und Managing Director der Kugler Maag CIE am 16.04.2008

[Mut08] Telefoninterview mit Florian Muth, DNV am 16.04.2008

[Mye95] Glenford J. Myers: Methodisches Testen von Programmen, Oldenbourg 1995

[Oca+05] Alexis Ocampo, Fabio Bella, Jürgen Münch: Software Process Commonality Analysis, in Software Process Improvement and Practice, Volume 10, Issue 3, Wiley 2005

[OEP06] Object Engineering Process Version 2.0 vom 14.04.2003, am 22.02.2006, www.oose.de/oep/index.html

[Oes+99] Bernd Oestereich, Peter Hruschka, Nicolai Josuttis, Hartmut Kocher, Hartmut Krasemann, Markus Reinhold: Erfolgreich mit Objekt-orientierung – Vorgehensmodelle und Managementpraktiken für die objektorientierte Softwareentwicklung, Hanser 1999

[Olf06] Dietrich Olfs, Software Architekt bei BenQ Mobile GmbH und Co OHG, in Erprobungsworkshops für das EVGM anhand realer Projekte am 20.10.2006 und 29.10.06

[OMG06] Object Management Group: Software Process Engineering Metamodel Specification, Vesion 1.1, herunter geladen von http://www.omg.org/docs/formal/05-01-06.pdf am 2.10.06

[Pal02] Stephen R. Palmer, John M. Felsing: A Practical Guide to Feature-Driven Development, Prentice Hall 2002

[Pab08] Telefoninterview mit Roland Pabst, Automotive SPICE Instructor bei Kugler Maag CIE am 22.04.2008

[Pat02] Jeff Patton: Hitting the target: adding interaction design to agile software development, OOPSLA 2002 Practitioners Reports, Seattle, Washington, USA, ACM Press 2002

[Ped04] Keld Pedersen: Software Thinking Improvement Learning Performance Improvement Lessons, in Proceedings of the 11th European Conference on Software Process Improvement (EuroSPI 2004), Trondheim, Norway, LNCS 3281, T. Dingsøyr (Ed.), Springer 2004

[Per+96] Graciela Pérez, Khaled El Emam, Nazim H. Madhavji: Evaluating the congruence of a software process model in a given environment, in Proceedings., Fourth International Conference on the Software Process, IEEE 1996.

[Plo02] Ralf Ploner: Studie It-Kosten und -Performance 2002, Vortrag am ZfU Projektmanagement-Kongress, 20.06.2002, in Auftrag gegeben durch Arthur Andersen AG

[Poo04] Charles J.Poole, Distributed Product Development Using Extreme Programming, Extreme Programming And Agile Processes in Software Engineering, Garmisch-Partenkirchen, Germany 2004, LNCS 3092, Jutta Eckstein; Hubert Baumeister (Eds.), Springer 2004

[Pop94] Karl Popper: Logik der Forschung, Mohr 1994

[Pop03] Mary Poppendieck, Tom Poppendieck: Lean Software Development – An Agile Toolkit, Addison-Wesley 2003

[Pro+99] Stacy J. Prowell, Carmen J. Trammell, Richard C. Linger, Jesse H. Poore: Cle-

anroom Software Engineering: Technology and Process, Addison-Wesley 1999

[Pow+04] Marlys Keeton Powers, Jeff Carter, Goef Lory, Andrew MacMurray: MSF, a pocket guide, Van Haren Publishing2004

[Rag+05] S. Raghunathan, A. Prasad, BK. Mishra et al.: Open source versus closed source: Software quality in monopoly and competitive markets, IEEE Transactions ofn Systems Man and Cyernetics Part A Systems and Humans, Vol. 35, Issue 6, p. 903-918, 2005

[Rai01] Austen Rainer, Tracy Hall: An Analysis of Some 'Core Studies' of Software Process Improvement, in Software Process Improvement and Practice, Volume 6, Issue 4, Wiley 2001

[Rea03] Kris Read, Frank Maurer: Issues in Scaling Agile Using an Architecture-Centric Approach: A Tool-Based Solution, in Extreme Programming and Agile Methods – XP/Agile Universe, New Orleans, LA, USA, LNCS 2753, Frank Maurer, Don Wells (Eds.), Springer 2003

[Ree04] Michael Reeves, Jihan Zhu: Moomba – A Collaborative Environment for Supporting Distributed Extreme Programming in Global Software Development, Extreme Programming And Agile Processes in Software Engineering, Garmisch-Partenkirchen, Germany 2004, LNCS 3092, Jutta Eckstein; Hubert Baumeister (Eds.), Springer 2004

[Rey08] Günter Daniel Rey, Karl F. Wender: Neuronale Netze: Eine Einführung in die Grundlagen, Anwendungen und Datenauswertung, Huber 2008

[Rez04] Mohsen Rezagholi: Prozess- und Technologiemanagement in der Softwareentwicklung – Ein Metrik basierter Ansatz zur Bewertung von Prozessen und Technologien, Oldenbourg 2004

[Ric03] Stefan Richter: Feature-based Programming – Planung, Programmierung, Projektmanagement: Über die Kunst systematisch zu planen und mit Agilität umzusetzen, Addison-Wesley 2003

[Rob04] Hugh Robinson, Helen Sharp: The Characteristics of XP Teams in Agile Software Development, in Extreme Programming and Agile Processes, Proceedings of the 5th International Conference XP 2004, Garmisch-Partenkirchen, Germany 2004, LNCS 3092, Jutta Eckstein; Hubert Baumeister (Eds.) Springer 2004

[Roe+00] Stephen T. Roehling, James S. Collofello, Brian G. Hermann, Dwight E. Smith-Daniels: System Dynamics Modeling Applied to Software Outsourcing Decision Support, in Software Process Improvement and Practice, Volume 5 Issue 2-3, Wiley 2000

[Ros03] Lutz von Rosenstiel: Grundlagen der Organisationspsychologie, Schäfer Poeschl, 5. Aufl.2003

[Ruc+97] Georg Rückriem, Joachim Stary, Norbert Franck: Die Technik wissenschaftlichen Arbeitens, Schöningh 1997

[Rum+93] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen: Objektorientiertes Modellieren und Entwerfen, Hanser 1993

[Rup01] Chris Rupp: Requirements-Engineering und -Management – Professionelle iterative Anforderungsanalyse für die Praxis, Hanser 2001

[RUP06] Evaluationsversion des Rational Unified Process, heruntergeladen von http://www-304.ibm.com/jct09002c/university/scholars/downloads/software.html am 27.09.06

[Rus03] Stuart Russel, Peter Norvig: Artificial Intelligence – A Modern Approach, Pearson Education 2003

[Sch02] Ken Schwaber, Mike Beedle: Agile Software Development with Scrum, Prentice Hall 2002

[Sch02a] Andreas Schramke: Organisatorische Gestaltung von Softwareentwicklungsprojekten, Emporas 2002

[Sch04] Ken Schwaber: Agile Project Management with Scrum, Microsoft Press 2004

[Sch06] Gerhard Schurz: Einführung in die Wissenschaftstheorie, Wissenschaftliche Buchgesellschaft 2006

[Sch08] Telefoninterview mit Dr. Jürgen Schmied, Principal Consultant und Vorstand der Method Park Software AG am 01.04.2008

[Sei96] Helmut Seiffert: Einführung in die Wissenschaftstheorie 1, Beck 1996

[SEI06] Software Engineering Institute, Carnegie Mellon University Pittsburg, USA: The IDEAL Model: Initiating, Diagnosing, Establishing, Acting & Learning http://www.sei.cmu.edu/ideal/ am 28.06.2006

[SEI06a] Software Engineering Institute, Carnegie Mellon University Pittsburg, USA: Capability Maturity Model Integration http://www.sei.cmu.edu/cmmi/ am 28.06.2006

[SEI06b] Software Engineering Institute, Carnegie Mellon University Pittsburg, USA:Standard CMMI Appraisal Mehtod for Process Improvement (SCAMPI ^{SM})A, Version 1.2: Method Definition Document http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06hb002.pdf am 09.03.2007

[Sel02] Robert Sell, Ralf Schirmweg: Probleme lösen – In komplexen Zusammenhängen denken, Springer 2002

[Sen94] Peter Senge: The fifth Discipline, Currency Doubleday 1994

[Sel+94] Bran Selic, Garth Gullekson, Paul T.Ward: Real-Time Object-Oriented Modeling, Wiley 1994

[Sha04] Helen Sharp, Hugh Robinson: An Ethnographic Study of XP Practice, Empirical Software Engineering 9, Kluwer 2004

[Sia02] Kerstin V. Siakas, Elli Georgiadou: Empirical Measurement of the Effects of Cultural Diversity on Software Quality Management, Software Quality Journal Vol. 10, Kluwer 2002

[Smi04] Darja Šmite: Global Software Development Project Management – Distance Overcoming, in Proceedings of the 11th European Conference on Software Process Im-

provement (EuroSPI 2004), Trondheim, Norway, LNCS 3281, T. Dingsøyr (Ed.), Springer 2004

[Som04] Ian Sommerville: Software Engineering, Addison Wesley 2004

[Spe80] Josef Speck (Hrsg.): Handbuch wissenschaftstheoretischer Begriffe, Bd. 1-3, Vandenhoeck und Ruprecht 1980

[SQI06] http://www.sqi.gu.edu.au/spice/ am 28.06.2006

[Sta94] Standish Group: The CHAOS-Report, www.standishgroup.com/sample_research/chaos_1994_1.html, am 02.03.2006

[Sta95] Standish Group: Unfinished Voyages – A follow-up to the Chaos Report: http://www.standishgroup.com/sample_research/unfinished_voyages_1.php am 24.03.2006

[Sta99] Standish Group: Chaos: A Recipe for Success, http://www.standishgroup.com/sample_research/PDFpages/chaos1999.pdf am 24.03.2006

[Sta01] Standish Group: Extreme Chaos, http://www.standishgroup.com/sample_research/PDFpages/extreme_chaos.pdf am 24.03.2006

[Sta04] Standish Group: 2004 Third Quarter Research Report, http://www.standishgroup.com/sample_research/PDFpages/q3-spotlight.pdf am 24.03.2006

[Sta98] Jennifer Stapleton: DSDM Dynamic Systems Development Method, Addison-Wesley 1998

[Sta03] Jennifer Stapleton: DSDM Business Focused Development, Addison-Wesley 2003

[Ste97] Wolfgang Stein: Objektorientierte Analysemethoden – Vergleich, Bewertung, Auswahl, Spektrum Akademischer Verlag 1997

[Ste98] Dirk Stelzer, Werner Mellis: Success Factors of Organizational Change, Software Process Improvement and Practice, Volume 4, Issue 4, Wiley 1998

[Suc01] Giancarlo Succi, Michele Marchesi: Extreme Programming Examined, Addison-Wesley 2001

[Sve05] Harald Svensson: A Framework for Improving Soft Factors in Software Development, in Proceedings of the 12th European Conference on Software Process Improvement (EuroSPI 2005), Budapest, Hungary, LNCS 3792, I. Richardson et al. (Eds.), Springer 2005

[Tha98] Georg Erwin Thaller: SPICE – ISO 9001 und Software in der Zukunft, bhv Verlag 1998

[The92] Manuel R. Theissen: Wissenschaftliches Arbeiten, Vahlen 1992

[Van08] Telefoninterview mit Bhaskar Vanamali, Competent Assessor bei Kugler Maag CIE am 11.04.2008

[Ver00] Gerhard Versteegen: Projektmanagement mit dem Rational Unified Process,

Springer 2000

[Ves99] Frederic Vester: Neuland des Denkens, dtv 1999

[VMX06] V-Modell XT Release 1.2 vom 01.02.2006, http://www.kbst.bund.de/doc,-308526/V-Modell-XT-Release-1.2.htm am 15.02.2006

[VMXT08] V-Modell XT Release 1.2.1.1 vom 06.2008, ftp://ftp.tu-clausthal.de/pub/institute/informatik/v-modell-xt/Releases/1.2.1.1/Dokumentation/V-Modell-XT-Gesamt.pdf am 04.07.2008

[Wäy+04] Jaana Wäyrynen, Marine Bodén, Gustav Boström: Security Engineering and eXtreme Programming: An Impossible Marriage?, In Extreme Programming and Agile Methods – XP/ Agile Universe 2004, Proceedings of 4th Conference on Extreme Programming and Agile Methods, Calgary, Canada 2004, LNCS 3134, C. Zannier, H. Erdogmus, L. Lindstrom (eds.), Springer 2004

[Wak02] William C. Wake: Extreme Programming Explored, Addison-Wesley 2001

[Wal01] Ernest Wallmüller: Software-Qualitätsmanagement in der Praxis – Softwarequalität durch Führung und Verbesserung von Software-Prozessen, Hanser 2001

[Wal04] Ernest Wallmüller: Risikomanagement für IT- und Softwareprojekte – Ein Leitfaden für die Praxis, Hanser 2004

[Wal07] Ernest Wallmüller: SPI – Software Process Improvement mit CMMI und ISO 15504, Hanser 2007

[Wal08] Gespräch mit Univ Doz. Dr. techn. Dipl. Ing. Ernest Wallmüller in Geroldswil am 07.03.2008

[Wes05] Hans Westerheim, Geir Kjetil Hanssen: The Introduction and Use of a Tailored Unified Process – A Case Study, in Proceedings of the 31st EUROMICRO Conference on Software Engineering and Advanced Applications, Porto, Portugal, IEEE 2005

[Wil00] Helmut Willke: Systemtheorie I: Grundlagen, UTB für Wissenschaft, Lucuis & Lucius 2000

[Wil99] Helmut Willke: Systemtheorie II: Interventionstheorie, UTB für Wissenschaft, Lucuis & Lucius 1999

[Wil01] Helmut Willke: Systemtheorie III: Steuerungstheorie, UTB für Wissenschaft, Lucuis & Lucius 2001

[Wic00] Thorsten Wichmann, Michael Nürnberg: Risikomanagement in Softwareunternehmen, in Praxis des Risikomanagements – Grundlagen, Kategorien, branchenspezifische und strukturelle Aspekte, Schäffer-Poeschl 2000

[Win+07] Jessica Winkler, Jens Dibbern, Armin Heinzl: Der Einfluß kultureller Unterschiede beim IT-Offshoring, Wirtschaftsinformatik 49 (2007) 2, Springer 2007

[Woi05] D.M.Woit: Requirements Interaction Management in an Extreme Programming Environment: A Case Study, in Proceedings of the 27th International Conference on Software Engineering (ICSE 2005), St. Louis, Missouri, USA, ACM Press2005

[Wol+05] Henning Wolf, Stefan Roock, Martin Lippert: eXtreme Programming – Eine Einführung mit Empfehlungen und Erfahrungen aus der Praxis, dpunkt.verlag 2005

[Xia+04] Yang Xiaohu, Xu Bin, He Zhijun: Extreme Programming in Global Software Development, in Proceedings of the Canadian Conference on Electrical and Computer Engineering (CCECE/CCGEI), IEEE Canada 2004

[XU05] Bin XU: Extreme Programming for Distributed Legacy System Engineering, in Proceedings of the 29th Annual International Computer Software and Applications, IEE 2005

[Yap05] Monica Yap: Follow the sun, in Proceedings of the Agile Development Conference, IEEE 2005

[You04] Edward Yourdon: Death March, Prentice Hall 2004

[Zie05] Alexander Ziegler: Ein gesamtheitliches Verfahren für das Software Process Improvement. In: Software Engineering – Objektorientierte Methoden in der Praxis, Oldenbourg 2005

8 Abbildungsverzeichnis

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, daß ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; aus fremden Quellen direkt oder indirekt übernommene Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

München, 18.02.2009 Alexander Ziegler

Anhang A1.Die Systematik

Im folgenden wird die Ontologie hergeleitet, welche in der vorliegenden Arbeit die Projekt- und Vorgehensmodellmerkmale systematisch einordnet. Die Systematik für die Projektmerkmale wird nachfolgend entwickelt.

Die Systematik für Projekte und Vorgehensmodelle

Der Zustand eines Projekts ist durch vielfältige Eigenschaften in verschiedenen Facetten gekennzeichnet. Dadurch werden Projekte schwer überschaubar. Übersicht ist unverzichtbar, wenn man ein Projekt aber insgesamt verstehen will, um es umfassend mit Maßnahmen durch Vorgehensmodelle beeinflussen zu können. Um die Situation in einem Projekt überschaubar zu machen, muß sie mit Hilfe einer Systematik beschrieben, überdeckt und strukturiert werden.
Die im folgenden vorgeschlagene Systematik zur Beschreibung von Projekten beschreibt zwei Dimensionen. Innerhalb dieser Dimensionen erfolgt eine Unterteilung nach Aspekten. Die Dimensionen und Aspekte zur Beschreibung von Projekten werden nachfolgend hergeleitet, motiviert und definiert.

Die Dimension Risikobereiche

Vorgehensmodelle dienen dem softwaretechnischen Projektmanagement, dessen Ziel es ist, Softwareentwicklungs-Probleme im Projekt zu lösen. Probleme können als manifestierte Risiken und Risiken als antizipierte Probleme betrachtet werden. Projektmanagement ist daher eng verbunden mit dem Risikomanagement, wie auch [Wal04] bestätigt. Daher läßt sich die Systematik für Projekte aus Risikoschwerpunkten herleiten. [You04] definiert hierzu die „risk areas“ „software project“, „organizational environment“ und „business environment“. Diese Risikobereiche (siehe Anhang, Abschnitt Glossar) lassen sich präziser systematisieren.
Der Risikobereich „software project“ läßt sich nach folgenden Risikobereichen diversifizieren:
Entwicklungs-Team umfasst für die den Projekterfolg wesentlichen Eigenschaften der Projektbeteiligten und des Projektteams.
Entwicklungs-Produkt umfasst die für den Projekterfolg wesentlichen Eigenschaften und Aspekte des Projektergebnisses.
Die Risikobereiche Entwicklungs-Team und Entwicklungs-Produkt dienen der systematischen Einordnung von Merkmalen, die wesentlich für den Zustand eines Projektes sind.
Die Risikobereiche „organizational environment“ und „business environment“ lassen sich verallgemeinern in den Risikobereich
Entwicklungs-Umfeld, der die für den Projekterfolg wesentlichen Eigenschaften umfasst, die nicht unmittelbar aus dem Software Projekt resultieren. Er wird verwendet, um Fragen zu Einflußfaktoren aus dem Projektumfeld systematisch einzuordnen, die Benutzern des EVGM helfen, Merkmale auszuwählen und mit Zuständen zu versehen.

Der Risikobereich **Entwicklungs-Prozeß** umfasst die für den Projekterfolg wesentlichen Eigenschaften von Vorgehensmodellen. Die Merkmale im Risikobereich Entwicklungs-Prozeß adressieren die Merkmale aller anderen Risikobereiche.

Zusammengefasst dargestellt müssen in der Dimension Risikobereiche Themen des

Teams, des Produkts, des Umfelds und des Prozesses abgedeckt werden.
Diese Darstellung umfasst die wesentlichen Risikobereiche im Kontext von Softwareentwicklungsprojekten und kann zur Überprüfung von dessen kompletter Überdeckung herangezogen werden.

Die Dimension Projektthemenbereiche

Als zweite Dimension der Systematik lassen sich die folgenden Projektthemenbereiche (siehe Glossar im Anhang) in Softwareprojekten finden:
Domänenwissen. Alle Merkmale des Projektes, welche durch im Projekt zu bewältigende, inhaltliche Aufgabenstellung bedingt sind.
Softwareentwicklung. Alle Merkmale des Projektes, welche durch die Bearbeitung der inhaltlichen Aufgabenstellung mit Hilfe des Software Engineerings und des Software Managements bedingt sind. Software Engineering besteht aus Analyse, Design, Implementierung und Test. Software Management besteht aus den Disziplinen Requirements Management, Change Management und Konfigurationsmanagement (siehe Glossar im Anhang).
Projektmanagement. Alle Merkmale des Projektes, welche durch die organisatorische und kaufmännisch-administrative Begleitung der Softwareentwicklung bedingt sind (siehe Kapitel 2).
Menschenführung. Alle Merkmale des Projektes, welche durch menschliche und zwischenmenschliche Faktoren sozialer, emotionaler, kultureller oder politischer Natur bedingt sind, welche die Bearbeitung der inhaltlichen Aufgabenstellung begleiten. Sie besteht aus der Motivation, Kommunikation und den Führungsfähigkeiten (siehe Kapitel 2).

Zusammengefasst dargestellt müssen in der Dimension Projektthemenbereiche fachliche, technische, administrative und menschliche Themen abgedeckt werden.
Diese Systematik umfasst die wesentlichen Projektthemenbereiche im Kontext von Softwareentwicklungsprojekten und kann zur Überprüfung von dessen kompletter Überdeckung herangezogen werden.
Diese Darstellung lehnt sich an die Unterteilung des Problembereiches in einen fachlichen, einen technischen, einen organisatorischen und einen menschlichen Aspekt in [Zie05] an.

Die komplette Systematik für die Überdeckung des Projektes wird nachfolgend anhand einer Tabelle veranschaulicht. Sie zeigt beide Dimensionen mit allen ihren Bereichen im Zusammenhang.

Risikobereich ***Projektthemenbereiche***	***Entwicklungs-Team***	***Entwicklungs-Produkt***	***Entwicklungs-Umfeld***	***Entwicklungs-Prozeß***
Domänenwissen				
Softwareentwicklung				
Projektmanagement				
Menschenführung				

Tabelle 16 Systematik für Projekte

Anforderungen an die Systematik

Die Systematik für Projekte und Vorgehensmodelle genügt den folgenden Anforderungen.
Sie überdeckt den Problembereich von Softwareprojekten komplett, da die Dimensionen, sowie die Bereiche innerhalb der Dimensionen komplett sind.
Sie ist in sich überschneidungsfrei und frei von Widersprüchen, da die Dimensionen sowie die Bereiche innerhalb der Dimensionen überschneidungsfrei definiert sind. Die Systematik überdeckt den Problembereich von Softwareprojekten bedarfsgerecht und fokussiert empirisch belegbare Risikoschwerpunkte.

Anhang A2.Herleitung der Projektmerkmale

Die Projektmerkmale werden im folgenden hergeleitet. Bei der Herleitung der Merkmale von Projekten werden in der vorliegenden Arbeit Projekterfolgs- und -risikofaktoren aus der Literatur herangezogen (siehe jeweils einzelne Projektmerkmale in Kapitel 2.)

Projekterfolgs-, und -risikofaktoren

Projekterfolgs- bzw. -risikofaktoren in der Literatur beschreiben wesentliche Aspekte von Projekten, die entscheidenden Einfluß auf den Erfolg oder das Scheitern eines Projektes haben. Auch [Boe04] betont, daß Risiken ein geeigneter Ausgangspunkt für Prozeßentscheidungen in Projekten sind.
Hierzu muß erwähnt werden, daß die Risikofaktoren in den entsprechenden Studien so gut wie immer als Umkehrung der Erfolgsfaktoren gesehen werden können. Risiko- und Erfolgsfaktoren unterscheiden sich nur durch ihren jeweiligen projektgefährdenden oder projektbegünstigenden Zustand. Daher können sowohl Projektrisiko- und Projekterfolgsfaktoren als Ausgangspunkt für die Bildung von Projektmerkmalen herangezogen werden.
Beispielsweise begünstigen stabile und transparente Anforderungen den Projekterfolg, instabile und intransparente Anforderungen gefährden ihn.
Definition:
Ein **Projekterfolgsfaktor** ist ein Projektmerkmal in einem Zustand, der den Projekterfolg begünstigt.
Ein **Projektrisikofaktor** ist ein Projektmerkmal in einem Zustand, der den Projekterfolg gefährdet(vergleiche „Glossar“ im Anhang).

Deshalb werden in der vorliegenden Arbeit Projekterfolgsfaktoren und Projektrisikofaktoren zu Projektmerkmalen verdichtet.
Projektmerkmale unterscheiden sich allerdings in der Sicht des Autors von Projekt zu Projekt durch ihren Anfangszustand. Die Zustände von Projektmerkmalen können sich während der Projektlaufzeit ändern.
Beispielsweise können die Anforderungen in einem Projekt sehr instabil sein und dieses Merkmal hat somit einen Zustand, der den Projekterfolg gefährdet. In einem anderen Projekt sind die Anforderungen unter Umständen weitestgehend stabil und dieses Merkmal ist nicht in einem kritischen Zustand.
Die Projektrisikofaktoren und Projekterfolgsfaktoren, welche als Grundlage für die Projektmerkmale herangezogen werden, sind der Literatur entnommen, welche nachfolgend im Überblick dargestellt wird.

Zur Erhebung benutzte Literaturquellen

Markus Gaulke führt in [Gau02], sowie in der überarbeiteten und erweiterten Auflage [Gau04] die nachfolgend dargestellten Studien auf. Einige der Studien aus [Gau04] konnten nicht im Original zitiert werden, da diese dem Autor der vorliegenden Arbeit nicht zugänglich waren. Um die betroffenen Zitate im Text zu kennzeichnen, werden sie in Anlehnung an [Ruc+97] und [The92] jeweils „bei“ der jeweiligen Bezeichnung ihrer Originalquelle und dem Zusatz „zitiert nach [Gau04]“ zitiert.
Nachfolgend wird ein Kurzüberblick über die zitierten Studien dargestellt.

- Der „Report on IT Runaway Systems“ zitiert nach [Gau04], durchgeführt 1994 von der KPMG Management Consulting bei über 120 Unternehmen verschiedenster

Branchen. Gefragt wurde nach Gründen für Projektmisserfolge.

- Die „TechRepublic“ Studie zitiert nach [Gau04], durchgeführt 2000 von der TechRepublic Inc., einer Tochter der Gartner Group, bei der 1375 nordamerikanische IT-Spezialisten von Unternehmen unterschiedlichster Größe und Branchen befragt wurden. Gefragt wurde nach Erfolgs- und Risikofaktoren für IT-Projekte.
- Der Chaos Report der Standish Group [Sta94], 1994 bei 365 amerikanischen Unternehmen unterschiedlichster Branchen durchgeführt. Gefragt wurde nach Gründen für Projektmisserfolge, sowie nach Erfolgsfaktoren bei IT-Projekten.
- „What went wrong? Unsuccessful Information Technology Projects“ zitiert nach [Gau04], durchgeführt von der KPMG Strategic and Technology Services 1997 bei 167 öffentlichen und privaten Unternehmen in Kanada. Gefragt waren die Gründe für das Scheitern von Projekten.
- Eine Umfrage der Fachzeitschrift Computerwoche zitiert nach [Gau04] durchgeführt 1997 bei 182 deutschen Unternehmen. Gefragt waren hier die häufigsten Probleme bei IT-Projekten.
- Eine Studie des Daily Telegraph zitiert nach [Gau04], durchgeführt 1998 in Großbritannien. Untersucht wurden die Gründe für das Scheitern von von IT-Projekten, sowie kritische Erfolgsfaktoren.
- Eine Befragung des forsa-Instituts zitiert nach [Gau04], durchgeführt 1998 bei Führungskräften deutscher Großunternehmen. Gefragt wurde nach Störfaktoren bei der Erreichung von Projektzielen.
- Die Untersuchung „Analyse und Evaluation der Softwareentwicklung in Deutschland“ zitiert nach [Gau04], durchgeführt 2000 durch die GfK Marktforschung in Zusammenarbeit mit dem Fraunhofer-Instituten für Experimentelles Software Engineering (IESE) und für Systemtechnik und Innovationsforschung (ISI) im Auftrag des Bundesministeriums für Bildung und Forschung (BMBF). Bei dieser Untersuchung wurden 920 Vertreter von repräsentativ ausgewählten Unternehmen, sowie 55 zusätzlich ausgewählte Experten befragt. Unter anderem gefragt waren hier auch die Probleme bei der Anwendung von Softwareprozessen im Projekt.
- Eine Untersuchung der Beratungsgesellschaft Droege & Comp. zitiert nach [Gau04], durchgeführt 2002 bei 164 deutschen Großunternehmen. Erhoben wurden auch hier Gründe für das Scheitern von Projekten.
- Die Untersuchung „Programme Management Survey“ der KPMG zitiert nach [Gau04], durchgeführt 2002 bei 124 internationalen Unternehmen. Auch hier wurden die Hauptgründe für das Scheitern von Projekten, sowie Erfolgsfaktoren ermittelt.

Gaulke führt weiterhin die folgenden Expertenmeinungen an:

- Eine Untersuchung von Magerl zitiert nach [Gau04], bei der 16 Projektrisikoeinschätzungen internationaler Projekte bei der Firma NCR aus den Jahren 1997-2000 analysiert wurden. Hierbei wurde nach Projektrisikofaktoren geforscht.
- Eine Untersuchung von Jalote zitiert nach [Gau04], durchgeführt im Rahmen der Befragung des CMM-Level 4 Unternehmens Infosys im Jahr 2000. Auch hier waren die Risikofaktoren von IT-Projekten gefragt.
- Eine Studie von McKinsey Global Institute 2002 zitiert nach [Gau04] führte auf vier Grundprinzipien erfolgreicher IT-Großprojekte.
- Eine Zusammenfassung eigener Erfahrung und diverser Studien von McConnel zitiert nach [Gau04], die klassische Projektfehler und Erfolgsfaktoren ergab.

- Die Darstellung von Boehm zitiert nach [Gau04] führt Grundprinzipien der Softwareentwicklung aus dem Jahr 1983, sowie Hauptrisiken bei der Softwareentwicklung an [Boe91].

Gaulke wertet diese Studien in [Gau02], [Gau04] selbst aus, indem er eigene Charakteristika für Projekte ableitet. Auch diese werden für die Herleitung der Projektmerkmale verwendet.

Die Standish Group hat nach der Chaos Studie von 1994 [Sta94] (siehe oben) weitere Untersuchungen durchgeführt, die in Auszügen frei verfügbar waren und im folgenden dargestellt werden.

- Die Studie „Unfinished Voyages – A follow-up to the Chaos Report“ von 1995 [Sta95] ergänzt zielführende Maßnahmen, um die Erfolgsfaktoren des Chaos Reports positiv zu beeinflussen.
- Die Studie „Chaos: A Recipe for Success“ von 1999 [Sta99], stellt den Zusammenhang zwischen Projektumfang, ausgedrückt in personeller Kapazität, Zeit und Budget, und Projekterfolg dar. Demnach sinken mit steigendem Projektumfang die Erfolgsaussichten eines Projektes. Weiterhin werden die Rollen „Function Representative“, „Executive Sponsor“ und „Project Manager“ charakterisiert.
- Die Studie „Extreme Chaos“ von 2001 [Sta01] bringt die klassischen Erfolgsfaktoren für IT-Projekte auf neueren Stand. Bemerkenswert ist hierbei, daß sich zwar Häufigkeitsreihenfolgen von Projekterfolgsfaktoren ändern, die Faktoren aber weitestgehend und prinzipiell die gleichen wie die von 1994 sind. Dies ist auch bei [Gau02] im Verhältnis zu [Gau04] zu beobachten. Die grundsätzlichen Erfolgsfaktoren für IT-Projekte stellen sich somit als relativ konstant über die Zeit dar.
  Die Studie benennt unter anderem erforderliche Skills von Project Managers.
- Die Studie „2004 Third Quarter Research Report“ von 2004 [Sta04] typisiert IT-Projekte nach ihrer Entwicklungstiefe.

Vorgehensweise bei der Herleitung der Merkmale

Aus den eben benannten Studien, ergänzt um weitere Literaturquellen, werden vom Autor der vorliegenden Arbeit Risiko- und Erfolgsfaktoren entnommen. Diese werden in der vorliegenden Arbeit zu Merkmalen verdichtet, wenn gezeigt werden kann, daß sie dezidiert von Vorgehensmodellen adressiert werden.

Bewertung der Faktoren aus der Literatur

Die Projekterfolgs- und -Risikofaktoren aus der Literatur werden vom Autor als Untermauerung der in der vorliegenden Arbeit verwendeten Projektmerkmale benutzt. Es werden hierbei viele unterschiedliche Quellen herangezogen, um die Herleitung der Projektmerkmale auf eine möglichst breite empirische Literatur-Grundlage zu stellen.
Hiermit wird der Tatsache Rechnung getragen, daß jede der hierbei verwendeten Studien in einem speziellen Kontext durchgeführt wurde und durch die von den Erstellern der Studien vorgegebenen geschlossenen Fragen die Möglichkeiten der Antworten für die Befragten begrenzt waren. Jede der Studien stellt also einen möglichen Blickwinkel dar und kann daher keine absolute Aussagekraft besitzen.
Weiterhin sind die in den Studien angeführten Faktoren nicht immer eindeutig formu-

liert, so daß durch ein Mitteln aus mehreren ähnlichen Faktoren eine höhere Sicherheit bezüglich der aus ihnen abgeleiteten Folgerungen erlangt werden kann. Deshalb werden in der vorliegenden Arbeit ähnliche und auch gleiche Faktoren, die aber aus unterschiedlichen Quellen stammen, explizit aufgeführt, um ein möglichst originalgetreues Bild der Gesamtheit der verwendeten Literaturquellen und ihrer verschiedenen Sichten zu vermitteln.
Auch in der SUCCESS-Studie [Bus+06] wird kritisiert, daß die genannten Studien nur bedingte Aussagekraft besitzen. Dies untermauert den Standpunkt, ein Kondensat aus den Ergebnissen mehrerer Studien zu bilden. Insgesamt kommt die SUCCESS-Studie zu weitgehend identischen Erfolgs- und Risikofaktoren für Softwareentwicklungsprojekte wie die vorliegende Arbeit.
Die Projekterfolgs- und risikofaktoren aus der weiteren Fachliteratur geben den anerkannten Stand der Fachkenntnis innerhalb des Software-Projektmanagements wieder und werden in der vorliegenden Arbeit zur Ergänzung der Ergebnisse der empirischen Studien verwendet. Auch diese stellen die Sichtweise des jeweiligen Autors dar, die ebenfalls keinen Anspruch auf Allgemeingültigkeit besitzen kann.
Durch die breite Literatur-Basis mit vielen unterschiedlichen und jeweils individuellen Blickwinkeln wird in der vorliegenden Arbeit eine umfassende Erhebung und eine stabile Basis für die Herleitung der Merkmale erreicht.
Der empirischen Argumentation, die sich auf die Literaturquellen stützt, wird in der vorliegenden Arbeit eine rationale Plausibilisierung als ergänzende Begründung zur Seite gestellt. Für die aus den Projekterfolgs- und Risikofaktoren aus der Literatur abgeleiteten Projektmerkmale wird vom Autor dieser Arbeit jeweils diskutiert, welche Bedeutung sie für Softwareentwicklungs-Projekte haben können. Somit wird plausibel gemacht, daß die Projektmerkmale wesentlich für die Charakterisierung von Projekten sind.
Zusätzlich wird begründet, warum sie für die Eignungseinstufung von Vorgehensmodellen in der vorliegenden Arbeit bedeutsam sind.

Definieren von Merkmalen

Merkmale. In der vorliegenden Arbeit werden aus den Literaturhinweisen jeweils systematisch Merkmale für die spätere Modellierung im EVGM herausgearbeitet und definiert. Dies dient der methodischeren Unterscheidung, Betrachtung und Abdeckung der Facetten der Problemdomäne Softwareentwicklungsprojekte. Merkmale sind konkret, projektspezifisch mit Werten von „--“ bis „++“ bewertbar (siehe Kapitel 4), miteinander vernetzt, betreffen direkt Entwicklungs-Produkt und Entwicklungs-Team und liegen somit im Kern des EVGM. Sie werden von Vorgehensmodellen adressiert und dienen somit dem Prozeßmanagement.
Für die Evaluierung der Merkmalen wurden die folgenden Kriterien verwendet, die der unten dargestellten Systematik folgen. Diese Kriterien für die Definition von Merkmalen unterstützen die Anwender dabei, das EVGM selbst geeignet um eigene Merkmale zu erweitern. Neue Merkmale können anhand dieser Kriterien auf ihre Eignung überprüft werden.

Was ist die Bedeutung des Merkmals, welchen Definitionsbereich soll es umfassen?

- Das Merkmal muß hinreichend konkret definiert sein, um überprüft werden zu können
- Das Merkmal muß eindeutig definiert sein

- Das Merkmal muß durch seine Definition von den anderen Merkmalen abgegrenzt sein

Warum ist das Merkmal bedeutungsvoll, wodurch wird es motiviert?
- Das Merkmal muß von seiner Bedeutung für das Projekt plausibel sein
- Das Merkmal muß von seiner Bedeutung für das EVGM plausibel sein

Wie wird das Merkmal benutzt, wie steht es in Wechselwirkung zu den anderen Merkmalen, welche Beeinflussungsbeziehungen hat es?
- Das Merkmal muß mit einem qualifizierten Zustand („++“ bis „--“) bewertbar sein
- Das Merkmal muß als Abdeckungsgrad eines Bedarfs beziehungsweise Erfüllungsgrad eines Kriteriums formuliert sein. Bei der Bezeichnung der Merkmale wird die Begriffssystematik „Bedarfsdeckung“, „Stabilität“, „Transparenz“ und „Einfachheit“ verwendet.
- Das Merkmal muß direkte Wirkung auf Projekterfolgsaussichten haben, also direkt relevant für den Zustand des Projektes sein
- Das Merkmal muß direkt durch Vorgehensmodellmerkmale beeinflußbar sein oder der Bedarf an Vorgehensmodellmerkmalen muß aus ihr herleitbar sein
- Das Merkmal muß in Beeinflussungsbeziehung mit anderen Merkmalen stehen

Wann ist das Merkmal von Bedeutung, welche Vorbedingungen sind erforderlich?
- Was sind die Signifikanzbedingungen für ein Merkmal, unter welchen Vorzeichen wird es besonders wichtig?
- Unter welchen Vorzeichen verliert ein Merkmal an Bedeutung?

Zu diesem Zweck werden Fragen zu Einflußfaktoren aus dem Projektumfeld formuliert, die in Beziehung mit Merkmalen gesetzt sind (siehe Abschnitt 2.4).

Für jedes Merkmal wird ähnlich zum Ansatz der Class-Responsibility-Collaboration-Cards (vgl. [Bal99])
- definiert, was das Merkmal bedeutet,
- begründet, warum das Merkmal für Softwareentwicklungsprojekte bedeutsam ist,
- dezidiert gezeigt, wie sie durch konkrete Vorgehensmodelle unterschiedlich adressiert wird, um ihre Bedeutung für die Problemstellung der Eignungseinstufung von Vorgehensmodellen zu verdeutlichen,
- dargestellt, mit welchen anderen Merkmalen sie in Wechselwirkung steht.

Motivation der Projektmerkmale

Für jedes Projektmerkmal wird in der vorliegenden Arbeit jeweils dargestellt, welche Bedeutung es für Softwareentwicklungsprojekte hat. Danach wird jeweils seine Relevanz für die Eignungseinstufung von Vorgehensmodellen belegt.
Hierbei wird jeweils gezeigt, daß konkrete Vorgehensmodelle das Projektmerkmal unterschiedlich adressieren.
Ein Projektmerkmal, welches nicht durch Vorgehensmodelle beeinflusst wird, bildet kein Unterscheidungskriterium für die Wahl eines Vorgehensmodells und ist somit für die Eignungseinstufung von Vorgehensmodellen nicht geeignet.

Verdeutlichen von Beeinflussungsbeziehungen

Jedes Projektmerkmal wird anhand seiner Beeinflussungsbeziehungen mit anderen Merkmalen in Zusammenhang gesetzt. Die Merkmale werden durch Veranschaulichung ihrer Beeinflussungsbeziehungen mit anderen zusätzlich hinsichtlich ihrer Relevanz und Eignung für das EVGM plausibel gemacht. Die Beeinflussungsbeziehungen stellen eine Auswahl aus der Vielzahl denkbarer und möglicher Wechselwirkungen dar. Sie entsprechen den Einschätzungen des Autors der vorliegenden Arbeit und haben Vorschlagscharakter. Die Beeinflussungsbeziehungen unterstreichen die Relevanz durch sie in Zusammenhang gesetzten Merkmale, weil durch deren hohe Vernetzung eine Veränderung ihres Zustandes große Auswirkungen auf den Gesamtzustand hat.

Anhang A3.Modellelementtypen in Vorgehensmodellen

Die Vorgehensmodell-Elementtypen, die unterhalb der Unterstützungsdisziplinen angeordnet sind, werden nachfolgend definiert. Die Modellelementtypen dienen der Beschreibung der Unterstützung von Projektthemenbereichen durch Vorgehensmodelle. Sie unterstützen die Beeinflussung der Projektmerkmale durch die Vorgehensmodellmerkmale. Die Modellelementtypen sind Rollen, Aktivitäten, Arbeitsprodukte, Prinzipien und Methoden entsprechend der Definition im Glossar. Die Auswahl der Modellelementtypen orientiert sich an [Chr92]. Modellelementtypen repräsentieren die unterschiedlichen Beschreibungsaspekte in Vorgehensmodellen.
Modellelementtypen werden in der vorliegenden Arbeit implizit innerhalb der Vorgehensmodellmerkmale dargestellt, da ihre isolierte Betrachtung nicht zielführend in Bezug auf die Verbesserung des Zustandes der Merkmalen von Projekten ist.
Beispielsweise bewirkt ein umfassendes Rollenmodell in einem Projekt nur dann eine bessere Organisation eines großen Requirements Engineering Teams, wenn es auch die Rollen des Projektthemenbereichs Requirements Engineering beschreibt. Die Modellelementtypen werden in der Abbildung 42 im Überblick veranschaulicht.

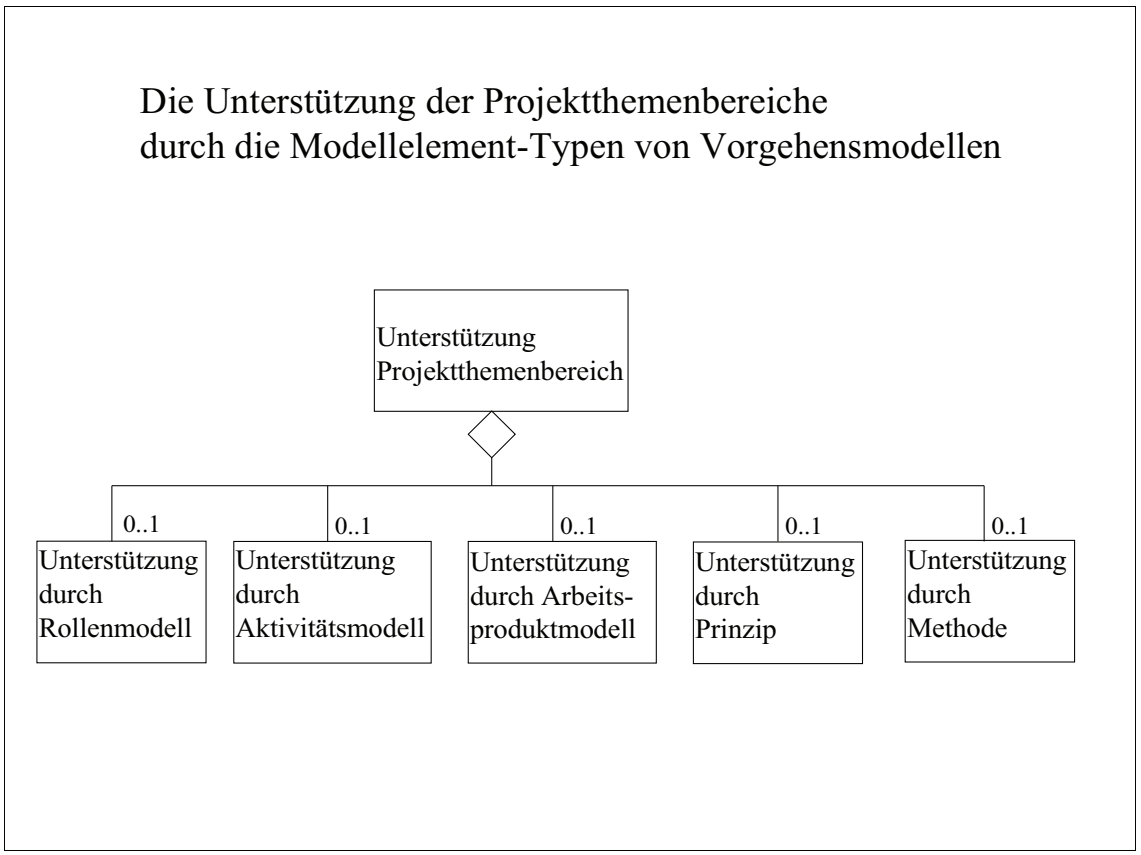

Abbildung 42 Unterstützung der Projektthemenbereiche durch die Modellelement-Typen

Die Modellelementtypen sind überschneidungsfrei und überdecken die Projektthemenbereiche.
Diese Überdeckung der Projektthemenbereiche durch die Modellelementtypen wird noch einmal anhand der folgenden Tabelle 17 veranschaulicht.

Modellelement-typen *Projektthemen-bereiche*	*Rollen, Rollenmodell*	*Aktivitäten, Aktivitätsmodell*	*Arbeitsprodukte, Arbeitsprodukt-modell*	*Prinzipien, Richtlinien, Werte*	*Methoden, Prak-tiken, Techniken*
Requirements Engineering					
Software Engineering					
Software Management					
Projektmanage-ment					
Menschenfüh-rung					

Tabelle 17 Systematik zur Überdeckung des Themenfeldes der Vorgehensmodelle

Unterstützung durch Rollenmodell

Definition der „Unterstützung durch Rollenmodell“:
Rollen (siehe auch Glossar) werden in Vorgehensmodellen zu Rollenmodellen verknüpft. Rollenmodelle beschreiben die durch Kommunikations- und Weisungspfade gegebenen Abhängigkeiten von Rollen. Rollen und Rollenmodelle sind ein etablierter Beschreibungsaspekt in Vorgehensmodellen [Chr92]. „Unterstützung durch Rollenmodelle“ bildet ab, wie umfassend und detailliert ein Vorgehensmodell die Rollenorganisation für das Softwareentwicklungsteam beschreibt. „Umfassend“ steht für die Überdeckung der relevanten Projektthemenbereiche durch ein Rollenmodell. „Detailliert“ repräsentiert, wie konkret und präzise die Beschreibungen von Rollen sind. Je detaillierter die Beschreibung von Rollen ist, desto leichter sind sie vom Anwender eines Vorgehensmodells umzusetzen, desto restriktiver wirken sie allerdings auch.
Beispiel: Der *Anforderungsanalytiker* im V-Modell XT [VMX06] ist eine Rolle des Projektthemenbereichs „Requirements Engineering“.

Motivation der „Unterstützung durch Rollenmodell“:
Bedeutung für Softwareprojekte. In Projekten sind unterschiedliche Verantwortlichkeiten zu koordinieren. Die Menschen im Projekt, ihre produktive Zusammenarbeit, Verantwortlichkeiten und Fähigkeiten sind unerlässlich für einen Projekterfolg. [Gau02] stellt unklare Verantwortlichkeiten als klassisches Problem in Projekten dar. Je nach Umfang eines Projektes und nach Größe des Teams sind in Projekten viele Verantwortlichkeiten zu definieren, koordinieren und kontrollieren. Weiterhin müssen Verantwortlichkeiten für Aktivitäten und Arbeitsprodukte festgelegt und die Kommunikation zwischen den Rollen bzw. Personen definiert und organisiert werden, um die Projektarbeit möglichst effizient und effektiv zu gestalten. Hierbei muß beachtet werden, welche Personen aufgrund ihrer Fähigkeiten welche Rollen übernehmen können. All dies ist aber bei steigender Projektgröße für den einzelnen Projektleiter nicht mehr aus dem Gedächtnis überschaubar, da Hinterkopfwissen nicht skaliert [Boe04]. Komplizierte Abhängigkeiten vieler Rollen können durch Rollenmodelle abgebildet werden.
Beispiel: Rollenmodelle beeinflussen die Unterstützung der Organisation der relevanten Projektthemenbereiche. Die wirkt sich auf das Merkmal *EPP-Einf-DurchTeamgröße* aus Kapitel 2 aus.

Bedeutung für die Eignungseinstufung von Vorgehensmodellen Vorgehensmodelle unterstützen ihre Benutzer mit unterschiedlich umfassenden und detaillierten Rollenmodellen. V-Modell XT [VMX06] und Rational Unified Process [RUP06] arbeiten mit sehr umfassenden Rollenmodellen, Extreme Programming [Bec00] und Scrum [Sch02] definieren Rollen nur ansatzweise (siehe Kapitel 3.3).
Beispiel: Scrum definiert die Rollen *Scrum Master, Product Owner, Scrum Team Member* und *Chickens*. Der Rational Unified Process definiert Rollen wie beispielsweise *Stakeholder, Project Manager, Process Engineer, Implementer, Tester, SW Architect, System Analyst, DB Designer, UI Designer, Graphic Artist und Technical Writer* [RUP06] und organisiert diese in *Role Sets* (siehe Kapitel 3.3).

Unterstützung durch Aktivitätsmodell

Definition der „Unterstützung durch Aktivitätsmodell“:
Aktivitäten (siehe auch Glossar) werden in Vorgehensmodellen durch Aktivitätsmodelle beschrieben. Diese beschreiben die Reihenfolgeabhängigkeiten und Steuerung der Aktivitäten. Aktivitäten und Aktivitätsmodelle sind ein üblicher Beschreibungsaspekt von Vorgehensmodellen [Chr92].
„Unterstützung durch Aktivitätsmodell“ bildet ab, wie umfassend und detailliert ein Vorgehensmodell eine große Menge von erforderlichen Entwicklungsschritten und deren Abhängigkeiten in einem Projektthemenbereich unterstützt. „Umfassend“ steht hier für die Unterstützung der relevanten Projektthemenbereiche durch ein Vorgehensmodell. „Detailliert“ repräsentiert, wie konkret und präzise die Beschreibungen eines Aktivitätsmodells jeweils sind. Je detaillierter die Beschreibung einer Aktivität ist, desto leichter ist sie für die Anwender eines Vorgehensmodells zu benutzen.
Beispiel: Die Aktivitäten des Vorgehensbausteins *Anforderungsfestlegung* im V-Modell XT [VMX06] sind ein Teil von dessen Aktivitätsmodell, der sich auf den Projektthemenbereich „Requirements Engineering“ bezieht.

Motivation der „Unterstützung durch Aktivitätsmodell“:
Bedeutung für Softwareprojekte. Oft sind in Softwareentwicklungsprojekten komplizierte Abfolgen von Entwicklungstätigkeiten erforderlich, die Schritt für Schritt vom Problem zur Lösung führen. Jeder Schritt betrachtet dabei jeweils unterschiedliche Aspekte wie fachliche Anforderungen, Savety- und Security-Aspekte sowie Restriktionen des Applikations-Frameworks und der technischen Plattform. Die hieraus resultierenden Erkenntnisse und Festlegungen werden jeweils dem Entwicklungs-Arbeitsprodukt hinzu gefügt. Eine große Anzahl von Entwicklungsschritten mit allen zu berücksichtigenden Details können Projektmitarbeiter nicht mehr nur aus dem Gedächtnis überblickt werden. Hinterkopfwissen ist nicht skalierbar [Boe04]. Komplizierte Entwicklungsabläufe mit vielen Einzelschritten, Abhängigkeiten, Vor- und Nachbedingungen können in einem in einem umfassenden Aktivitätsmodell beschrieben werden.
Beispiel: Die Unterstützung durch ein Aktivitätsmodell bewirkt eine verbesserte Unterstützung der entsprechenden Projektthemenbereiche.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen ihre Anwender unterschiedlich umfassend und detailliert durch Aktivitätsmodelle. Das V-Modell XT [VMX06] und der Rational Unified Process [RUP06] bieten sehr umfassende Aktivitätsmodelle, Extreme Programming [Bec00] und Scrum [Sch02] definieren keine Aktivitäten (siehe Kapitel 3.3).
Beispiel: Das V-Modell XT [VMX06] definiert Aktivitäten wie *Projekt planen, SW-*

Architektur erstellen oder *Anforderungen festlegen* für die entsprechenden Projektthemenbereiche.

Unterstützung durch Arbeitsproduktmodell

Definition der „Unterstützung durch Arbeitsproduktmodell“:
Arbeitsprodukte *(*siehe auch Glossar im Anhang) werden in manchen Vorgehensmodellen in Arbeitsproduktmodellen zusammengefasst. Arbeitsproduktmodelle stellen die Reihenfolgeabhängigkeiten, Aggregatsbeziehungen oder durch Redundanzen bedingte Abhängigkeiten von Arbeitsprodukten dar. Arbeitsprodukte und Arbeitsproduktmodelle sind ein etablierter Beschreibungsaspekt von Vorgehensmodellen (vgl. [Chr92], dort unter dem Synonym „Ergebnis“). Umfangreiche Informationen mit vielen Abhängigkeiten und Beschreibungsattributen können in einem Projekt mit Hilfe von Arbeitsproduktmodellen systematisch und strukturiert festgehalten werden. „Unterstützung durch Arbeitsproduktmodell“ bildet ab, wie umfassend und detailliert ein Vorgehensmodell den Umgang mit großen Informationsmengen und deren wechselseitigen Abhängigkeiten regelt. „Umfassend“ bedeutet, daß das Dokumentieren von viele Informationen verschiedener Projektthemenbereiche unterstützt wird. Dies kann auch Redundanzen mit sich bringen. „Detailliert“ heißt, daß das strukturierte und präzise Dokumentieren von Informationen unterstützt wird. Je detaillierter die Beschreibung eines Arbeitsproduktes in einem Vorgehensmodell ist, desto leichter ist es durch die Projektmitarbeiter benutzbar.
Beispiel: Das V-Modell XT [VMX06] definiert Arbeitsprodukte (dort „Produkte“) wie *QS-Handbuch*, *Vertrag* oder *SW-Modul* in den entsprechenden Projektthemenbereichen.

Motivation der „Unterstützung durch Arbeitsproduktmodell“:
Bedeutung für Softwareprojekte. In Projekten müssen oft sehr viele Informationen wie Anforderungen, technische Restriktionen, kaufmännische und organisatorische Plan- und Ist-Zahlen verarbeitet, verwaltet und in Zusammenhang gesetzt werden. Diese Informationen weisen Abhängigkeiten auf. Beispielsweise hat der Umfang der Anforderungen Auswirkungen auf die Projektlaufzeit, den Ressourcen- und Budgetbedarf, aber auch auf die Anforderungen an die technische Architektur einer Anwendung. Projektmitarbeiter können nicht beliebig viele Informationen im Gedächtnis behalten [Boe04]. Sie müssen übersichtlich, strukturiert und präzise dargestellt und ihre Abhängigkeiten explizit gemacht werden. So können Projekte überschaubar gehalten, Festlegungen bei Bedarf gezielt wieder aufgefunden und die Seiteneffekte von Entscheidungen und Veränderungen abgeschätzt werden. Komplizierte Informationszusammenhänge können mit Hilfe von Arbeitsproduktmodellen dokumentiert und strukturiert werden.
Beispiel: Der Zustand der Merkmale *EPD-Einf-DurchUmfangFunktionen*, *EPSWE-Einf-DurchUmfangTechnischeAnforderungenUndQualitätskriterien* aus Kapitel 2 kann durch die Unterstützung eines Arbeitsproduktmodells in den Projektthemenbereichen „Requirements Engineering“ und „Software Engineering“ verbessert werden.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Vorgehensmodelle unterstützen ihre Benutzer unterschiedlich durch Arbeitsproduktmodelle. Das V-Modell XT [VMX06] und der RUP [RUP06] beschreiben sehr umfassende Arbeitsproduktmodelle, Extreme Programming [Bec00] und Scrum [Sch02] definieren diesen Aspekt nur ansatzweise (siehe Kapitel 3.3).
Beispiel: Scrum [Sch02] beschreibt die Arbeitsprodukte *Product Backlog*, *Release Backlog, Sprint Backlog* und *Backlog Graph.* Das V-Modell XT [VMX06] beschreibt mehr als einhundert Produkte.

Unterstützung durch Prinzipien

Definition der „Unterstützung durch Prinzipien“:
Prinzipien (siehe auch Glossar) sind ein Beschreibungselement von Vorgehensmodellen (siehe auch [Chr92]). Sie bieten grundlegenden, übergeordneten Hilfen zur Orientierung, Entscheidungsunterstützung, Vorgehens- und Ergebnisüberprüfung während der Arbeit in einem Projekt. Prinzipien weisen definitionsgemäß keine hinreichend expliziten Abhängigkeiten auf, um in Modellen zusammengefasst zu werden. Unterstützung steht hier für die umfassende und detaillierte Beschreibung von Prinzipien durch Vorgehensmodelle. Umfassend beschreibt die Unterstützung aller relevanten Projektthemenbereiche durch Prinzipien. Detailliert steht für ihre jeweilige Anwendbarkeit.
Beispiel: Extreme Programming [Bec00] beschreibt das Prinzip *Open Honest Communication* (siehe Kapitel 3.3). Dies ist in den Projektthemenbereich „Menschenführung einzuordnen“.

Motivation der „Unterstützung durch Prinzipien“:
Bedeutung für Softwareprojekte. Nicht jede Entscheidung in einem Projekt kann in einem Vorgehensmodell vorausgedacht, nicht alle Eventualitäten vorhergesehen und durch Aktivitäten ausgedrückt und konkret geregelt werden. Durch Prinzipien kann ein Vorgehensmodell eine grundsätzliche, universelle Entscheidungsunterstützung für die Arbeit im Projekt bieten. Prinzipien lassen den Benutzer eines Vorgehensmodells mehr Entscheidungsspielräume und schränken ihn weniger ein als konkrete Aktivitäten, Rollen und Arbeitsprodukte. Aufgrund ihres generischen Charakters sind Prinzipien schwieriger anzuwenden als Aktivitäten, Rollen und Arbeitsprodukte. Sie erfordern vom Anwender eines Vorgehensmodells einen größeren Transferschritt.
Beispiel: Das Prinzip *Open Honest Communication* in Extreme Programming [Bec00] unterstützt den Projektthemenbereich „Menschenführung“. Dies verbessert den Zustand des Merkmals *EPM-BedDeck-KommunikationImTeam* aus Kapitel 2.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. In den Agilen Methoden wird mehr Inhalt über Prinzipien ausgedrückt, um den Mitarbeitern Entscheidungsspielräume zu überlassen. Die eher plan- oder prozeßgetriebenen Ansätze formulieren weniger Prinzipien als konkrete Anweisungen.
Beispiel: Extreme Programming [Bec00] beschreibt Prinzipien wie *Local Adaptions, Travel Light, Self-Similarity, Improvement* [Bec04] (siehe Kapitel 3.3), das V-Modell XT [VMX06] dagegen enthält keine generischen Prinzipien als Modellelemente.

Unterstützung durch Methoden

Definition der „Unterstützung durch Methoden“:
Methoden (siehe auch Glossar) sind ein Beschreibungselement von Vorgehensmodellen [Chr92]. Sie weisen typischerweise keine hinreichend expliziten Abhängigkeiten auf, um sie zu Methodenmodellen zusammenzufassen. Unterstützung steht für die umfassende und detaillierte Beschreibung von Methoden durch ein Vorgehensmodell. Umfassend beschreibt die Überdeckung der relevanten Projektthemenbereiche, detailliert die konkrete Anwendbarkeit der Methodenbeschreibungen. Je detaillierter die Methoden in einem Vorgehensmodell beschrieben sind, desto leichter sind sind von Projektmitarbeitern anwendbar.
Beispiel: In Extreme Programming [Bec00] wird die *Practice Sit together* beschrieben (siehe Kapitel 3.3). Sie hat Einfluß im Rahmen des Projektthemenbereichs „Menschen-

führung“.

Motivation der „Unterstützung durch Methoden“:
Bedeutung für Softwareprojekte. Methoden bieten konkret umsetzbare Anweisungen zur Bearbeitung dezidierter und isolierter Problemstellungen in Projekten. Sie sind unabhängig von einem eingesetzten Entwicklungsprozeß. Je konkreter sie beschrieben sind, desto einfacher sind sie vom Anwender eines Vorgehensmodells einsetzbar.
Beispiel: Die Methode (hier *Technique*) *Pair Programming* in Extreme Programming [Bec00] unterstützt den Projektthemenbereich Software Engineering. Dies beeinflusst den Zustand des Merkmals *EPSWE-BedDeck-Fähigk-SoftwareEngineering* aus Kapitel 2.
Bedeutung für die Eignungseinstufung von Vorgehensmodellen. Eher menschengetriebene Ansätze wie die Agilen Prozesse beschreiben tendenziell mehr Methoden. Dies läßt den Projektmitarbeitern mehr Entscheidungsspielräume über den Methodeneinsatz. Prozeßgetriebene Ansätze wie das V-Modell XT [VMX06] sind mitunter methodenneutral.
Beispiel: Extreme Programming [Bec00] beschreibt die *Techniques On-Site Customer, Planning Game, Metaphor, Short Releases, Testing, Simple Design, Refactoring, Pair Programming, Collective Ownership, Continuous Integration, Coding Standards* und *40 Hour Week* und die
Practices Sit together, Whole Team, Informative Workspace, Energized Work, Stories, Weekly Cycle, Quarterly Cycle, Slack, Ten-Minute-Build, Incremental Design, Real Customer Involvement, Incremental Deployment, Team Continuity, Shrinking Teams, Root-Cause-Analysis, Shared Code, Code and Tests, Single Code Base, Daily Deployment, Negotiated Scope Contract, Pay-Per-Use [Bec04] (siehe Kapitel 3.3).

Anhang A4.Ontologie zur Herleitung der Merkmale

Die Herleitung und systematische Anordnung der Merkmale wurde durch eine Ontologie repräsentiert, welche in den vorhergehenden Kapiteln hergeleitet und auf der folgenden Seite vollständig dargestellt wird.
Die Herleitung der Ontologiebezeichner wird hier noch einmal im Zusammenhang gezeigt:

Einstufung wird abgekürzt mit „E“. Einstufungen gibt es für
- Projektthemenbereiche (abgekürzt mit „P“),
- Risiken (abgekürzt mit „R“) und
- Vorgehensmodelle (abgekürzt mit „V“).
- Von den Projektthemenbereichen gibt es
  - Domänenwissen (abgekürzt mit „D“),
  - Softwareentwicklung (abgekürzt mit „S“)
  - Projektmanagement (abgekürzt mit „P“) und
  - Menschenführung (abgekürzt mit „M“).
  - Unterhalb der Projektthemenbereiche wird
  - Bedarfsdeckung Ressourcen mit „BedDeck“ abgekürzt,
  - Einfachheit mit „Einf“.
  - Unterhalb der Ressourcen werden
    - Fähigkeiten mit „Fähigk“ und
    - Kapazität mit „Kap“ abgekürzt.
- Unterhalb der Risiken (bzw. Risikobereiche) gibt es
  - Entwicklungs-Produkt,
  - Entwicklungs-Team,
  - Entwicklungs-Umfeld und
  - Entwicklungs-Prozeß.
- Unter den Vorgehensmodellen gibt es die
  - Unterstützung von Projektthemenbereichen („UnterstProjTheBer“) und die
  - Flexibilität (abgekürzt mit „Flexi“).
  - Bei der Unterstützung der Projektthemenbereiche gibt es
    - „Requirements Engineering“,
    - „Software Engineering“,
    - „Software Management“,
    - „Projektmanagement“ und
    - „Menschenführung“.
  - Bei der Flexibilität gibt es die
    - „IterativitätUndInkrementalität“.

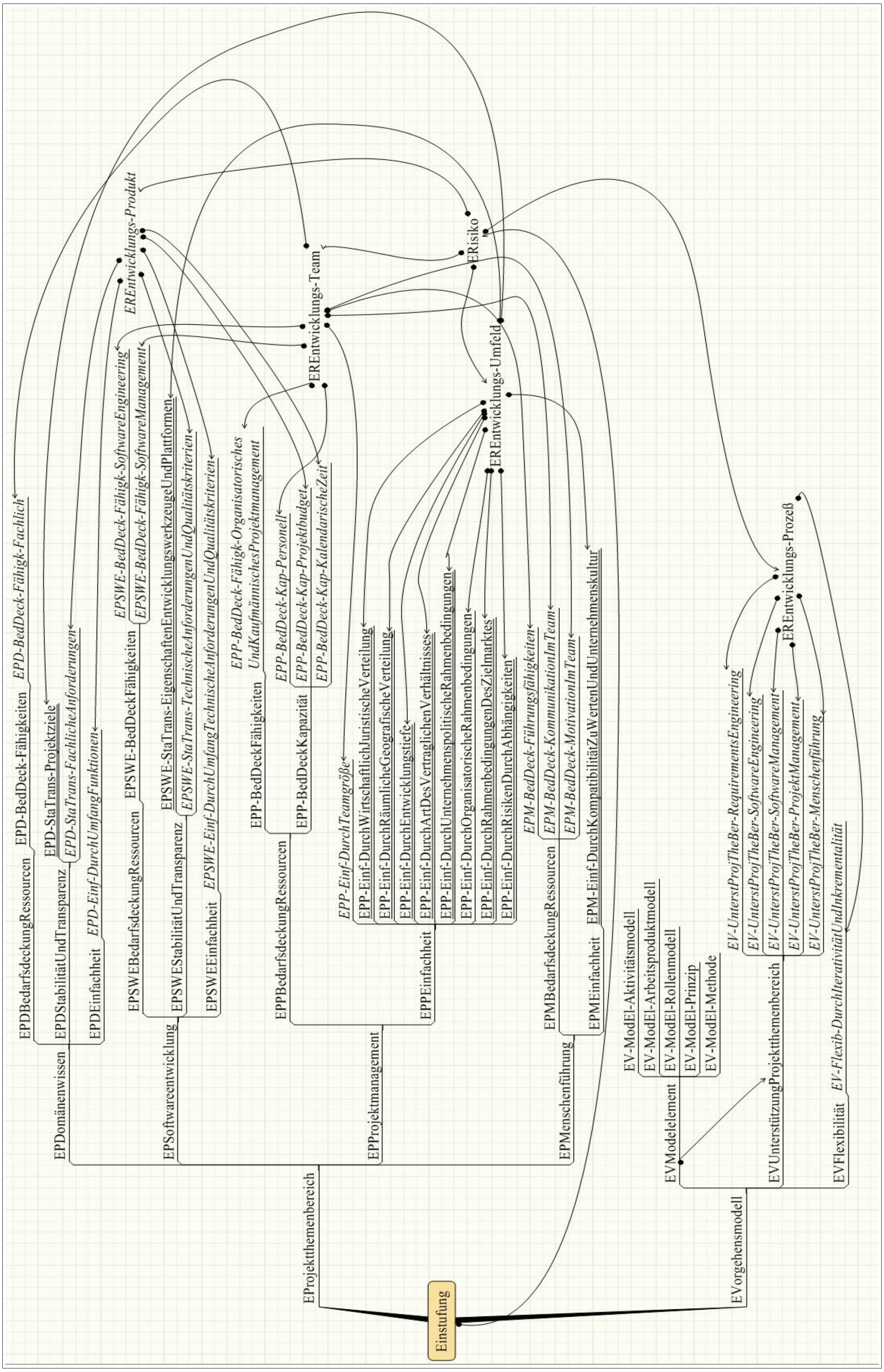

Abbildung 43 Gesamtontologie

Anhang A5.Fragen zur Erhebung des State of the Art

- Werden in den Unternehmen, die Sie besucht haben, Vorgehensmodelle (wie etwa V-Modell XT, Rational Unified Process, aber auch Extreme Programming oder Scrum) eingesetzt?
- Wenn ja, welche Vorgehensmodelle werden eingesetzt?
- Wie werden sie gegebenenfalls eingesetzt (projektweise ausgewählt oder als Organisationsstandard)?
- Wie werden sie gegebenenfalls ausgewählt?
- Welche Branchen (Systemhäuser, Softwareprodukthersteller, Sekundärbranche (Finance, Automotive etc.)) setzen gegebenenfalls welche Vorgehensmodelle ein?
- Wie viel tragen Vorgehensmodelle gegebenenfalls zu bedarfsgerechten Entwicklungsprozessen bei?

Anhang A6.Überblick Beeinflussungsbeziehungen

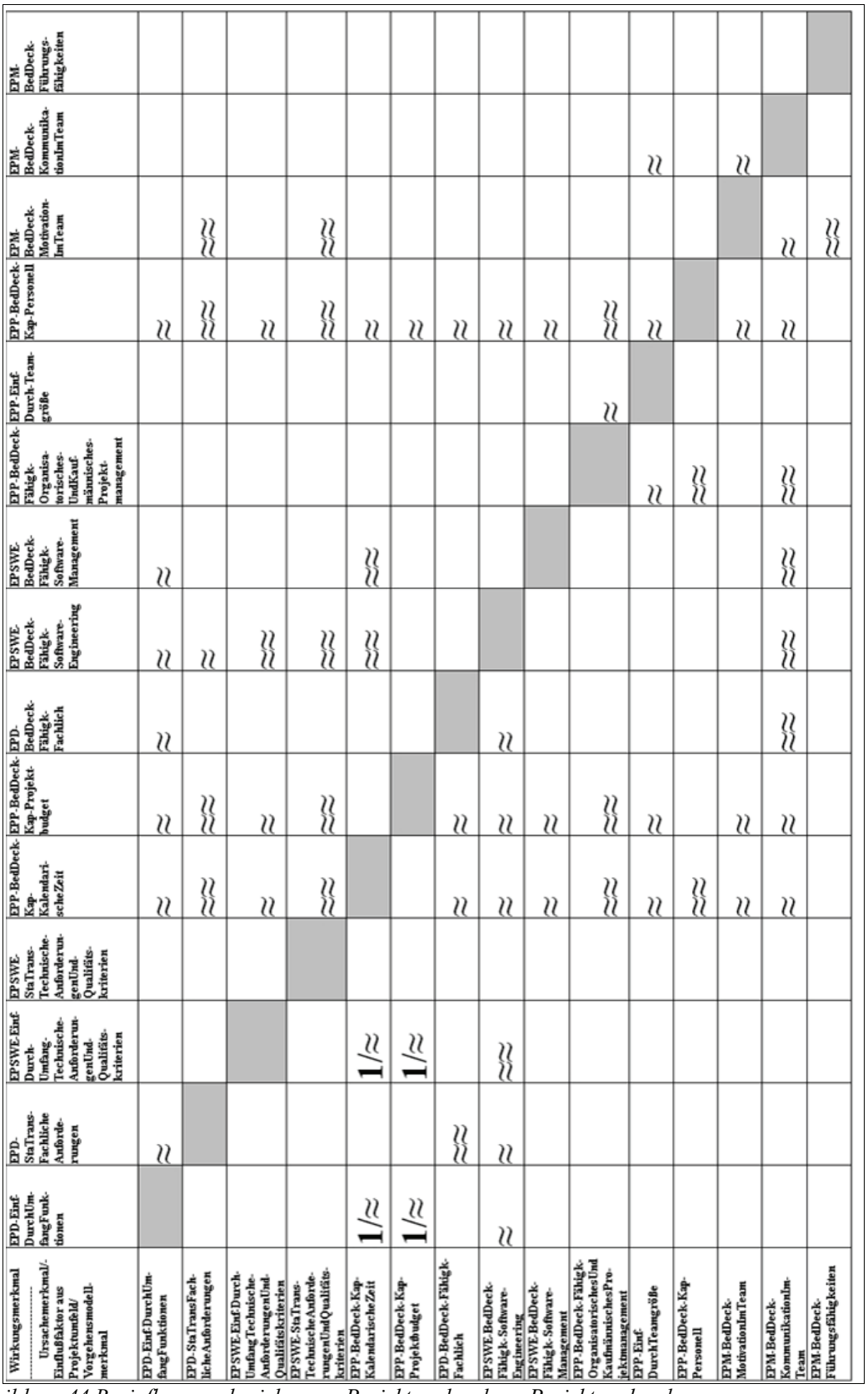

Abbildung 44 Beeinflussungsbeziehungen Projektmerkmale zu Projektmerkmalen

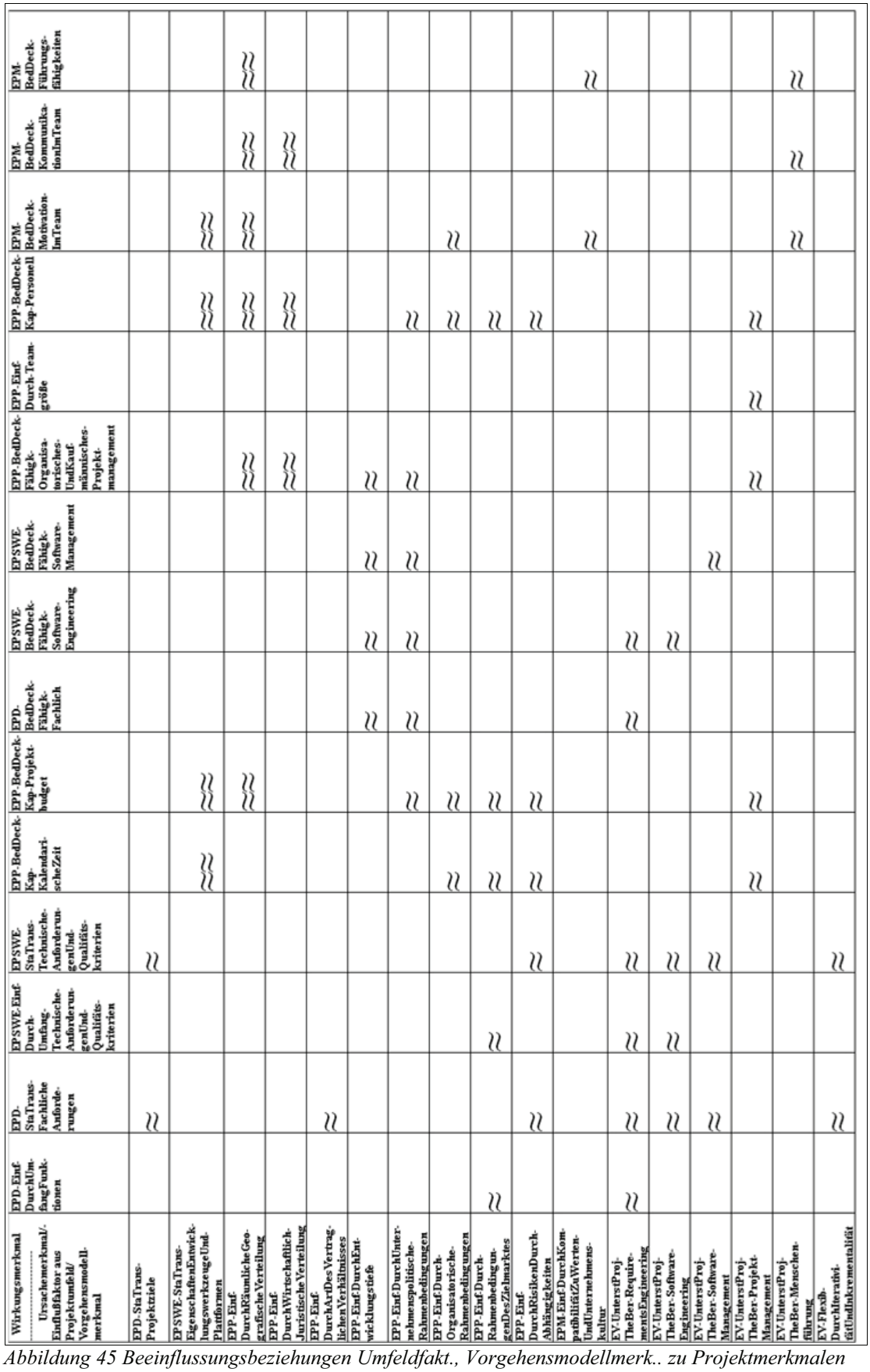

Abbildung 45 Beeinflussungsbeziehungen Umfeldfakt., Vorgehensmodellmerk.. zu Projektmerkmalen

Anhang A7.Glossar

Für die vorliegende Arbeit waren umfangreiche Begriffsdefinitionen erforderlich. Um die Aussagen der Arbeit nachvollziehen zu können ist es daher unerläßlich, sich mit der verwendeten Begriffswelt vertraut zu machen. Dieses Glossar definiert das Begriffssystem, welches in der vorliegenden Arbeit verwendet wird. Es enthält Basisdefinitionen, auf die in der Arbeit mit weiterführenden Definitionen aufgebaut wird. Das Glossar erhebt keinen hierüber hinaus gehenden Gültigkeitsanspruch.

Begriff (Akronym)	Bedeutung (Synonym)
Änderbarkeit	Diese umfasst im einzelnen: Analysierbarkeit: „Aufwand, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedürftige Teile zu bestimmen“ Modifizierbarkeit: „Aufwand zur Ausführung von Verbesserungen, zur Fehlerbeseitigung oder Anpassung an Umgebungsveränderungen“ Stabilität: „Wahrscheinlichkeit des Auftretens unerwarteter Wirkungen von Veränderungen“ Prüfbarkeit: „Aufwand, der zur Prüfung der geänderten Software notwendig ist“ entsprechend DIN ISO 9126 zitiert nach [Bal98]
Agiler Ansatz	Intuitive, nicht trennscharfe Bezeichnung für eine Gruppe von Softwareentwicklungsansätzen, die sich Minimalisierung, Flexibilität, Nutzenorientierung zum Ziel setzen. Sie sind *menschengetrieben*, streben Projekterfolg mit Hilfe menschlicher Fähigkeiten an und und adressieren vorwiegend *Unsicherheiten* im *Projekt*. Sie haben tendenziell reaktiven Charakter und bauen stärker auf das Hinterkopfwissen der Projektmitarbeiter [Boe04]. Sie sind eher für kleine, nicht verteilte Projekte in instabilen Umfeldern geeignet [Boe04].
Aktivität	Methodenunabhängige Beschreibung einer Tätigkeit zur Bearbeitung eines *Arbeitsproduktes* in einem *Softwareentwicklungsprojekt*. Sie kann aus einzelnen Schritten bestehen [OMG06]. Aktivitäten können Vor- und Nachbedingungen aufweisen. Für ihre Ausführung können verschiedene *Methoden* benutzt werden. Aus Gründen der Vereinfachung wurde auf das eigentlich korrekte „Aktivitätstyp“ verzichtet, da in dieser Arbeit nur Beschreibungen von Tätigkeiten, aber nicht Tätigkeiten (beispielsweise mit konkreten Aufwänden hierfür in einem Projekt) selbst diskutiert werden.
Analyse	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der Darstellung der *fachlichen Anforderungen* in einer IT-näheren, aber lösungsneutralen Darstellung, oft als Modell.
Anforderung	Legen die qualitativen und quantitativen Eigenschaften eines Produktes aus der Sicht des Auftraggebers fest [Bal96]. Der Begriff Auftraggeber wird zu *Stakeholder* verallgemeinert. Die Eigenschaften können *funktionale, nicht funktionale, fachliche* und *technische Anforderungen* sein

Begriff (Akronym)	Bedeutung (Synonym)
Arbeitsprodukt	Beschreibung eines Resultats (Artefakt, Deliverable, Ergebnis im Sinne von [Chr92]) von Aktivitäten in einem *Softwareentwicklungsprojekt*. Ein Arbeitsprodukt ist ein Teilresultat des Projektergebnisses, es kann ein Dokumententemplate mit kommentierten Gliederungspunkten oder die Beschreibung der Inhalte eines Modells sein. Es dient der Aufnahme von Informationen. Aus Gründen der Vereinfachung wurde auf das eigentlich korrekte „Arbeitsprodukttyp“ verzichtet, da in dieser Arbeit nur Beschreibungen von Resultaten, aber nicht Resultate (beispielsweise mit konkreten Inhalten aus einem Projekt) selbst diskutiert werden.
Beeinflussungsbeziehung	Qualifizierbarer, definierter Ursache-Wirkungs-Zusammenhang zwischen *Merkmalen* oder zwischen *Fragen zu Einflußfaktoren aus dem Projektumfeld* und *Merkmalen* die in einem *Ursache-Wirkungs-Diagramm* modelliert werden können.
Benutzbarkeit	Diese umfasst im einzelnen: Verständlichkeit: „Aufwand für den Benutzer, das Konzept und die Anwendung zu verstehen“ Erlernbarkeit: „Aufwand für den Benutzer, die Anwendung zu erlernen (z.B. Bedienung, Ein-, Ausgabe)“ Bedienbarkeit: „Aufwand für den Benutzer, die Anwendung zu bedienen“ entsprechend DIN ISO 9126 zitiert nach [Bal98].
Change Management	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der kontrollierten und konsistenten Veränderung der *Arbeitsprodukte* des *Projektes*
Design	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der Weiterentwicklung der *Arbeitsprodukte* der *Analyse,* in dem technische Lösungsaspekte, meist aber noch unabhängig von der Implementierungssprache hinzugefügt werden.
Effizienz	Diese umfasst im einzelnen: Zeitverhalten: „Antwort- und Verarbeitungszeiten sowie Durchsatz bei der Funktionsausführung“ Verbrauchsverhalten: „Anzahl und Dauer der benötigten Betriebsmittel für die Erfüllung der Funktionen“ entsprechend DIN ISO 9126 zitiert nach [Bal98]
Eignungseinstufung	Bestimmung und Vergleich des Nutzens des Einsatzes eines konkreten *Vorgehensmodells* für Softwareentwicklungsprojekte in einem konkreten Projekt
Einflußfaktor	Als wesentlich für den Zustand und die Zustandsveränderung einer Problemstellung beurteilte Größe, die eine Vorlage für ein *Merkmal* ist
Engineering-orientiert	Tendenziell auf die Erzeugung von Engineering Arbeitsprodukten ausgerichtet
Ergebnis	Resultat eines Projektes

Begriff (Akronym)	Bedeutung (Synonym)
Etablierter Ansatz	Intuitive, nicht trennscharfe Bezeichnung für Entwicklungsansätze, die sich nicht den Agilen Methoden zuordnen. Sie sind eher *prozeßgetrieben*, streben Projekterfolg durch Prozeßqualität an und adressieren *Kompliziertheit.* Sie haben tendenziell proaktiven Charakter und beanspruchen weniger das Hinterkopfwissen der Projektmitarbeiter [Boe04]. Sie sind eher für große Projekte in stabilem Umfeld geeignet [Boe04]. (Synonym: Plan-driven [Boe04])
EVGM	*Verfahren* zur *Eignungseinstufung* für *Vorgehensmodelle* für Softwareentwicklungsprojekte
evolutionär	Entwicklungsansatz, bei dem auch grundlegende Projektziele und Anforderungen während der Projektlaufzeit verändert werden
Fachliche Anforderung	Fachliche *Anforderungen* beschreiben, welche domänenrelevanten Eigenschaften und Funktionen durch ein zu entwickelndes Softwaresystem abgedeckt werden soll. Sie spezifizieren, was die Software leisten soll. Fachliche Anforderungen beschreiben den Zweck eines Softwaresystems.
Fähigkeit	Wissen, Fertigkeiten, Erfahrung bezüglich eines *Projektthemenbereichs* in einem *Softwareentwicklungsprojekt*
Funktionale Anforderung	Funktionale *Anforderungen* sind für ein Softwaresystem verlangte Eigenschaften, die direkt einen Arbeitsschritt unterstützen und dem Systemzweck dienen. Funktionale Anforderungen beziehen sich auf dezidierte Teile des zu entwickelnden Softwaresystems und sind oft isoliert darstellbar.
Funktionalität	Diese umfasst im einzelnen: Richtigkeit: „Liefern der richtigen oder vereinbarten Ergebnisse oder Wirkungen, z.B. die benötigte Genauigkeit von Werten“ Angemessenheit: „Eignung der Funktionen für spezifizierte Aufgaben, z.B. aufgabenorientierten Zusammensetzung von Funktionen aus Teilfunktionen“ Interoperabilität: „Fähigkeit, mit vorgegebenen Systemen zusammenzuwirken“ Ordnungsmäßigkeit [sic!]: „Erfüllung von anwendungsspezifischen Normen Vereinbarungen, gesetzlichen Bestimmungen und ähnlichen Vorschriften“ Sicherheit: „Fähigkeit, unberechtigten Zugriff, sowohl versehentlich als auch vorsätzlich, auf Programme und Daten zu verhindern“ entsprechend DIN ISO 9126 zitiert nach [Bal98]
Implementierung	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der Entwicklung des ablauffähige Codes. Hierbei wird entweder manuell programmiert oder auch Code generiert.
Inkrement	Arbeitsprodukte einer *Iteration* in einem definierten Zustand
Iteration	Sequentielles Ausführen von *Aktivitäten* verschiedener Disziplinen im *Projekt*
Kardinalskala	Anordnung von skalaren, zustandsbehafteten, gleichartigen Größen, die in einer Verhältnis-Relation stehen und auf denen gerechnet werden kann

Begriff (Akronym)	Bedeutung (Synonym)
Kausal gegenläufig	Eigenschaft einer *Beeinflussungsbeziehung*, bei der eine Zustandsverbesserung des Ursache-*Merkmals* eine Zustandsverschlechterung des Wirkungs-*Merkmals* bewirkt und eine Zustandsverschlechterung des Ursache-*Merkmals* eine Zustandsverbesserung des Wirkungs-*Merkmals* bewirkt
Kausal nebenläufig	Eigenschaft einer Beeinflussungsbeziehung, bei der eine Zustandsverbesserung des Ursache-*Merkmals* eine Zustandsverbesserung des Wirkungs-*Merkmals* bewirkt und eine Zustandsverschlechterung des Ursache-*Merkmals* eine Zustandsverschlechterung des Wirkungs-*Merkmals* bewirkt
Kommunikation	Verständigung, wechselseitige Mitteilung [Dud91]
Komplexität	Mangelnde Überschau- und Beherrschbarkeit einer Situation wegen *Kompliziertheit* und *Unsicherheit (*in Anlehnung an [Gom99])
Kompliziertheit	Mangelnde Überschau- und Beherrschbarkeit einer Situation wegen einer großen Anzahl an *Einflußfaktoren* und *Beeinflussungsbeziehungen* (in Anlehnung an [Gom99])
Konfigurationsmanagement	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der Verwaltung und Konsistenthaltung der *Arbeitsprodukte* des *Projektes*
Management-orientiert	Tendenziell auf die Verwaltung von Engineering *Arbeitsprodukten* und die Erzeugung von Verwaltungs-*Arbeitsprodukten* ausgerichtet
menschengetrieben	Denkweise, bei der tendenziell die Mehrzahl von Entscheidungen menschlichen *Fähigkeiten* überlassen wird
Metamodell	Präzise und zielführende Abstraktion von einer Klasse von konkreten *Modellen*, ein *Modell* von *Modellen*
Methode	Eine systematische Handlungsvorschrift, um Aufgaben einer bestimmten Klasse zu lösen. (bei Gerisch M., Schuhmann J.: Software-Entwurf, Rudolf Müller Verlag 1998 zitiert nach [Chr92]). Präzises, konkretes Schritt-für-Schritt-Verfahren zur Lösung eines dezidierten Problems. (Synonyme: Vorgehensweise, Technik, Praktik, Verfahren, Konzept)
Modell	Präzise und zielführende Abstraktion von einer Klasse von konkreten Gegenständen oder Begrifflichkeiten, die als wesentlich erachtete Eigenschaften berücksichtigt und die anderen vernachlässigt
Motivation	Summe der Beweggründe, Antrieb zum Handeln [Dud91]
Nicht funktionale Anforderung	Nicht funktionale *Anforderungen* sind Rahmenrestriktionen, die bei der Erfüllung der *funktionalen Anforderungen* nicht verletzt werden dürfen. Nicht funktionale Anforderungen beziehen sich oft auf das gesamte zu entwickelnde Softwaresystem und haben Querschnittscharakter.
Nominalskala	Anordnung von skalaren, zustandsbehafteten, gleichartigen Größen, die in einer „<>“ Relation stehen.
Ordinalskala	Anordnung von skalaren, zustandsbehafteten, gleichartigen Größen, die in einer „<“ Relation stehen.

Begriff (Akronym)	Bedeutung (Synonym)
Prinzip	Allgemeingültiger Grundsatz, der aus der Verallgemeinerung von Gesetzen und wesentlichen Eigenschaften der objektiven Realität abgeleitet ist und als Leitfaden dient. (bei Gerisch M., Schuhmann J.: Software-Entwurf, Rudolf Müller Verlag 1988 zitiert nach [Chr92]). Übergreifend gültiger Grundsatz zur Orientierung und Entscheidungsunterstützung. Prinzipien sagen nicht aus, wie man sie erreichen kann [Bal96]. (Synonym: Wert im Sinne einer Wertvorstellung, Richtlinie, Leitsatz)
Projekt	„Ein Vorhaben, das im wesentlichen durch eine Einmaligkeit der Bedingungen in ihrer Gesamtheit gekennzeichnet ist“ bei DIN69 901 zitiert nach [Lid95] Abgrenzung: Die hier auch gebräuchlichen Aussagen über definiertes Ziel, sowie fixierten Zeit- und Kostenrahmen lassen sich speziell für Projekte, die mit evolutionären Ansätzen durchgeführt werden, nicht halten
Projektthemenbereich	*Aktivitäten*, *Rollen*, *Arbeitsprodukte*, *Prinzipien* und *Methoden* mit einem gemeinsamen Thema. Sie wird im Gegensatz zu Phasen *temporal nebenläufig* ausgeführt (in Anlehnung an [OMG06]).
Projekterfolgsfaktor	In der Literatur belegter *Einflußfaktor*, dessen Zustand als förderlich für den Projekterfolg angesehen wird
Projektrahmenbedingungen	Organisatorische, juristische, wirtschaftliche, kulturelle, sozio-emotionale, politische und personelle *Einflußfaktoren*, Gegebenheiten und Restriktionen, die ein *Projekt* von außen beeinflussen und nicht seinen inneren Zustand repräsentieren
Projektrisikofaktor	In der Literatur belegter, wesentlicher *Einflußfaktor*, dessen Zustand als gefährdend für den Projekterfolg angesehen wird
Projektziel	In der Zukunft liegend, vorstellbar und realistisch, nur durch aktives Handeln erreichbar, bewußt angestrebt, erwünscht [Kel94]. Ein Zustand, der durch ein *Projekt* auf organisatorischer oder Unternehmensebene erreicht werden soll, der Grund warum das *Projekt* gemacht wird.
prozeßgetrieben	Denkweise, bei der tendenziell die Mehrzahl von Entscheidungen durch die Festlegungen innerhalb von Prozessen unterstützt wird.
Qualitätskriterium	Überprüfbare und verbindliche Festlegung einer geforderten Eigenschaft
Qualitätsmanagement	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur proaktiven Definition von *Qualitätskriterien,* zur proaktiven Unterstützung des Erfüllens dieser *Qualitätskriterien* und zu ihrer reaktiven Überprüfung im Rahmen einer Qualitätsplanung, -kontrolle und -Steuerung für eine Organisation.

Begriff (Akronym)	Bedeutung (Synonym)
Qualitätssicherung	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur proaktiven Definition von *Qualitätskriterien* zur proaktiven Unterstützung des Erfüllens dieser *Qualitätskriterien* und zu ihrer reaktiven Überprüfung im Rahmen einer Qualitätsplanung, -kontrolle und -Steuerung für ein *Projekt*. Sie dient dazu, kontrolliert die an die Software geforderten *Qualitätskriterien* zu entwerfen, und deren Erfüllung durch konkrete Maßnahmen zu planen, zu unterstützen und zu überprüfen.
Qualitätsziel	Konkreter Wert eines *Qualitätskriteriums*
Requirements Engineering	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der Erhebung, Strukturierung und Spezifizierung der *fachlichen* und *technischen Anforderungen*
Requirements Management	Alle *Aktivitäten*, *Rollen, Arbeitsprodukte*, *Prinzipien* und *Methoden* zur Unterstützung der Verwaltung der *Arbeitsprodukte* des *Requirements Engineering* Das Requirements Management unterstützt und organisiert den kontrollierten Ablauf des *Requirements Engineerings*. Eine Tätigkeit im Rahmen des Requirements Managements kann auch sein, *Anforderungen* zu priorisieren und zu bewerten.
Risikobereich	Von *Projektthemenbereichen* unabhängige Abgrenzung und Zusammenfassung von Aspekten in einem *Softwareentwicklungsprojekt*, denen auftretende Probleme zugeordnet werden können
Rolle	Beschreibung eines Verantwortlichkeitsbereichs bezüglich *Aktivitäten* und *Arbeitsprodukten* in einem *Softwareentwicklungsprojekt*, sowie die hierfür geforderten *Fähigkeiten*. Aus Gründen der Vereinfachung wurde auf das eigentlich korrekte „Rollentyp“ verzichtet, da in dieser Arbeit nur Beschreibungen von Verantwortlichkeitsbereichen, aber nicht Verantwortlichkeitsbereiche (beispielsweise mit Besetzung einer konkreten Person in einem Projekt) selbst diskutiert werden. (Synonym wird auch Akteur verwendet).
Software Process Improvement (SPI)	Gesamtheit der Ansätze zur Verbesserung von *Softwareentwicklungsprozessen*
Softwareentwicklungsprozeß	Gesamtheit der Festlegungen bezüglich *Aktivitäten*, *Rollen*, *Arbeitsprodukten*, *Rollen*, *Methoden* und *Prinzipien* in einem *Softwareentwicklungsprojekt*
Softwareentwicklungsprojekt	*Projekt* zur Entwicklung von Software, keine Entwicklung von Hardware
Stabilität	Beständigkeit, Dauerhaftigkeit, Standfestigkeit [DUD91]. Wesentliche Informationen und Sachverhalte sind beständig gegen unerwünschte Veränderungen.
Stakeholder	Alle Personen, die von der Systementwicklung und vom Einsatz und Betrieb des Systems betroffen sind, zum Beispiel auch Benutzer, Betreiber oder Trainer [Rup01]
System	Organisiertes und abgegrenztes Wirkungsgefüge aus Elementen unterschiedlicher Funktionen, das einem dezidierten Zweck dient, Gegenstand der *Systemtheorie*

Begriff (Akronym)	Bedeutung (Synonym)
Systemtheorie	Paradigma eines gesamtheitlichen Verständnisses von komplexen Wirkungsgefügen, sogenannten *Systemen*
Merkmal	Definierte und für die Modellierung in einem *Ursache-Wirkungs-Diagramm* zur Auswahl stehende Größe mit bewußt nicht definiertem Zustand, entsteht durch konsolidieren und systematisieren von *Einflußfaktoren* aus der Literatur. Alle *Merkmale* haben explizite *Beeinflussungsbeziehungen* mit anderen *Merkmalen*. *Merkmale* beschreiben Eigenschaften von *Projekten* und *Vorgehensmodellen*.
Technische Anforderung	Technische *Anforderungen* beschreiben die technischen Rahmenrestriktionen, die bei der Erfüllung der *fachlichen Anforderungen* nicht verletzt werden dürfen, also wie ein zu entwickelndes Software etwas leisten soll. Technische Anforderungen sind konkretisierte *Qualitätskriterien* für ein Softwaresystem.
Technologie	Eigenschaften der Entwicklungs- und Ziel- bzw. Produktivplattform einer zu entwickelnden Software.
Temporal nebenläufig	Im selben Zeitraum ausgeführt, muß nicht synchron sein, nicht sequentiell
Time Box	*Iteration* mit fixierter Dauer, in der die Entwicklung eines definierten und vom Team zugesagten Funktionsumfangs bei unveränderlichen *Anforderungen* erfolgt. Sie ist nicht unterbrechbar und nur durch das Team abbrechbar.
Transparenz	Deutlichkeit, Verstehbarkeit [DUD91], im Sinne von Erkennbarkeit. Wesentliche Informationen und Sachverhalte stehen dem Projektteam rechtzeitig und im ausreichenden Maße zur Verfügung, Gegenteil von Intransparenz. (in Anlehnung an [Doe89])
Übertragbarkeit	Diese umfasst im einzelnen: Anpaßbarkeit: „Möglichkeiten, die Software an verschiedene, festgelegte Umgebungen anzupassen, wenn nur Schritte unternommen oder Mittel eingesetzt werden, die für diese Zweck für die betrachtete Software vorgesehen sind“ Installierbarkeit: „Aufwand, der zum Installieren der Software in einer festgelegten Umgebung notwendig ist“ Konformität: „Grad, in dem die Software Normen oder Vereinbarungen zur Übertragbarkeit erfüllt“ Austauschbarkeit: „Möglichkeit, diese Software anstelle einer spezifizierten anderen in der Umgebung jener Software zu verwenden, sowie der dafür benötigte Aufwand“ entsprechend DIN ISO 9126 zitiert nach [Bal98].
Unsicherheit	Mangelnde Überschau- und Beherrschbarkeit einer Situation wegen mangelnder *Stabilität* und *Transparenz* von *Einflußfaktoren* und *Beeinflussungsbeziehungen*
Unterstützungsfrage	Dient zur Untersuchung der *Projektrahmenbedingungen* und des Projekt-Umfeldes, hat *Beeinflussungsbeziehungen* zu *Merkmalen*
Veränderungsmanagement	Handhabung der Umstellung von Unternehmensprozessen mit Fokus auf Soft Factors, in der vorliegenden Arbeit speziell für Software-Entwicklungsprozesse

Begriff (Akronym)	Bedeutung (Synonym)
Vorgehensmodell (VGM)	Referenzprozeß für *Softwareentwicklungsprozesse,* Summe der *Rollen*, *Arbeitsprodukte*, *Aktivitäten*, *Methoden, Prinzipien* und deren Abhängigkeiten über relevante Disziplinen eines *Projektes* und in einer Vorschrift zur Entwicklung von Software, ggf. ergänzt um Konzepte und Notationen
Vorgehensmodell-merkmal	Charakteristikum von Referenzprozessen für die Softwareentwicklung, die durch *Aktivitäten*, *Arbeitsprodukte*, *Rollen*, *Prinzipien* oder *Methoden* und deren Anordnung und Zusammenhang beschrieben ist und die *Projektmerkmale* adressiert. Ein Beispiel ist die Flexibilisierung durch *Iterativität und Inkrementalität*.
Wert	„Auffassungen von wünschenswertem, ... die explizit oder implizit für einen Einzelnen oder eine Gruppe kennzeichnend sind und die Auswahl der zugänglichen Weisen, Mittel und *Ziele* des Handelns beeinflussen.“ bei Kluckhohn zitiert nach [Ros03]
Zuverlässigkeit	Diese umfasst im einzelnen: Reife: „Geringe Versagenshäufigkeit durch Fehlzustände“ Fehlertoleranz: „Fähigkeit, ein spezifiziertes Leistungsniveau bei Software-Fehlern oder Nicht-Einhaltung ihrer spezifizierten Schnittstelle zu bewahren“ Wiederherstellbarkeit: „Fähigkeit, bei einem Versagen das Leistungsniveau wiederherzustellen und die direkt betroffenen Daten wiederzugewinnen. Zu berücksichtigen sind die dafür benötigte Zeit und der benötigte Aufwand“ entsprechend DIN ISO 9126 zitiert nach [Bal98].

Tabelle 18 Glossar der Arbeit